Code Next Raspberry Pi Guide

Python on the Pi Club 2020-21

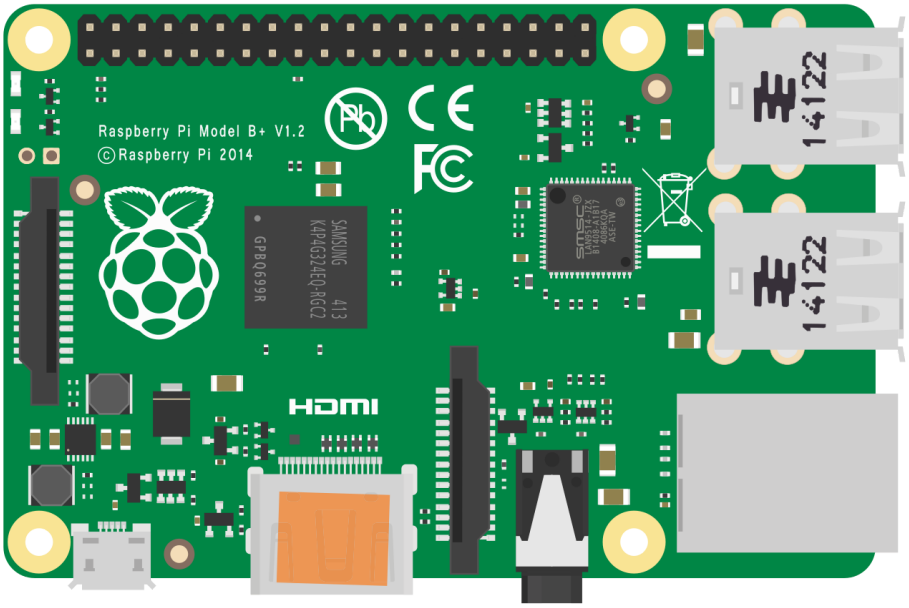


Table of contents

Overview

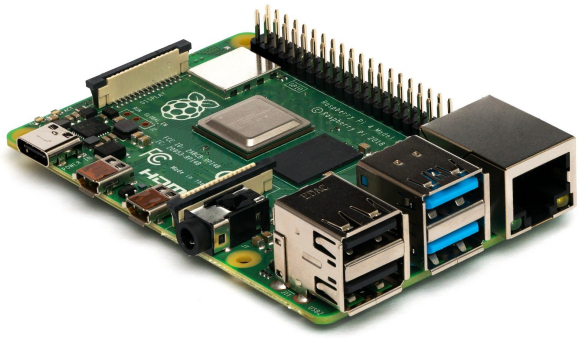
The Raspberry Pi is a revolutionary computer that has put pretty amazing computing power into a small package, revolutionizing how people and computers interact. Since the first model was released in 2012, millions of computing enthusiasts, entrepreneurs, hackers, students and makers have used this tiny and relatively inexpensive machine to build amazing projects that incorporate hardware and software, the web, AI, robotics and so much more.

The Raspberry Pi 4 comes loaded with features, such as Wifi, bluetooth, USB ports, an Ethernet connection, two mini-HDMI outputs, a camera port and more than two dozen GPIO pins that allow us to connect all sorts of electronic components and circuits for physical computing projects.

You may be surprised to find that all of the operating system and files are stored on a micro SD card. You can in fact swap these cards out when you want your Pi to load other Operating systems of configurations, such as a Retro gaming system, a media player, etc.

You can also connect various peripherals via USB, such as the mouse, keyboard, USB cameras, joysticks, etc.

Raspberry Pi 4



Raspberry Pi 400
Does not need a keyboard because it comes already encased in one.

Initial Setup

Connect your Pi following either of these videos.

Raspberry Pi 400: Quickstart guide, | setup video.

Raspberry Pi 4 with HDMI monitor: Setup video

If this is the first time you are booting up this Pi, you will need to set up a few things, such as the language and location settings and will be prompted to go online, so be prepared to enter your network credentials.

Learn more about this final setup here.

Getting to know Linux

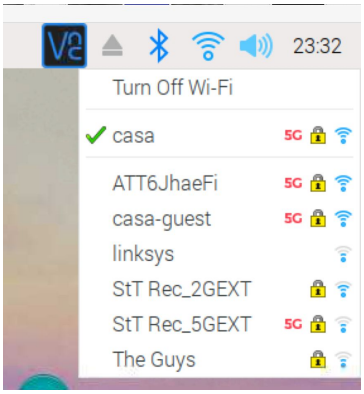
Your Raspberry Pi is running an open source operating system called Linux. Linux powers vehicles, supercomputers, enterprise servers, and it is the base operating system for Android OS and Chromebooks. This is mainly because it is so reliable and stable, so you are in for a good experience.

