TUTORIAL 1: Setting up a Raspberry Pi

Last revision: January 26th, 2022

1 INTRODUCTION & MOTIVATION

Pervasive systems must often be able to locally monitor sensors (for example, PIR sensors) and control devices (for example, a light strip or a power plug). Thus, a local computer is needed to perform these operations. One versatile platform is a Raspberry Pi 4.

In this tutorial you will learn how to set up a Raspberry Pi and run a simple Python application. Initially, you will develop your code directly in the Raspberry, i.e. you will use the version of Raspberry Pi OS with a GUI desktop, where the development tools are available to write your code and run it.

In the end of this tutorial, you should have a running Raspberry Pi, using two different methods, and run Python applications using an IDE and/or the command line.

2 WHAT TO READ BEFORE THE TUTORIAL

It is expected that you have a basic knowledge of the following topics:

- The Raspberry Pi hardware architecture, specifically which ports are available and how new hardware can be connected;
- The Linux command line (or shell), specifically manage files, directories and navigate through them;
- What the Python interpreter is and how to use it to run applications with it.

3 MATERIALS

You will need the following materials:

- Computer (Windows, Linux, macOS)
- Raspberry Pi 4
- SD card with Raspberry Pi OS
- HDMI to mini-HDMI cable
- Power supply
- Keyboard, mouse and monitor (any wired keyboard and mouse should work)
- Raspberry Pi Imager (you can download it https://www.raspberrypi.org/software/)
- Raspberry Pi OS 32-bits (optional, as you can download it with Raspberry Pi Imager)

4 PREPARING THE RASPBERRY

You will start by preparing the SD card with Raspberry Pi OS (RPiOS). The SD card that was provided to you should already be formatted with RPiOS (desktop version). If you use the

default installation, then you should be ready to go and you can skip to section 4.2. If your SD card gets corrupted, you can use the instructions provided in section 4.1 to prepare a new SD card.

4.1 Write Raspberry OS to the SD card

The first step is to write RPiOS to a SD card. This process is performed in your computer. For that, first download the Raspberry Pi Imager – it is available for Windows, macOS and Ubuntu – and insert the SD card in your computer. Once installed, you will be greeted by a screen as like the one depicted in Figure 1.

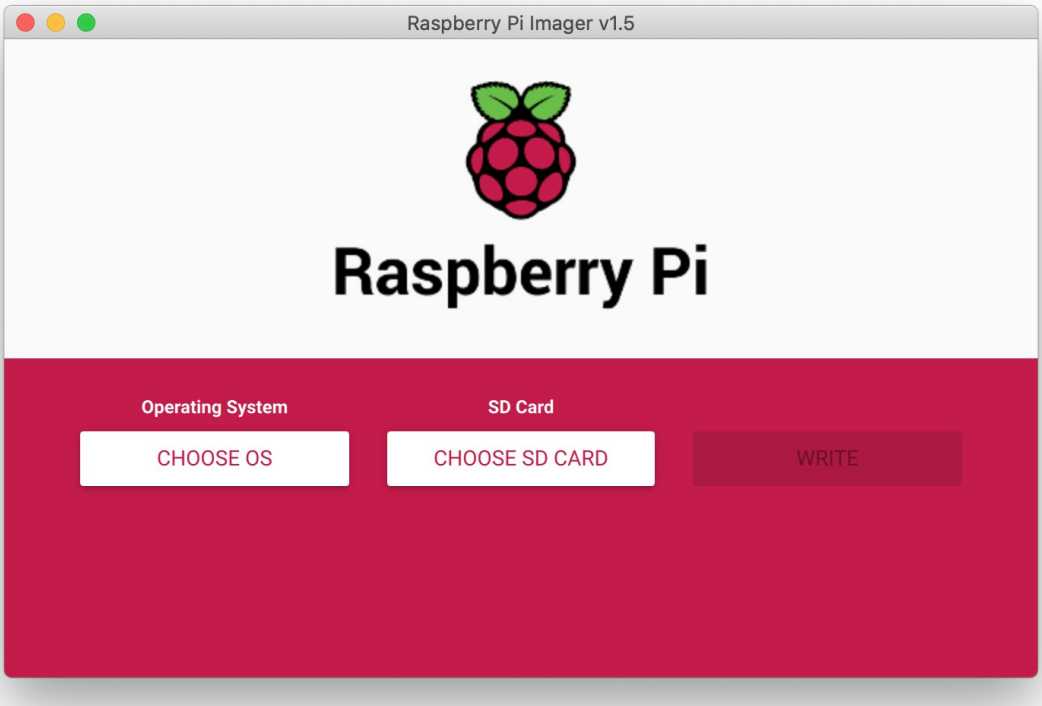

Figure 1. Raspberry Pi Imager's main screen.

