Installing Python 3 and Python Packages

Learn how to install Python packages and download Python 3 with Anaconda and Miniconda on Mac and Windows.

Share

Learning a new programming language, such as Python, on the Codecademy platform is super convenient and a ton of fun! Sooner or later the time will come when you desire to build your own projects and continue learning on your own computer. If you are ready to take things to the next level and can invest about an hour of your time installing and setting up your own Python development environment, then this article is for you!

By the end of this article, we will understand the difference between the standard Python distribution and alternatives, such as Anaconda and Miniconda, and how to install each on Mac and Windows. We will also learn what a package manager is and how to install packages using `pip` and `conda`. Lastly, we will install the Jupyter Notebook package so we can create and share interactive Python notebooks.

If you have a Chromebook, we have a separate article to teach you how to install Python 3 as well as an article to teach you how to install Jupyter Notebook on your Chromebook device.

Prerequisites

Before we begin, this article assumes that you are familiar with the basics of the command line and can execute simple commands just as moving around between files and folders in your file system. If needed, be sure to checkout the free Learn the Command Line lesson for a quick refresher!

What is Python?

At its heart, Python is a general purpose programming language that is often regarded as both easy to learn and very user-friendly. While Python is great for scripting and writing “glue” code to integrate existing components together, it is also a powerful full-fledged language that is capable of solving pretty much any software problem. Popular applications of Python include web applications, data analysis, and machine learning.

What is a Python Distribution?

While Python itself is just a programming language, a Python distribution bundles the core language with various other libraries and packages generally geared toward a specific problem domain (such as Data Analytics or Web Development). The standard Python distribution is released on python.org and includes the Python standard library as well as the package manager pip.

Though the list of Python distributions is always growing, two other popular distributions are Anaconda and Miniconda. The next section introduces both of these distributions, the pros and cons for each, and a recommendation on which to install.

Anaconda vs. Miniconda

Anaconda is an open source Python distribution that is purpose built for data science, machine learning, and large-scale data processing. It includes the core Python language, over 1,500 data science packages, a package management system called conda, IPython (an interactive Python interpreter), and much more. While it is a very comprehensive distribution, it is also quite large and therefore can take a while to download and consumes a lot of disk space.

Miniconda on the other hand, is a slimmed down version of Anaconda and includes all of the same components except for the pre-installed 1,500 data science packages. Instead, we can simply install these packages individually as needed using conda (the Anaconda/Miniconda package manager). We essentially get all the benefits of Anaconda, but without the hassle and burden of the large size. Feel free to use either distribution, however Miniconda is probably the better choice for getting set up with the least amount of fuss.

What is a Package Manager?

While Python the language is a great tool, a huge benefit of the Python is the large ecosystem and plethora of reusable packages (sometimes called libraries) that other developers maintain and share for anyone’s use. A package manager is the tool by which packages can be downloaded, installed, and managed within your projects. It is also responsible for keeping track of package versions and *dependencies*, packages that we don’t install ourselves but are imported by the package we’re installing. Even though a package manager is a complex tool, they are very simple to use.

Similar to how there are many different Python distributions, there are also several different package managers available. Despite having several options, most package managers function in the same manner and use a simple command structure. Each Python distribution is usually bundled with a specific package manager. Some of the more common ones are `pip` and `conda`.

Which Python Distribution Should I Install?

Good question! Now that the differences between Python distributions is clear, you are probably wondering which is the best distribution for you to install. There really is no right answer here, since each distribution is designed for varying needs. If you are interested in data science or machine learning then starting with Miniconda is a good choice. Otherwise, beginning with the standard Python distribution is a good choice for learning Python in general. You can always install the packages you need later.

Regardless of what you decide, keep in mind that you really only need to install a single Python distribution (and not several). So select your pick below and follow the corresponding installation instructions. Also keep in mind that there are a handful of different ways to install each of these Python distributions, however the instructions below are the easiest way to get started!

Installing Python

Mac

Most modern versions of MacOS come pre-installed with Python 2, however Python 3 is now the standard and should be installed as well. Python 3 can be installed using the official Python 3 installer.

1. Go to the Python Releases for Mac OS X page and download the latest stable release macOS 64-bit/32-bit installer.
2. After the download is complete, run the installer and click through the setup steps leaving all the pre-selected installation defaults.
3. Once complete, we can check that Python was installed correctly by opening a Terminal and entering the command `python3 --version`. The latest Python 3.7 version number should print to the Terminal.