There are many different flavors of Linux, or distributions. The Raspberry Pi is most likely running a version of the Debian distribution called Raspberry Pi Operating System (formerly Raspbian).

Like all operating systems, it allows you to interact with your computer both graphically and through a command line interface (CLI), also known as the terminal. You will spend a lot of time typing commands into the terminal when working on the Pi.

Being open source, Linux software usually is free. The Pi OS comes pre-loaded with some software for graphics, browsing the internet, media playback, programming, and general productivity. Feel free to explore these, but for the most part, you will either be coding in a code editor, browsing the web, or working in the terminal.

Going Online
If you have not already done so, make sure you connect to a local wireless network.



The top menu bar has a Wifi icon. Click on that to select a network and enter a password.

You can also click on the Raspberry Pi Menu to open up any browsers you have installed, such as Chromium, a light version of Chrome, which works great on the Pi.

Test out the browser and do an internet speed test.

Customize your Pi
Follow this guide to learn about the Raspbian OS, including how to change appearance, add more programs, customize things, etc:
Using your Raspberry Pi - Introduction

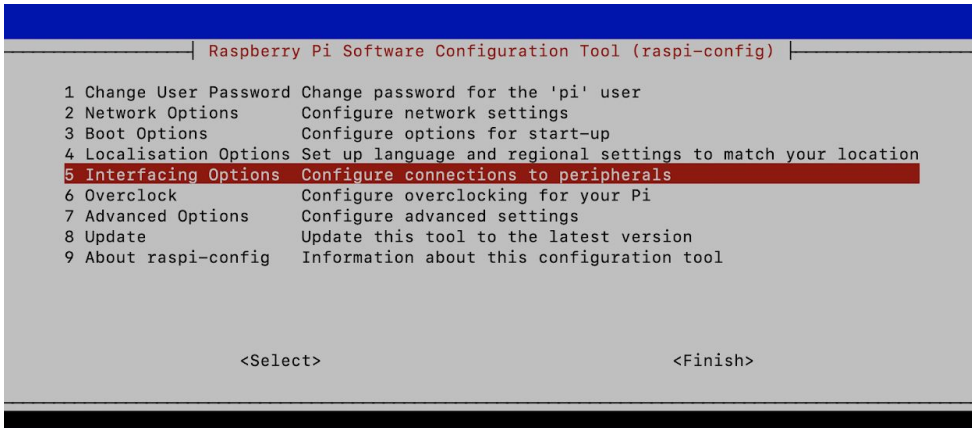
Connecting via SSH:

SSH (Secure Shell) is a secure way for two computers to talk to each other over a network connection. In order to control your Raspberry Pi without a monitor, keyboard or mouse, you can “tunnel in” using the SSH protocol from another computer on the same network. All you need is the IP address of the Raspberry Pi and the login credentials. You also need to configure your Raspberry Pi to allow for SSH connections.

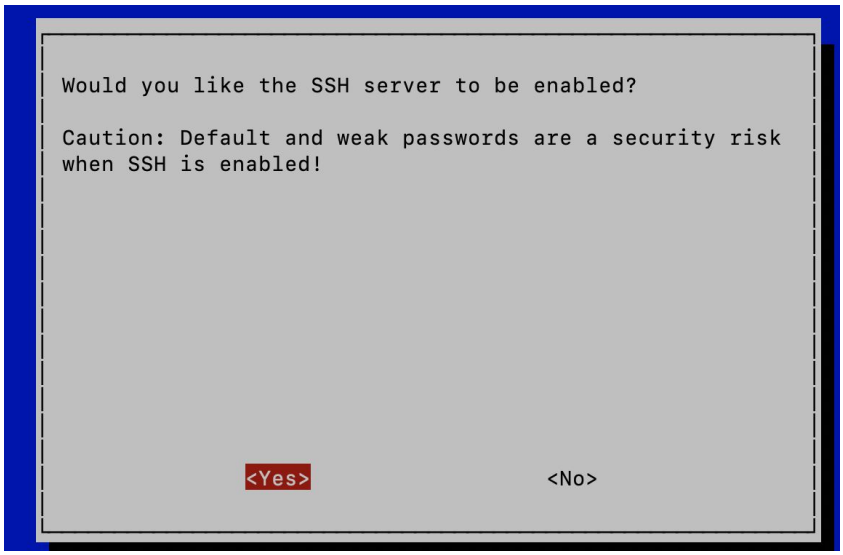
To configure your pi, (or to make sure this is already set) use the following command in the Terminal:

```
$ sudo raspi-config
```

This simple interface will open. To find the SSH setting, you can navigate with the arrow keys to Interfacing Options. On some Pi models, this option will be in the main menu.