Press the first button, "CHOOSE OS", and then select the OS "Raspberry Pi OS (32-bit)", as depicted in Figure 2. Then, press the button "CHOOSE SD CARD" and select the SD card where the OS will be written. Finally, press "WRITE". A warning will pop up saying that all data in the SD card will be erased. The writing process will take some minutes and a message will pop up once it is finished. **Do not remove the SD card during the writing process or it might become unusable**. Once the SD is written, you can eject it and insert it in the Raspberry Pi.

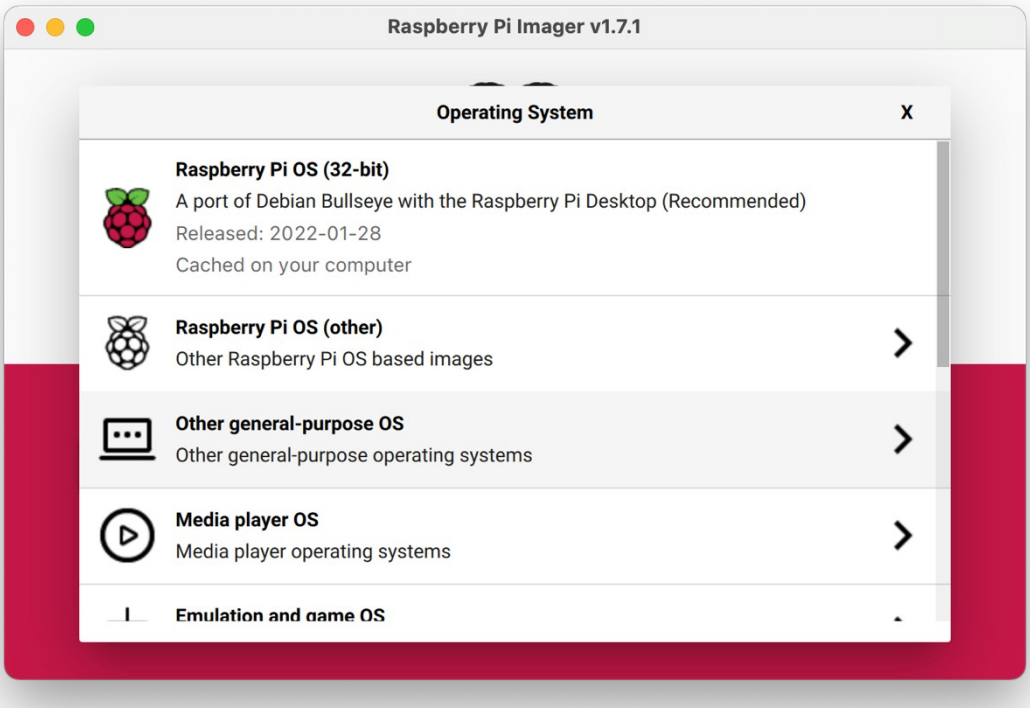

Figure 2. Raspberry Pi Imager's operating system selection screen.

4.2 Connecting the Raspberry Pi and booting

Before powering up the Raspberry, connect the keyboard and the mouse to two of its USB ports, the HDMI cable to your monitor and one of the mini-HDMI ports in the Raspberry and insert the SD card. Then, plug the power supply and the Raspberry should start booting.

When the Raspberry boots, by default it looks for an installed SD card with an operating system. If it is not found, nothing will be displayed in your screen. If an OS is found, then a booting screen will be presented (it might take some seconds to do so).

Warning: once the Raspberry is powered on and running, you must avoid remove the SD card, or power off the Raspberry without shutting it down properly, as the SD card might get corrupted.

