Python Programming

贾子熙 副教授

机器人科学与工程学院

Chapter 1 Getting Started

1.1 Setting Up Your Programming Environment

1.2 Python on Different Operating Systems

1.3 Troubleshooting Installation Issues

1.4 Running Python Programs from a Terminal

1.5 Summary

python简介

Python是一个高层次的结合了**解释性、编译性、互动性和面向对象**的脚本语言。

Python的设计具有很强的可读性，相比其他语言经常使用英文关键字和标点符号，它采用更具特色的语法结构。

- Python是一种解释型语言： 这意味着开发过程中没有了编译这个环节。类似于PHP和Perl语言。
- Python是交互式语言： 这意味着，您可以在一个Python提示符，直接互动执行写你的程序。
- Python是面向对象语言: 这意味着Python支持面向对象的风格或代码封装在对象的编程技术。
- Python是初学者的语言：Python对初级程序员而言，是一种伟大的语言，它支持广泛的应用程序开发，从简单的文字处理到 WWW 浏览器再到游戏。

python特点

1.易于学习：Python有相对较少的关键字，结构简单，和一个明确定义的语法，学习起来更加简单。

2.易于阅读：Python代码定义的更清晰。

3.易于维护：Python的成功在于它的源代码是相当容易维护的。

4.一个广泛的标准库：Python的最大的优势之一是丰富的库，跨平台的，在UNIX，Windows和Macintosh兼容很好。

5.互动模式：互动模式的支持，您可以从终端输入执行代码并获得结果的语言，互动的测试和调试代码片断。

6.可移植：基于其开放源代码的特性，Python已经被移植（也就是使其工作）到许多平台。

7.可扩展：如果你需要一段运行很快的关键代码，或者是想要编写一些不愿开放的算法，你可以使用C或C++完成那部分程序，然后从你的Python程序中调用。

8.数据库：Python提供所有主要的商业数据库的接口。

9.GUI编程：Python支持GUI可以创建和移植到许多系统调用。

10.可嵌入: 你可以将Python嵌入到C/C++程序，让你的程序的用户获得"脚本化"的能力。

为什么人工智能用python？

Python开发效率非常高，Python有非常强大的第三方库，基本上你想通过计算机实现任何功能，Python官方库里都有相应的模块进行支持，直接下载调用后，在基础库的基础上再进行开发，大大降低开发周期，避免重复造轮子，还有python的是可移植性、可扩展性、可嵌入性、少量代码可以做很多事，这就是为何人工智能首选Python。

两大深度学习平台

- TensorFlow
- Pytorch

1.1 Setting Up Your Programming Environment

1.1.1 Python 2 and Python 3

1.1.2 Running Snippets of Python Code

1.1.3 Hello World!

1.1.1 Python 2 and Python 3

The main differences are in the following areas:

- print function
- Integer division
- Unicode
- exceptional handling
- xrange
- map function
- Does not support has_key

If you need to install Python, use Python 3. If Python 2 is the only version on your system and you'd rather jump into writing code instead of installing Python, you can start with Python 2. But the sooner you upgrade to using Python 3 the better.

P.S. *Python2.7 will be discontinued by 2020!*

1.1.2 Running Snippets of Python Code

Python comes with an interpreter that runs in a terminal window, allowing you to try bits of Python without having to save and run an entire program.

1.1.3 Hello World

In Python, you can write the *Hello World* program in one line:

```
print("Hello world!")
```

1.2 Python on Different Operating Systems

Python runs on all the major operating systems.
However, the methods for setting up Python on different operating systems vary slightly.

1.2.1 Python on Linux

Linux systems are designed for programming, so Python is already installed on most Linux computers. For this reason, there’s very little you have to install and very few settings you have to change to start programming.

1. Checking Your Version of Python

Open a terminal window by running the Terminal application on your system, enter **python** with a lowercase p.

```
$ python
Python 2.7.6 (default, Mar 22 2014, 22:59:38)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for
more information. >>>
```

This output tells you that Python 2.7.6 is currently the default version of Python installed on this computer.

If the output displayed Python 2.7 as the default version, try the command python3 :

```
$ python3
Python 3.5.0 (default, Sep 17 2015, 13:05:18)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for
more information. >>>
```

This output means you also have Python 3 installed.

