Chapter 2. Editing And Executing

Now that the main concepts of programming have been explained, it’s time to actually do some programming. In order for you to edit and execute a program, you’ll need a program to type, an editor, and Python. At first, what you will type will be provided for you – you’ll type exactly what this book tells you to type. But as we move forward, you’ll have more and more of an opportunity to write parts yourself, and ultimately write whole programs yourself.

Some of the instructions for editing and executing are “system-dependent” – that is, they depend on whether you are using a Windows PC or a Mac or something else. The presentation in this chapter is specific to Microsoft Windows on a PC and Apple OSX on a Mac. But where it’s appropriate to do so, alternate instructions for UNIX/Linux are included.

Here's a checklist of the things you will need to consider:

<table>
<thead>
<tr>
<th>Checklist</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ Choose an editor. Install, if necessary. Configure for your use.</td>
</tr>
<tr>
<td>✓ Choose a compiler. Install, if necessary. Configure for your use.</td>
</tr>
<tr>
<td>✓ Choose a folder on your system for storing your files.</td>
</tr>
<tr>
<td>✓ Configure your PC (if you are using Microsoft Windows)</td>
</tr>
</tbody>
</table>

Once this checklist is completed, you will be ready to edit and run your first program in Python!

2.1 Choosing An Editor

The first thing to do is to choose an editor. Our choice for PCs is Notepad, which can be found under Start|All Programs|Accessories|Notepad. No installation is necessary for Notepad. Our choice for Mac is TextEdit, which can be found in Finder under Applications – also no installation.

2.1.1 Configuring Notepad On A Windows PC
Chapter 2. Editing And Compiling

Two settings need to be made in Notepad, so that you can see “line numbers”. The first is to turn off “word wrap” – to do so, use the Format menu. Make sure that there is no checkmark next to the “Word Wrap” option. If there is, click it to make it go away. Then turn on the “status bar”. To do so, use the View menu. If there is a checkmark beside “Status Bar”, leave it. Otherwise, click it. The Format and View menus should look like this when you’re done:

On your own computer these settings will persist from session to session. But in the computer lab, you probably will have to configure these each time you sit at a computer workstation.

2.1.2 Configuring TextEdit On A Mac

For Mac TextEdit, the default configuration is for “rich text”. That will not do for programming. So go to the Format menu and click “Make Plain Text”. Thereafter the menu will show “Make Rich Text” instead, and the edit window will look like the one on the right.

And this one is really important! Go to TextEdit’s preferences and uncheck “Smart quotes” and “Smart dashes”:

2.1.3 Installing JNotePad

The JNotePad editor was written by the author of this book. It is similar to Notepad and TextEdit, except that it works on almost any operating system, including Windows, OSX, and versions of Linux with a graphical user interface. It also contains menu-accessible Python (and C++ and Java) “code
Chapter 2. Editing And Compiling

block” templates that match the examples in this book. Also, JNotePad is a good choice if you plan to go back and forth between a PC and a Mac, because it automatically deals with the extra blank lines and the loss of line breaks that happen with files that get traded back and forth between Macs and PCs. The JNotePad window looks like the one shown here.

The installation of JNotePad is very easy – it just involves copying its startup icon from the Internet URL www.rdb3.com/jnnotepad to your desktop or flash drive. Users of Mac computers can start using JNotePad right away for Python program editing. But in order for Windows PC users to use the icon, they first need to download and install either the “Java SE JRE” or the “Java SE JDK” from the Internet URL www.oracle.com/technetwork/java/javase/downloads, where the latest version at the time of this writing is Java SE 8u25. For Java compiling, you need the JDK, as explained below. The lighter JRE just for running programs that others (like yourself and like the author of JNotePad) wrote in Java.

2.2 Choosing Which Python To Use

There’s more than one version of Python, and code written for one does not usually work in another without modification. For purposes of this book we use Python 3. Instructions for using Python on Windows PCs, Macs, Linux, and UNIX are presented below.

Actually, your system may already have Python installed. To find out if one is installed and working on your system, go to a “command prompt” and enter the command python -V.

2.2.1 What’s A Command Prompt?
Just about all systems have a command prompt. It’s a solid-color window (usually black or white) with a message (or “prompt”) that indicates it’s ready for a user to type a command. After a command gets typed, it gets sent to the system when the user presses the ENTER (or “return”) key. Before we can talk about running a Python program, we need to get a command prompt so that we can type commands.

How you get to a command prompt depends on the system you’re using. On a Mac, you simply run the Terminal app that is part of its OSX operating system – it’s in the Applications folder. On Linux and UNIX systems, you probably start out with a command prompt when the computer boots up.

In Microsoft Windows there are lots of ways to get to a command prompt, so take your pick. One way is to use either the “run” or “search” option, and enter the three letters cmd – that should find a file named “cmd” or “cmd.exe”, which you would then choose and run.
Another way is to look for the command prompt icon on the desktop or in the menu system, and click it.