5 PROGRAMMING IN THE RASPBERRY PI OS

RPiOS already comes with pre-installed software for developing Python applications, specifically IDEs and the Python interpreter. In this course you should use Python 3 – the support for Python 2 ended in January 1st, 2020, and it should not be used, although it is available in RPiOS.

As for the IDE, you are free to use the one you prefer, although, for the purpose of this tutorial, Thonny will be used for presenting the Python programming workflow.

Thonny, depicted in Figure 3, is a simple IDE for Python development, with features such as code completion, an integrated basic debugger and Python package management via PIP (Package Installer for Python).

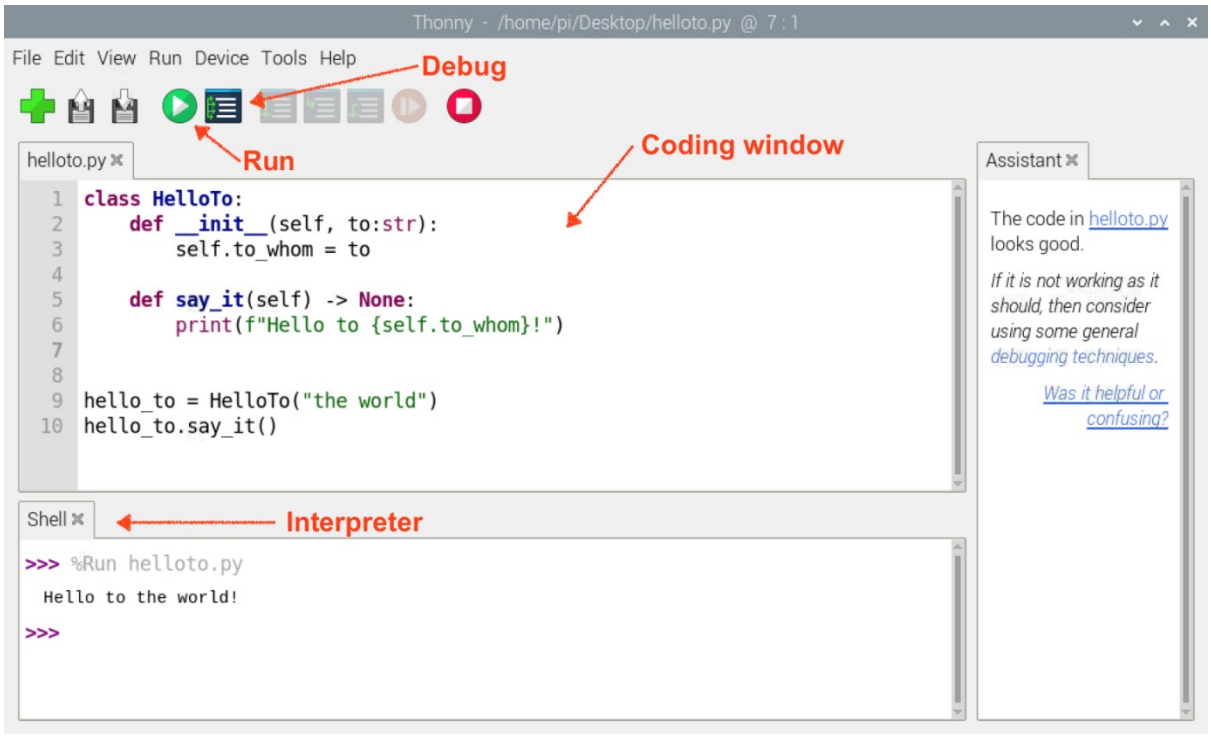

Figure 3. Thonny IDE.

In Figure 3 you can see a Python program, based in a simple class, that says "`Hello to`" a string passed as argument to the class initializer. To run the application, you can use the "Run" button in the tool bar, or the Debug button. Both will start the interpreter and display, as it can be seen in the example above, the output "`Hello to the world!`". In the case of debug, you can step through the code to analyze it – Thonny does not support breakpoints.

It is also possible to run your applications in the terminal, using the `python3` command. This is depicted in Figure 4.