Change the SSH settings:



As you can see, it warns you to change your login credentials. Which can also be done from this config menu. Change them if you want, but remember, all Raspberry Pis come with the following default settings:

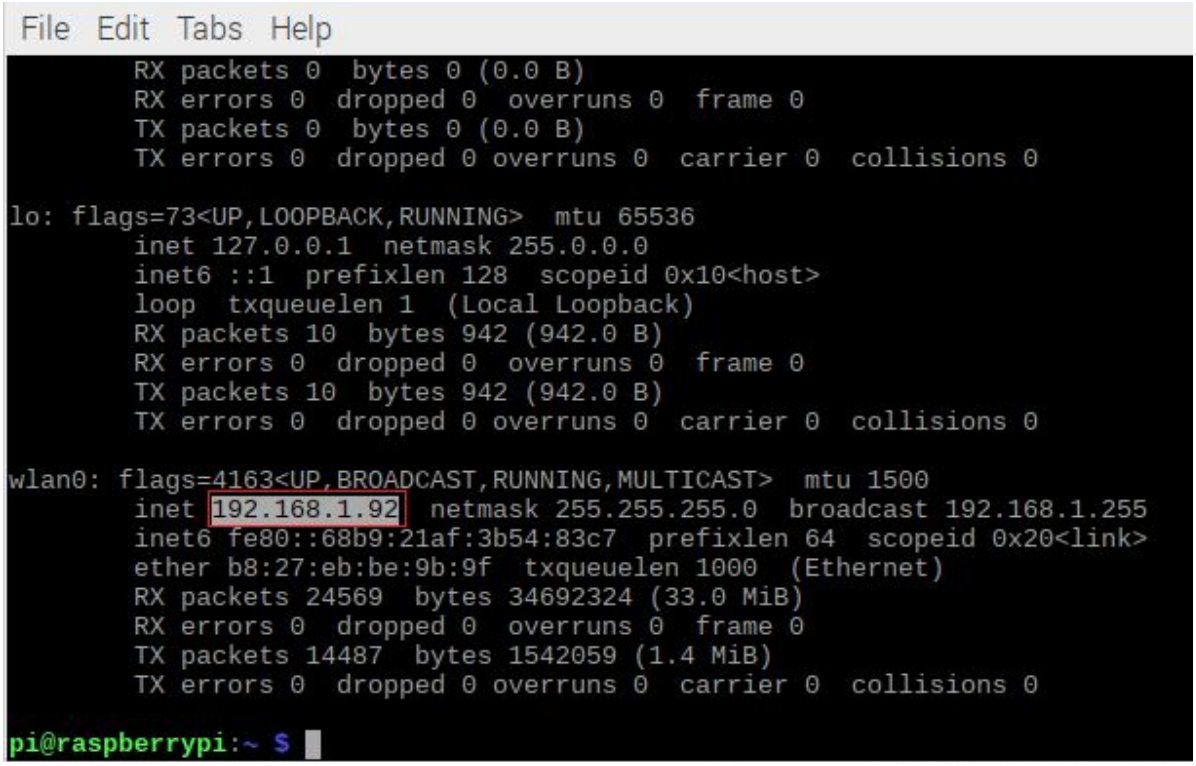
- Username: **pi**
- password : **raspberry**

Now that the Raspi SSH server is enabled, you have to find out its IP address.

On the Raspi, go to a terminal and enter:

```
$ ifconfig
```

Output will display an inet address: such as below. You may see a similar address under the lo. Use the wlan0 or eth0, depending on your connection type (wifi or ethernet)



On another computer on the same network, open a terminal window and enter a similar command as follows, replacing the IP address with your Raspi IP address. Note the **pi@** at the beginning of the statement, which is the username. You will be prompted for the password in order to connect. If you change the username from pi to another word, you will enter that instead.

```
$ ssh pi@192.168.1.92
```

You should now be able to “tunnel” into your pi and control it in a “headless” environment. Sometimes the connection is broken and you will have to run the ssh command again.