If any of this works, you'll see something similar to this – a command prompt:

You’ll be doing this so often that you may wish to “pin” its icon to the task bar.

### 2.2.2 Maybe You Already Have Python

On your Windows PC, see if you have a folder on your “C” drive named “Python34”. If so, see if there’s a file there named **python** or **pythonw** or **python.exe**. If so, you have Python, and you can skip to the next section.

On your Mac, look in your Applications folder for something named “Python 3.4”. If so, you have Python, and you can skip to the next section.

On any other system, go to a “command line” and enter the command **python -V** with an uppercase “V”. If the reply is something like “Python 3.4.2”, you’re good. Anything with 3.4.something (or higher) is good.

Otherwise, you’ll need to install Python 3.
Chapter 2. Editing And Compiling

Download and install from the Python download page, [www.python.org/downloads](http://www.python.org/downloads), where the latest “3” version at the time of this writing is 3.4.3.

### 2.2.3 Configuring Python On A Windows PC

With Python installed on a Windows PC, it is important to confirm that it is in fact working correctly. Even if you found it was already installed, it is still important to perform the same confirmation steps as if you yourself installed it.

In order to use Python, you need to type some configuration commands. First confirm that you have a folder named C:\Python34 or something similar. Note that there may be a different name for the Python34 folder on your Windows system. You may even have more than one installation. If so, pick the latest one. Once you determine the folder name for your system, modify the following instructions to use that folder name instead of Python34.

To begin a Python programming session using command-line compiling, open a command prompt. Here’s what it should look like (in Windows XP and 8):

Every time you begin a Python session, you may have to enter this command:

```bash
path=c:\Python34;%path%
```

If yours works without it, great!

There is no feedback or other output produced by the `path` command, but the `python -V` command should report “version 3”, like this:
2.2.4 Configuring Python On A Mac

Once Python is installed on a Mac, it is important to confirm that it is in fact working correctly and that you’re using Python 3.

To begin a Python programming session using command-line compiling, launch the Terminal app, and you should see a window appear on your desktop:

![Terminal window]

The `python -V` command should work, but if it says version 2, then try the command `python3 -V`. If that works, then do this:

```
Every time you begin a Python session, enter this command:
alias python='python3'
```

Then proceed as normal – you’re running Python 3! Or you can just use the command `python3` instead of `python` to run your Python programs.

2.2.5 Using Python Online

You can also use Python fully online through free websites like repl.it, tutorialspoint.com, and ideone.com. You can edit and execute code directly in your favorite browser! While you ultimately may not decide to rely on these for your programming assignments, they do offer handy access to Python when you need it.

2.3 Choosing A Folder For Storing Files

To store your Python “source files” on a PC or Mac and do your programming work, you should have a folder reserved specifically for that purpose. This is called your “working folder”. Decide upon this and get it ready before writing your first program.

You can put your working folder on a hard drive or on removable media, like a flash drive. If you are not yet comfortable with “command line” mode or file structure navigation, it’s easiest to simply use a flash drive or a cloud folder. Also, with your files on a flash drive they are readily portable between home and classroom and lab, and so on – even between PC and Mac. The rest of this section is organized for students using flash drives, as others with more experience can easily adapt the instructions for themselves.
Chapter 2. Editing And Compiling

On a PC, use Windows Explorer to create your working folder on your flash drive. On a Mac, your flash drive icon should already appear on your desktop with a default name, like “NO NAME”. You can rename it as you would rename any file or folder on your Mac desktop. It’s better to name it without spaces – something like `programming`, and thereafter it will appear on your desktop like this:

Now create your working folder on your flash drive, named for example, `python`. While it’s possible to put this folder inside some other folder on the flash drive, the rest of this section is written as though the working folder is not inside some other folder.

Next, figure out how to locate your working folder in command line mode. On a PC, open a command prompt window as explained previously. Navigate to your flash drive by typing the letter designation of the drive, then a colon, and then press ENTER. That is, enter the command `j:` if your flash drive designation is “J”. Then navigate to your working folder using a command like `cd\python`, for example.

On a Mac, start the Terminal app and enter the command `cd /volumes/programming/python`, for example. That’s `cd`, not `CD`, and it’s slash `/`, not backslash `\`. And don’t forget the space after `cd` on the Mac.

Or on either Mac or PC, type `cd .`, drag and drop your working folder onto it, and press ENTER. That’s `c d space`, not just `c d`. Then you can do the drag and drop to complete the command, and then press ENTER.