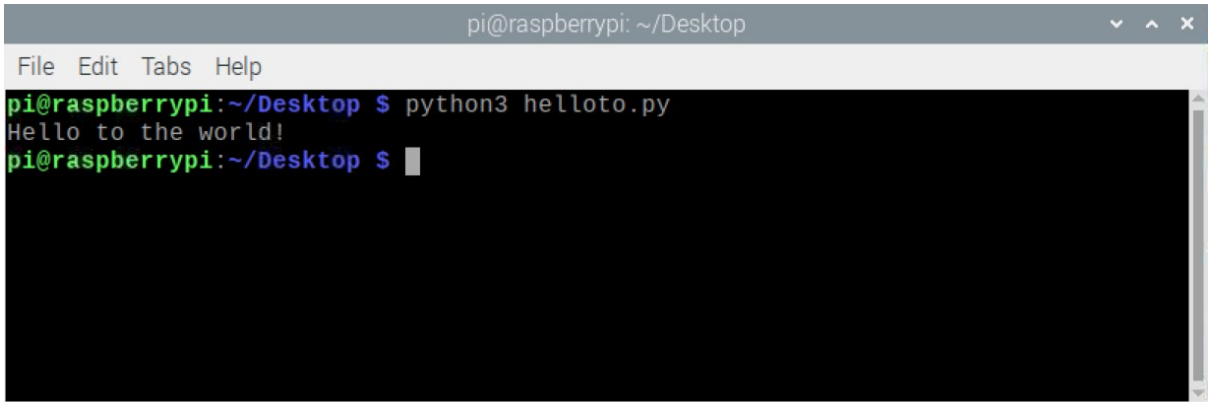

Figure 4. Running an application via terminal.

Warning: when you execute the command `python` in the terminal, the interpreter that is executed is version 2! To use version 3, use the command `python3`. To check which version of Python is installed, you can use the command `python3 -V`.

6 NEXT STEPS

6.1 Headless mode

You now have a Raspberry Pi configured as a desktop. But, what if you do not want the desktop, i.e. run the Raspberry as a server, where the desktop is not needed? This mode is commonly called "headless", i.e. the Raspberry is running without a monitor, keyboard and mouse attached. You can run the Raspberry in this mode with any of the RPiOS versions, but, usually, the Lite version is preferred, as it does not have unnecessary software such as the GUI.

Still, it might be necessary access the gateway to configure it or maybe install software. To do it you can use SSH (Secure Shell Mode). The Raspberry will have to be connected to a network and then, in your computer, you can use the `ssh` command to do so. In Windows, you will have to install Putty [1]; in macOS or Linux the command should be available by default. The SSH server in the Pi is not enabled by default, so you will have to do it manually. For more information on how to enable it in RPiOS refer to [2].

Another aspect that you should take in account is how is the Raspberry connected to the network: is it via Ethernet cable or WiFi? If it is connected via cable, no extra configurations are needed. As for WiFi, you have to set the network and password so RPiOS knowns to which network it has to connect to. You can read on how to do it in [3].

6.2 IDEs

In this tutorial Thonny was used for writing a Python application. As written above, you can use other IDEs that suit you best. Some examples are Visual Studio Code, nano or vim (the last two are available in RPiOS using their respective commands, via terminal). You can consult [4] on how to install Visual Studio Code and [5] in how to get started.

The main advantage for your team to adopt a more advanced IDE is to, ultimately, make the project development easier, since these aggregate tools (interface with Python interpreter and debugger) and provide features (code completion, refactoring, code style, …) that enable general productivity.

7 FURTHER READING

In this section, a list of extra readings are presented, so you can further explore the topics of this tutorial.

- Extended guide for setting up a Raspberry Pi, including some troubleshooting: https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

- Raspberry Pi's general documentation page: https://www.raspberrypi.org/documentation/

8 REFERENCES

[1] Putty: https://www.putty.org

[2] SSH (Secure Shell) in the Raspberry Pi: https://www.raspberrypi.org/documentation/remote-access/ssh/README.md

[3] Setting up a Raspberry Pi headless: https://www.raspberrypi.org/documentation/configuration/wireless/headless.md

[4] Installing Visual Studio Code on the Raspberry Pi: https://pimylifeup.com/raspberry-pi-visual-studio-code/

[5] Getting Started with Python in VS Code: https://code.visualstudio.com/docs/python/python-tutorial