VNC Viewer

Sometimes we need to work with the Raspberry Pi GUI, or Desktop instead of just the CLI (Command Line Interface). To work on your Raspi’s graphical interface remotely you will have to load the VNC Viewer on the Pi, as well as download and install VNC Viewer on the computer you want to use to control your Pi.

Check if your pi already has a VNC viewer installed. You can find it under the Pi/Internet menu. Before allowing other computers to connect via VNC, you will need to enable this just like the SSH service above.

Run this command again:

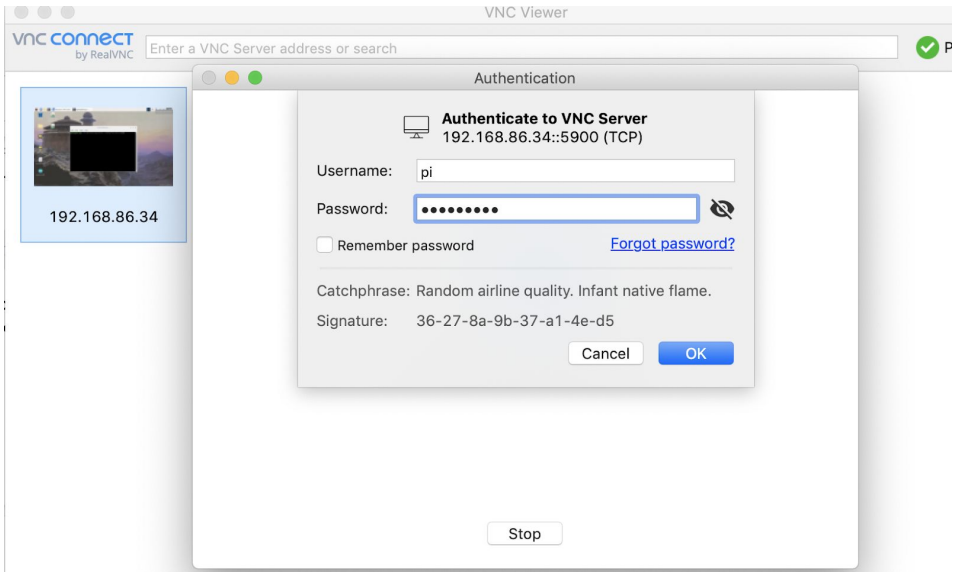
```
$ sudo raspi-config
```

Find VNC in the Interfacing Options menu and change the setting to Enabled or YES.

[This official guide will help you as well.]

Create an account
To use VNC, create an account on the web from the Real VNC site. Use this log in to authenticate on both client and server (your laptop and the Raspberry Pi)

To connect to you Pi, enter its IP address into the search bar at the top of the window. You will be prompted to enter the credentials of the Pi (see below). Make sure you are logged into your new RealVNC account.



Note: You may also need to change some settings on your client device before connecting. Mac OS should prompt you if you do.

For a more complete guide on remote connections to the Pi visit this official link: [Remote Access]

Terminal Commands

Use these commands in the terminal to do cool things! Linux uses the same BASH language as a mac, so most Mac terminal commands work in the Raspberry Pi Terminal.

Basic bash commands:

`$ cd`	change directory (`$ cd team_edge` or `cd ..` to navigate out of the directory)
`$ ls`	list files in current directory (`$ ls -a` lists all files, including hidden files)
`$ mkdir`	to create a new directory inside the current one
`$ touch`	creates a new file (ex: `$ touch my_program.py`)
`$ pwd`	prints current directory, helpful to find where you are at
`$ cp`	Copy files from one location to another
`$ mv`	This moves a file from one location, but does not copy the file. You can also use this to rename a file by not specifying a destination directory.
`$ rm`	To remove a file (Be careful! For directories adding the -r flag will recursively delete all the subfolders as well)

Other common bash commands:

`$ curl`	Use it to fetch URL data, such as files, images, full websites. Output can be saved.
`$ wget`	Similar to curl, but wget automatically downloads the data to a specific directory.
$ grep	Used to search files and contents of files. Used with piping | to filter out data from other files.
$ echo	Print out statements to the terminal.
$ cat	Use it to print out contents of a file. Used with | to pass contents to other commands.

Raspberry Pi-Specific Commands

The following commands run on the Raspberry Pi.