Advanced

Since our system now has both Python 2 (which came pre-installed) and Python 3, we must remember to use the `python3` command (instead of just `python`) when running scripts. If you would rather not have to remember the `python3` command and just use `python` instead, then creating a command alias is your best bet.

1. Execute `open ~/.bash_profile` from a Terminal (if the file was not found, then run `touch ~/.bash_profile` first).
2. Copy and paste `alias python="python3"` into the now open `.bash_profile` file and save.
3. While we're at it, go ahead and copy and paste `alias pip="pip3"` into the file as well in order to create an alias for the Python 3 `pip` package manager.
4. Finally, restart the Terminal and run `python --version`. We should see the exact same output as running `python3 --version`.

Windows

Follow the below steps to install Python 3 on Windows.

1. Go to the Python Releases for Windows page and download the latest stable release Windows x86-64 executable installer.
2. After the download is complete, run the installer.
3. On the first page of the installer, be sure to select the "Add Python to PATH" option and click through the remaining setup steps leaving all the pre-select installation defaults.
4. Once complete, we can check that Python was installed correctly by opening a Command Prompt (CMD or PowerShell) and entering the command `python --version`. The latest Python 3.7 version number should print to the console.

Installing Miniconda

Mac

Follow the below instructions to install the latest Miniconda version for Mac.

1. Go to the Miniconda Download page and download the Python 3.7 Mac OS X 64-bit `.pkg` installer.
2. After the download is complete, run the installer and click through the setup steps leaving all the pre-selected installation defaults.

3. Once complete, we can check that Miniconda was installed correctly by opening a Terminal and entering the command `conda list`. This will print a list of packages installed by Miniconda.

Windows

Follow the below instructions to install the latest Miniconda version for Windows.

1. Go to the Miniconda Download page and download the Python 3.7 Windows 64-bit `.exe` installer.
2. After the download is complete, run the installer and click through the setup steps leaving all the pre-selected installation defaults.
3. Once complete, we can check that Miniconda was installed correctly by opening a Command Prompt (CMD or PowerShell) and entering the command `conda list`. This will print a list of packages installed by Miniconda.

Running Your First Python Script

With a Python installation complete, let's try out our new development environment by writing and executing our first script.

1. Open a Terminal (if on Mac) or Command Prompt (if on Windows) and browse to a directory of your choice.
2. Create a new file called `mycode.py` and open it with your favorite text editor.
3. Copy and paste the following code and save the file.

```
print("I'm running Python code on my own environment!")
```

4. Finally, run the script by executing the following command from the Terminal/Command Prompt in the folder where the file was saved.

```
python mycode.py
```

5. Our console should print the following message

```
I'm running Python code on my own environment!.
```

6. That's it! We just ran Python code on our own personal environment.

Installing Jupyter Notebook

Now that we have a Python distribution installed and were able to run some Python code, let’s install the Jupyter Notebook package. Jupyter Notebook is an open-source web application that allows us to create and share documents that contain live code, equations, visualizations and narrative text. This is a fantastic way to both learn and share Python code and the installation is made easier by our package installer! The installation steps differ depending on which Python distribution we installed above, so be sure to jump to the appropriate section. If you have both pip and Miniconda installed, we recommend using Miniconda to install the Jupyter Notebook package.

Using Miniconda

Follow the below instructions to install the Jupyter Notebook package using the Miniconda package manager `conda`.

1. Open a new Terminal (Mac) or Command Prompt (Windows).
2. Run `conda install jupyter` to download and install the Jupyter Notebook package.
3. Once complete, we can check that Jupyter Notebook was successfully installed by running `jupyter notebook` from a Terminal (Mac) / Command Prompt (Windows). This will startup the Jupyter Notebook server, print out some information about the notebook server in the console, and open up a new browser tab to http://localhost:8888

Using Standard Python

Follow the below instructions to install the Jupyter Notebook package using the `pip` Python package manager.

1. Open a new Terminal (Mac) or Command Prompt (Windows).
2. Run `pip install jupyter` to download and install the Jupyter Notebook package.
3. Once complete, we can check that Jupyter Notebook was successfully installed by running `jupyter notebook` from a Terminal (Mac) / Command Prompt (Windows). This will startup the Jupyter Notebook server, print out some information about the notebook server in the console, and open up a new browser tab at http://localhost:8888.

Wrapping Up

Congratulations, you now have a fully functioning Python environment running on your own computer and have taken the next step in your Python journey. Have fun and don’t forget to check out the many Python courses available right here at Codecademy!