# A

# INSTALLATION AND TROUBLESHOOTING

Python has several versions, and there are a number of ways to set it up on each operating system. Use this appendix to install Python if the approach in Chapter 1 didn't work or if you want to install a different version of Python than the one that came with your system.

## Python on Windows

The instructions in Chapter 1 show you how to install Python using the official installer at *https://python.org/*. If you couldn't get Python to run after using the installer, the troubleshooting instructions in this section should help you get Python up and running.

### *Finding the Python Interpreter*

If you've entered the simple command `python` and get an error, such as *python is not recognized as an internal or external command*, you most likely forgot to select the *Add Python to PATH* option when you ran the installer. In this case, you'll need to tell Windows where to find the Python interpreter. To find it, open your C drive and find the folder that starts with the name *Python* (you might need to enter the word `python` in the Windows Explorer search bar to find the right folder, because it might be nested further down). Open the folder, and look for a file named *python* in lowercase. Right-click this file and choose **Properties**; the path to this file will be listed under the heading *Location*.

To tell Windows where to find the interpreter, open a terminal window and enter the path followed by the `--version` command, like so:

```
$ C:\Python37\python --version
Python 3.7.2
```

Your path might look something more like `C:\Users\username\Programs\Python37\python` on your system. Using this path, Windows should then run the Python interpreter.

### *Adding Python to Your Path Variable*

It's annoying to type the full path each time you want to start a Python terminal session, so let's add the path to the system so you can just use the `python` command. Open your system's **Control Panel**, click **System and Security**, and then click **System**. Click **Advanced System Settings**. In the window that appears, click **Environment Variables**.

In the box labeled *System variables*, look for a variable called `Path`. Click the word **`Path`**, and then click **Edit**. You should see a list of locations that your system searches through when it looks for programs. Click **New**, and paste the path to your *python.exe* file in the text box that appears. If your system is set up like mine, that would be:

```
C:\Python37
```

Notice that we're not including the name of the *python.exe* file; we're just telling the system where to look for it.

Close your terminal window and open a new one. Doing so will load the new `Path` variable into your terminal session. Now when you enter `python --version`, you should see the version of Python you just added to your `Path` variable. You can now start a Python terminal session by just entering `python` at a command prompt.

NOTE

> *If you're using an earlier version of Windows, you might see a box labeled* Variable value *when you click* ***Edit****. If you see this box, use the right arrow key to scroll all the way to the right. Be careful not to overwrite the existing variable; if you do, click* ***Cancel*** *and try again. Add a semicolon and the path to your* python.exe *file to the existing variable:*
>
> ```
> %SystemRoot%\system32\...\System32\WindowsPowerShell\v1.0\;C:\Python37
> ```

### *Reinstalling Python*

If you're still unable to run Python, oftentimes uninstalling Python and running the installer again will address any problems that occurred in your first attempt.

To do this, open your system's **Control Panel** and click **Programs and Features**. Scroll down until you see the version of Python you just installed, and select it. Click **Uninstall/Change**, and then click **Uninstall** in the dialog that appears. Then run the installer again using the instructions in Chapter 1, but this time make sure you select the **Add Python to PATH** option and any other settings that are relevant to your system. If you're still running into trouble and aren't sure where to get help, see the suggestions in Appendix C.

## Python on macOS

The installation instructions in Chapter 1 use the official Python installer at *https://python.org/*, which I recommend you use unless you have a specific reason not to. Another approach uses Homebrew, a tool you can use to install a variety of software on macOS. If you're already using Homebrew and want to use it to install Python, or if the people you're working with use Homebrew and you want a similar setup to what they're using, you can use the following instructions.

### *Installing Homebrew*

Homebrew depends on some of the command line tools from Apple's Xcode package, so you'll first need to install the Xcode command line tools. Open a terminal and run this command:

```
$ xcode-select --install
```

Click through the confirmation dialogs that appear (this might take a while, depending on your connection speed). Next, install Homebrew by entering the following command:

```
$ /usr/bin/ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

You can find this command at *https://brew.sh/*. Make sure you include a space between `curl -fsSL` and the URL.

NOTE

*The `-e` in this command tells Ruby (the programming language Homebrew is written in) to execute the code that's downloaded here. You should only run commands like this from sources you trust.*

To confirm that Homebrew installed correctly, run this command:

```
$ brew doctor
Your system is ready to brew.
```

This output means you're ready to use Homebrew to install packages to your system.

### *Installing Python*

To install the latest version of Python, enter the following command:

```
$ brew install python
```

Check which version was installed by using this command:

```
$ python3 --version
Python 3.7.2
$
```

Now you can start a Python terminal session using the command `python3`. You can also use the `python3` command in your text editor so it runs programs with the version of Python you just installed instead of the system's earlier version. If you need help configuring Sublime Text to use the version you just installed, see the instructions in Chapter 1.

## Python on Linux

Python is included by default on almost every Linux system. But if the default version is earlier than Python 3.6, you should install the latest version. The following instructions should work for most apt-based systems.

We'll use a package called `deadsnakes`, which makes it easy to install multiple versions of Python. Enter the following commands:

```
$ sudo add-apt-repository ppa:deadsnakes/ppa
$ sudo apt-get update
$ sudo apt install python3.7
```

These commands should install Python 3.7 onto your system.

Enter the following command to start a terminal session that runs Python 3.7:

```
$ python3.7
>>>
```

You'll also want to use this command when you configure your text editor and when you run programs from the terminal.

## Python Keywords and Built-in Functions

Python comes with its own set of keywords and built-in functions. It's important to be aware of these when you're naming variables: your variable names cannot be the same as these keywords and shouldn't be the same as the function names, or you'll overwrite the functions.

In this section, we'll list Python's keywords and built-in function names, so you'll know which names to avoid.

### *Python Keywords*

Each of the following keywords has a specific meaning, and you'll see an error if you try to use any of them as a variable name.

| | | | | |
|---|---|---|---|---|
| False | await | else | import | pass |
| None | break | except | in | raise |
| True | class | finally | is | return |
| and | continue | for | lambda | try |
| as | def | from | nonlocal | while |
| assert | del | global | not | with |
| async | elif | if | or | yield |

### *Python Built-in Functions*

You won't get an error if you use one of the following readily available built-in functions as a variable name, but you'll override the behavior of that function:

| | | | | |
|---|---|---|---|---|
| abs() | delattr() | hash() | memoryview() | set() |
| all() | dict() | help() | min() | setattr() |
| any() | dir() | hex() | next() | slice() |
| ascii() | divmod() | id() | object() | sorted() |
| bin() | enumerate() | input() | oct() | staticmethod() |
| bool() | eval() | int() | open() | str() |
| breakpoint() | exec() | isinstance() | ord() | sum() |
| bytearray() | filter() | issubclass() | pow() | super() |
| bytes() | float() | iter() | print() | tuple() |
| callable() | format() | len() | property() | type() |

| | | | | |
|---|---|---|---|---|
| chr() | frozenset() | list() | range() | vars() |
| classmethod() | getattr() | locals() | repr() | zip() |
| compile() | globals() | map() | reversed() | __import__() |
| complex() | hasattr() | max() | round() | |