`$ free -h`	Print the available system memory of your Pi in human-readable text
`$ cat /proc/cpuinfo`	Get CPU info
`$ vcgencmd measure_temp`	Geta quick print-out of the chip temperature.
`$ lsusb`	Get info on connected USB devices (add --verbose flag for tons of details)
`$ sudo reboot`	Reboots your Pi
`$ raspi-config`	This command is one of the most used. Change system settings from networking, to passwords and access to services.
`$ sudo apt-get update`	Run this command before installing new software. It automatically updates your packages.
`$ ifconfig`	Prints network information, including IP address
`$ hostname -I`	Prints just the IP address for the pi
`$ cat /proc/version`	Print the Raspberry Pi version you have
`$ pinout`	Prints out an image of all the GPIO pins on the Pi, including active connections.
`$ sudo shutdown -h now`	Shuts down the Pi (do this when connecting components to GPIO pins, for example)

Note: commands beginning with sudo are for “superusers” or admins, which means you will need to enter the system password (raspberry is the default, unless you changed it) Sometimes you will see a message that says a command must be run as ‘root’, which means you should add **sudo** at the beginning.

For more bash commands see this article or this one from Raspberry Pi

Sense Hat

The Sense Hat is a popular add on or extension module that attaches to the Pi GPIO pins. These kinds of devices are called HATs or Hardware Attached on Top. The Sense Hat comes with built-in sensors for temperature, humidity, air pressure, a magnetometer, accelerometer, gyroscope, and a 8x8 full color RGB LED matrix.

To get started, attach the Sense Hat on top of the Raspberry Pi by plugging it in carefully on the GPIO pins.

Below is code to help you get started, but you can also go to the Raspberry Pi page and learn more.

On the command line, first import the Sense Hat package to your Pi:

```
$ sudo update
```

Then add the package.

```
$ sudo apt install sense-hat
```

Reboot to make sure the installation was successful.

Once this is all set up, check out the project examples in these directories:

```
$ ls /usr/src/sense-hat/examples
```

Or copy them to another directory that is easier to get to like your desktop:

```
$ cp /usr/src/sense-hat/examples/python-sense-hat ~/Desktop -a
```

Run those projects with Python by going to the directory and running the file name like this:

```
$ python3 rainbow.py
```

Learn more:

- Read the full API documentation on how to use the Sense Hat.
- See example projects on Github

GPIOZero Library

The GPIOZero library is a Python extension library that allows you to use your Raspberry Pi pins and connect them to sensors, motors, buttons, LEDs, etc.

Here is the documentation with example code for the usage of this library, which comes pre-installed on Raspberry Pis: https://gpiozero.readthedocs.io/en/stable/

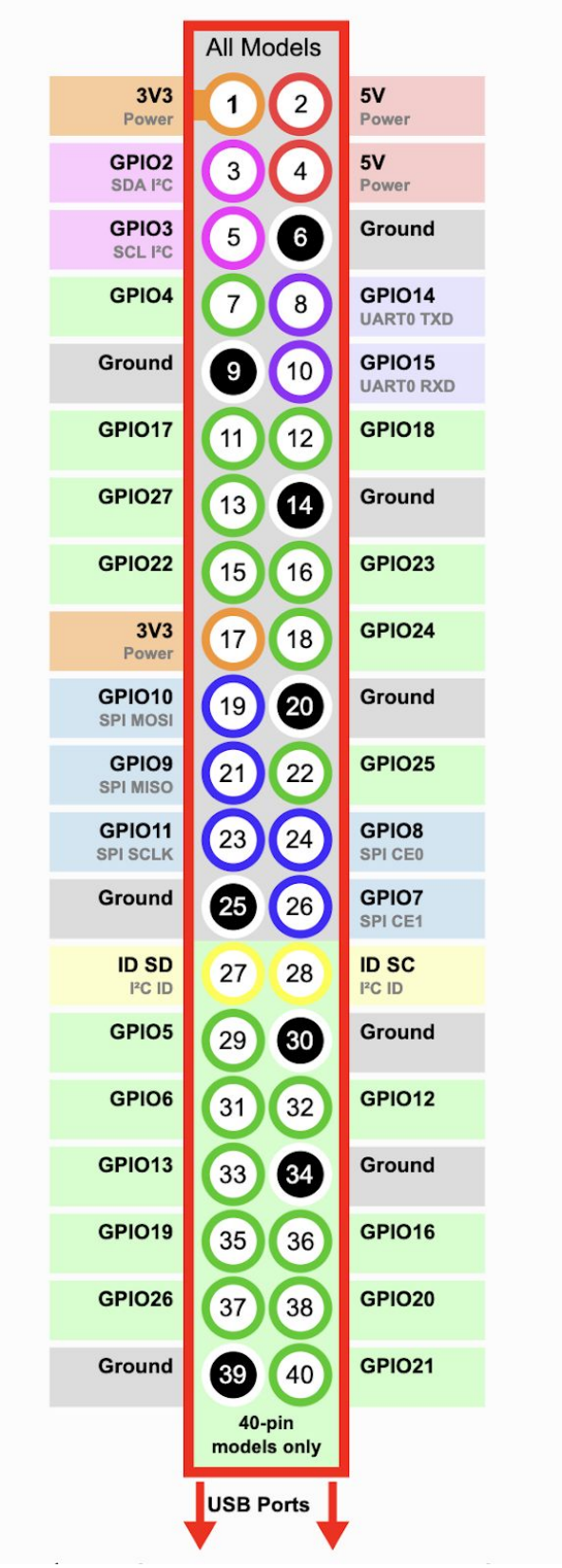
The GPIOZero library is used inside a Python script. To import use this command:

```
import gpiozero
```