2. Installing a Text Editor

Geany is a simple text editor:

- it’s easy to install;
- will let you run almost all your programs directly from the editor instead of through a terminal;
- uses syntax highlighting to color your code;
- runs your code in a terminal window so you’ll get used to using terminals.

You can install Geany in one line on most Linux systems:

```
$ sudo apt-get install geany
```

3. Running the Hello World Program

To start your first program, open Geany.
After you've opened Geany and saved your file, enter the following line:

```
print("Hello Python world!")
```

Geany assumes the correct command for each is python , but if your system uses the python3 command, you'll need to change this.

```
python3 -m py_compile "%f"
```

You need to type this command exactly as it's shown. Make sure the spaces and capitalization match what is shown here.

Your Execute command should look like this:

```
python3 "%f"
```

Again, make sure the spacing and capitalization match what is shown here.

Now run *hello_world.py* by selecting Build ▶ Execute in the menu, by clicking the Execute icon(which shows a set of gears),or by pressing F5. A terminal window should pop up with the following output:

```
Hello Python world!
------------------
(program exited with code: 0)
Press return to continue
```

4. Running Python in a Terminal Session

You can try running snippets of Python code by opening a terminal and typing `python` or `python3` . And enter the following line in the terminal session:

```
>>> print("Hello Python interpreter!")
    Hello Python interpreter!
>>>
```

You should see your message printed directly in the current terminal window.

1.2.2 Python on OS X

Python is already installed on most OS X systems. Once you know Python is installed, you'll need to install a text editor and make sure it's configured correctly.

1.Checking Whether Python Is Installed

Open a terminal window, enter python with a lowercase p. You should see output like this:

```
$ python
Python 2.7.5 (default, Mar 9 2014, 22:15:05)
[GCC 4.2.1 Compatible Apple LLVM 5.0 (clang-500.0.68)] on
darwin
Type "help", "copyright", "credits", or "license" for
more information.
>>>
```

2.Running Python in a Terminal Session

You can try running snippets of Python code by opening a terminal and typing python or python3, and enter the following line in the terminal session:

```
>>> print("Hello Python interpreter!")
    Hello Python interpreter!
>>>
```

3. Installing a Text Editor

Sublime Text is a simple text editor:

- it’s easy to install on OS X;
- will let you run almost all of your programs directly from the editor instead of through a terminal;
- uses syntax highlighting to color your code;
- runs your code in a terminal session embedded in the Sublime Text window to make it easy to see the output.

4. Configuring Sublime Text for Python 3

If you use a command other than python to start a Python terminal session, you'll need to configure Sublime Text. Issue the following command to find out the full path to your Python interpreter:

```
$ type -a python3
python3 is /usr/local/bin/python3
```

Now open Sublime Text, and go to Tools ▶ Build System ▶ New Build System，which will open a new configuration file for you. Delete what you see and enter the following:

```
{
  "cmd": ["/usr/local/bin/python3", "-u", "$file"],
}
```

This code tells Sublime Text to use your system’s python3 command when running the currently open file.

5. Running the Hello World Program

To start your first program, launch Sublime Text.

Make a folder called *python_work* somewhere on your system for your projects.

After you've saved your file, enter the following line:

```
print("Hello Python world!")
```

This sets Python 3 as the default version of Python, and you'll be able to select Tools ▶ Build or just press Command+B to run your programs from now on.
A terminal screen should appear at the bottom of the Sublime Text window, showing the following output:

```
Hello Python world!
[Finished in 0.1s]
```

1.2.3 Python on Windows

Windows doesn’t always come with Python, so you’ll probably need to download and install it, and then download and install a text editor.

1. Installing Python

First, check whether Python is installed on your system.
In the terminal window, enter **python** in lowercase：

- If you get a Python prompt (>>>), Python is installed on your system.
- However, you'll probably see an error message telling you that python is not a recognized command.

2. Starting a Python Terminal Session

Open a command window and enter **python** in lowercase.
If you get a Python prompt (>>>), Windows has found the version of Python you just installed:

```
C:\> python
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 22:15:05)
[MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

If this worked, you can move on to the next section, “Running Python in a Terminal Session.”

However, you may see output that looks more like this:

```
C:\> python
'python' is not recognized as an internal or external
command, operable program or batch file.
```

When you think you know the path, test it by entering that path into a terminal window. Open a command window and enter the full path you just found:

```
:\> C:\Python35\python
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 22:15:05) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

If this worked, you know how to access Python on your system.

