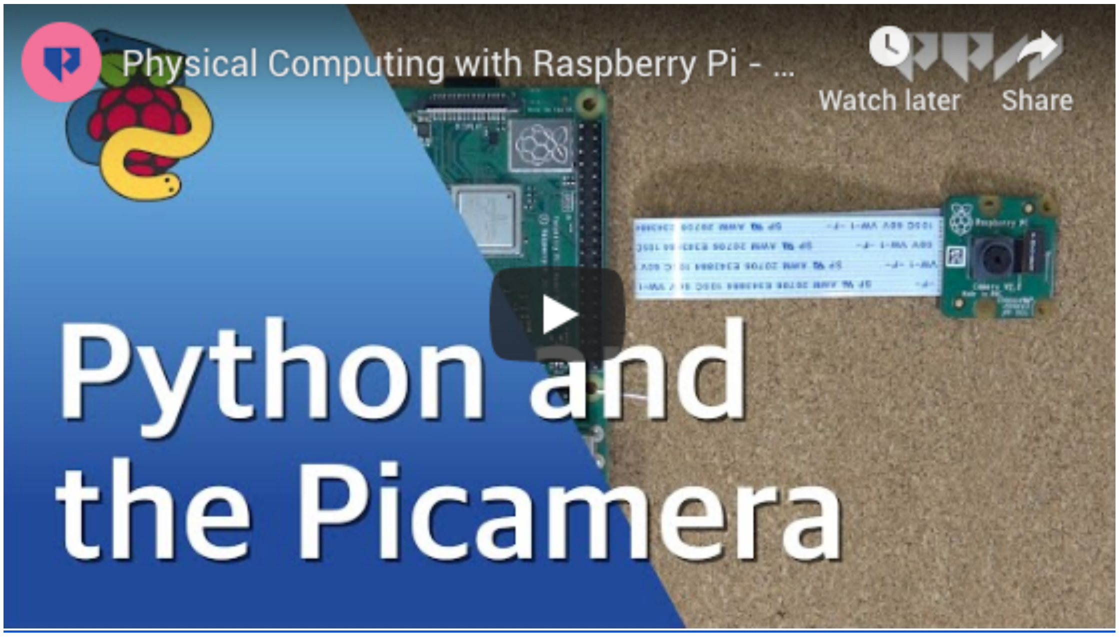
The Picamera



One of the most exciting parts of Raspberry Pi is the ability to add a camera. The Picamera can take photos and record video. You can control your camera with just a few lines of code and you can even create your own photobooth, portable camera, or time-lapse camera system by using the Picamera and your new Python skills.

Materials

- Raspberry Pi Model 3 B+ 8GB
- Micro SD Card

- Picamera
- Micro USB Power Supply
- Mouse
- HDMI Cord
- Keyboard
- HDMI Monitor
- Computer