2.3.1 How To Backup Your Files
The rule is simple: “only backup the files that you do not want to lose.” After you finish editing your source files, and save them onto a hard drive or flash drive, imagine for a moment what would happen if the hard drive fails tonight or you misplace your flash drive. It’s a good idea to backup your files by putting a copy of them on the Internet. There are several free options, including emailing attachments to yourself, and file hosting services like Dropbox and Google Drive. If you are doing assignments for a computer course you are taking, the course may even offer a class website for submitting your files, and you could use that to backup your work.

Decide on your own process for backing up your work, and make it a habit to email or upload your files at the end of each work session.

2.4 Configuring A Windows PC For Programming

Actually, you can use PCs for programming without any configuration changes. But there is one change that makes things easier for identifying files, and it is highly recommended. The problem is that Windows hides filename extensions by default. That is, a program’s file `hello.py` may appear only partially – that is, without the dot and the “py” after the dot – in a file listing. To change this behavior, start `Windows Explorer` from its icon, which looks like this in Windows 8:
Then use the menu command “Organize|Folder and Search Options”. Click the “View” tab, and remove the checkmark from “Hide extensions for known file types“.

The Windows 7 folder options are also located in the Control Panel, under Appearance and Personalization. Look for Folder Options, then the View tab, and then uncheck “Hide extensions for known file types”. Click OK and then close.

On your own computer, this setting will persist from session to session. But in a computer lab, you may have to configure this each time. Note that Macs do not hide filename extensions, so no configuration is necessary.

2.5 Editing

Okay – we are ready to program! Let’s start by writing the world’s smallest Python program, and compiling and running that. It actually doesn’t do anything, but if you are able to get this far, then you can at least establish that you can use an editor and that you have Python. With these details out of the way, we can focus on learning to program! Here's our first program:

```
World's Smallest Python Program (well, almost)
print("Hello")
```

Start up your text editor. The examples in this book are shown in Windows Notepad, but you can use any other editor and system. So if you are using Mac, you can use TextEdit, configured for plain text. And if you are using Linux or UNIX, use vi or any other text editor with which you are familiar.
Chapter 2. Editing And Compiling

Note that the line is fully left justified. That is very important in Python. Python does not use “curly-braces” like many other languages do, so how lines are indented or not indented does in Python what curly-braces do in other languages.

2.5.1 How To Type Code

The tendency is to type code “linearly” – that is, from the top line to the bottom, from left to right. But that is not the best way to type code. Matching parentheses and quotes and levels of indent appear throughout code. The toughest part is to keep track of these and make sure that each open parenthesis has a matching closing parenthesis, etc. The easiest way to do that is to type opening and closing symbols together, and then separate them.

In the “World's Smallest Python Program” above, you should type print, then the opening parenthesis followed immediately by its matching closing parenthesis. Then put your edit cursor between the parentheses and type two quote marks (as a container for the word hello). Then put your edit cursor between the quote marks and type hello.

Note that as you type the program, there is an indication on the editor about the line number that is currently being edited. In Notepad this is located in the lower right with the abbreviations “Ln” for line number and “Col” for column number. (If this does not appear, then revisit section 2.1.1 above.) This is not so important for now, but knowing the line number will be important when we get into larger programs, in case Python detects and reports typing errors. Unfortunately the line number does not appear in TextEdit on a Mac – one reason why JNotePad may be the better choice for an editor.

2.5.2 Saving A File

Save the file as hello.py, into the “working folder” you created for storing your programming files (j:\python on a PC in the examples in this book). If you use Notepad on a PC, you may have to enclose the filename in quotes, or else .txt may be appended to the filename! The saved file is called the “source file”, and it contains “source code”. It should look something like this on PCs and Macs:
Chapter 2. Editing And Compiling
Chapter 2. Editing And Compiling

2.6 Executing (Or “Running”)

Now that the program’s source file has been saved to the drive, you are ready to execute it, or as it’s also called, “run”. The two words are used interchangeably in programming.

First, go to a command prompt as explained above, and navigate to the drive and working folder containing your edited source file. It should look like this on a PC or Mac, with what you would type appearing like this: `cd /Volumes/programming/python` in the Mac screenshots:

2.6.1 How To Execute

Here’s how to EXECUTE it. Invoke Python by typing this command:

`python hello.py`

If there are any errors, it should be evident from the output. Line numbers should also appear in Python’s output, guiding you to the problem, like this (with a missing quote mark):
2.6.2 Rerunning
When you use the command line to run your programs, it will seem as if there is a lot of typing to do – repetitive typing. But you do not have to retype a command over and over again. PCs and Macs both let you use the UP and DOWN ARROWs of the keyboard to recall a recently typed command. On PCs, you can also use the F7 key to get a menu of recently typed commands, although the UP ARROW is the easiest way to recall the last-typed command.

2.7 Exercises, Sample Code, Videos, And Addendums

Go to www.rdb3.com/python/2 for extended materials pertaining to this chapter.
Chapter 2. Editing And Compiling