3. Running Python in a Terminal Session

Enter the following line in your Python session, and make sure you see the output *Hello Python world!*

```
>>> print("Hello Python world!")
Hello Python world!
>>>
```

Any time you want to run a snippet of Python code, open a command window and start a Python terminal session. To close the terminal session, press Ctrl + Z and then press ENTER, or enter the command exit().

4. Installing a Text Editor

Geany is a simple text editor:

- it’s easy to install;
- will let you run almost all of your programs directly from the editor instead of through a terminal;
- uses syntax highlighting to color your code;
- runs your code in a terminal window so you’ll get used to using terminals. Appendix B provides information on other text editors, but I recommend using Geany unless you have a good reason to use a different editor.

To start your first program, open Geany.

After you've saved your file, type the following line:

```
print("Hello Python world!")
```

If you needed to enter a path like C:\Python35\python， to start a Python interpreter, follow the directions in the next section to configure Geany for your system.

5. Configuring Geany

In the Compile and Execute commands, add the drive your python command is on and the folder where the python command is stored. Your Compile command should look something like this:

```
C:\Python35\python -m py_compile "%f"
```

Your path might be a little different, but make sure the spaces and capitalization match what is shown here.
Your Execute command should look something like this:

```
C:\Python35\python "%f"
```

6. Running the Hello World Program

You should now be able to run your program successfully. Run *hello_world.py* by selecting Build ▶ Execute in the menu, by clicking the Execute icon(which shows a set of gears), or by pressing F5. A terminal window should pop up with the following output:

```
Hello Python world!
------------------
(program exited with code: 0) Press return to continue
```

1.3 Troubleshooting Installation Issues

Hopefully, setting up your programming environment was successful, but if you've been unable to run *hello_world.py*, here are a few remedies you can try:

- When a program contains a significant error, Python displays a *traceback*. Python looks through the file and tries to report the problem. The traceback might give you a clue as to what issue is preventing the program from running.
- Step away from your computer, take a short break, and then try again.Remember that syntax is very important in programming, so even a missing colon, a mismatched quotation mark, or mismatched parentheses can prevent a program from running properly. Reread the relevant parts of this chapter, look over what you've done, and see if you can find the mistake.

- Start over again. You probably don't need to uninstall anything, but it might make sense to delete your *hello_world.py* file and create it again from scratch.
- Ask someone else to follow the steps in this chapter, on your computer or a different one, and watch what they do carefully. You might have missed one small step that someone else happens to catch.
- Find someone who knows Python and ask them to help you get set up. If you ask around, you might find that you know someone who uses Python.
- The setup instructions in this chapter are also available online, through https://www.nostarch.com/pythoncrash-course/. The online version of these instructions may work better for you.
- Ask for help online. Appendix C provides a number of resources and areas online, like forums and live chat sites, where you can ask for solutions from people who've already worked through the issue you're currently facing.

1.4 Running Python Programs from a Terminal

1.4.1 On Linux and OS X

Open a new terminal window and issue the following commands to run *hello_world.py*:

```
❶ ~$ cd Desktop/python_work/
❷ ~/Desktop/python_work$ ls
hello_world.py
❸ ~/Desktop/python_work$ python hello_world.py
Hello Python world!
------------------
(program exited with code: 0) Press return to continue

  Hello Python world!
```

1.4.2 On Windows

Open a new terminal window and issue the following commands to run *hello_world.py*:

```
C:\> cd Desktop\python_work
C:\Desktop\python_work> dir
hello_world.py
C:\Desktop\python_work> python hello_world.py
Hello Python world!
```

If you haven’t configured your system to use the simple command python , you may need to use the longer version of this command:

```
C:\$ cd Desktop\python_work
C:\Desktop\python_work$ dir
hello_world.py
C:\Desktop\python_work$ C:\Python35\python hello_world.py Hello Python world!
```

Most of your programs will run fine directly from your editor, but as your work becomes more complex, you might write programs that you’ll need to run from a terminal.

1.5 Summary

In this chapter you learned a bit about Python in general, and you installed Python to your system if it wasn't already there. You also installed a text editor to make it easier to write Python code. You learned to run snippets of Python code in a terminal session, and you ran your first actual program, *hello_world.py*. You probably learned a bit about troubleshooting as well.

In the next chapter you'll learn about the different kinds of data you can work with in your Python programs, and you'll learn to use variables as well.