You can also import just the component you need, for example, for a LED:

```
from gpiozero import LED
```

Before you connect any components, you must first determine the pin layout on your Pi. Use this guide to help you find the correct pins. Using these incorrectly can result in you Pi being ruined or shorted out.



As you can see, there are two numbering systems for the pins. **Pins 1, 2, 4,** and **17** all provide power, but only **2**, **4** supply 5 Volts of power.

Pins **6, 9. 14, 20, 25, 30, 34** and **39** are Ground pins.

The remaining pins are the GPIO pins. These follow the same numbering system, but the GPIOZero library follows another numbering system called BCM.

When using the GPIOZero library, you make sure you declare or assign the component to the pin you want to use. All GPIO zero pins will supply power.

Below is **blink.py** a simple script that shows how to make an LED at GPIO17 blink. GPIO17 is pin # 11.

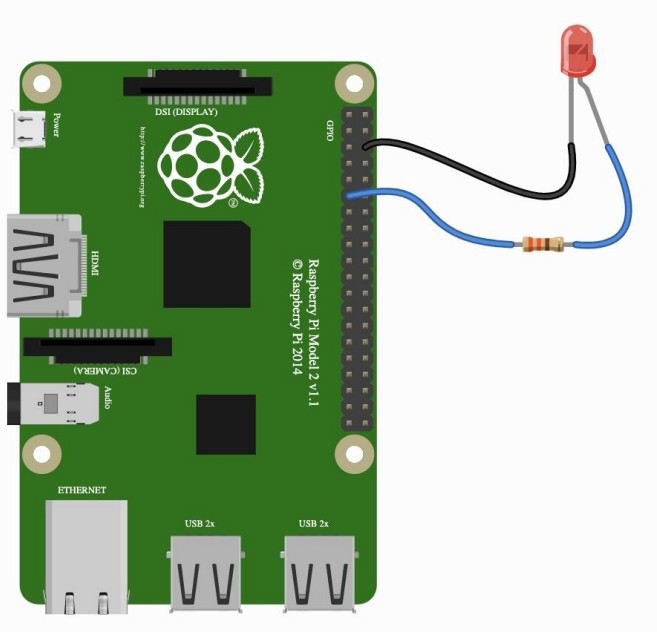
```
#blink.py
from gpiozero import LED
from time import sleep

myLed = LED(17)

while True:
    myLed.on()
    sleep(1)
    myLed.off()
    sleep(1)
```

This example shows how to use the time library's sleep method to make the LED blink at 1 second intervals.

Below is a wiring diagram of this same connection on the PI. Notice the use of a **330 Ohm** resistor to protect against a power surge.



Notice there is a longer leg, or lead on the LED. this is the (+) end and is attached to the correct pin GPIO17. The black wire (-) is grounded on the 6th pin.

To run this code and make the LED blink, simply open the terminal and navigate to the directory where the script is saved and run python3:

```
$ python3 blink.py
```

For a full overview of the Library, including how to use it with different components, visit the official documentation page.

Links and Resources:

- Raspberry Pi Official site: Raspberypi.org | Project tutorials (Python) | Downloads
- YouTube: Playlists | Raspberry Pi Channel |
- Make Use Of: Raspberry Pi Terminal commands you should know.
- Hackaday: Raspberry Pi projects/articles
- Github | GPIOZero sample scripts