Key Concepts

- Picamera
- Effects
- Video

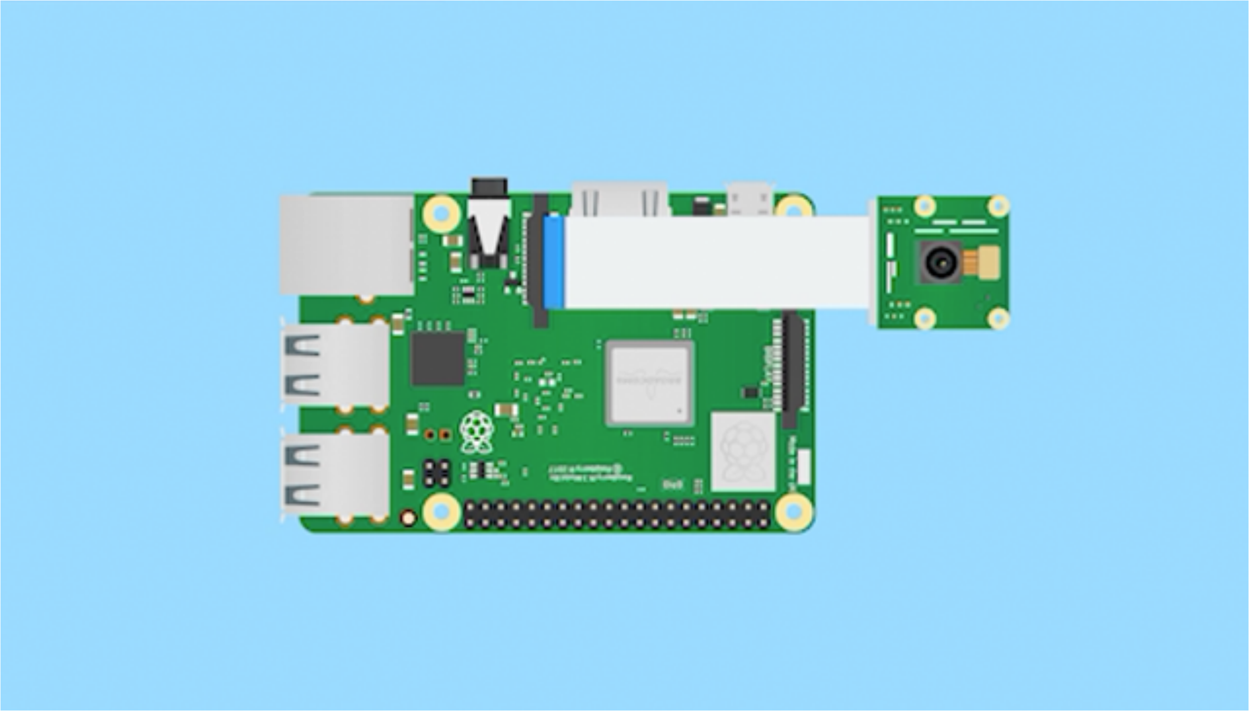
Project: Taking photos with Picamera

The Hardware

Setting up the Picamera

1. One of the best part of a Raspberry Pi is that you can connect a camera and create a bunch of very cool projects with it. Let's start by connecting the camera to our Raspberry Pi board. Check out the picture for help.

   Make sure your pi is unplugged from power before you add the camera. You will notice the metallic side and the blue side of the picamera ribbon. You want to make sure that the blueside is facing the headphone port and the metallic side is facing the HDMI port. Be very gentle when you lift the lid of the camera port. **You do not need to pull very hard to get it up. You can break it with too much force.** Slide the ribbon all the way down and then close the camera port. Once everything is plugged in, you can turn on the power to the Raspberry Pi.



2. When the Pi boots to the desktop, go to the Menu, go to Preferences, and select Raspberry Pi Configuration. Click the interfaces tab and check the camera section. It should be enabled. If not, enable it. You will need to reboot the computer. If it was already enabled, close everything and get ready to take some pictures.

3. Point your camera somewhere, open the terminal, and type the following,

   ```
   raspistill -o picone.jpg
   ```

   This is telling the computer to take a picture. Wait for a second after you hit enter and the picture will be taken. We called it picone, but you could call it whatever you'd like. It is saved in ***/home/pi***.

4. To view the photo, you need to type in the following in the terminal, gpicview picone.jpg

   ```
   gpicview picone.jpg
   ```

5. This will bring up the image. If it is blurry, make sure to wipe the lens clean if you touched it with your fingers during installation or if you left the plastic protective piece on the lens.

Project: Taking photos with Picamera

Let's get started with Python script now that we know our camera is working well. Let's do a quick preview test of the camera. Type the following in a new Thonny tab,

```
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview()
sleep(10)
 camera.stop_preview()
```

We set the preview to last for 10 seconds. You can adjust that if it seems too long. This is always a nice code to run to see if your camera is in the right spot for a project. We use simple start and stop commands with the camera function.

Project: Taking photos with Picamera - Selfie Time!

To take a picture, type the following,

```
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview()
sleep(5)
camera.capture('/home/pi/Desktop/image.jpg')
 camera.stop_preview()
```

In this code, we are starting the camera preview, letting it stay on for 5 seconds, then taking a picture. After the picture is taken, it is saved and named based on what we type in the parentheses after camera.capture. We are saving it on the desktop and calling it image.jpg. After the picture is taken and stored on the desktop, the preview will end and the program will end. Take as many pictures as you like. The photo will be written over after every picture.

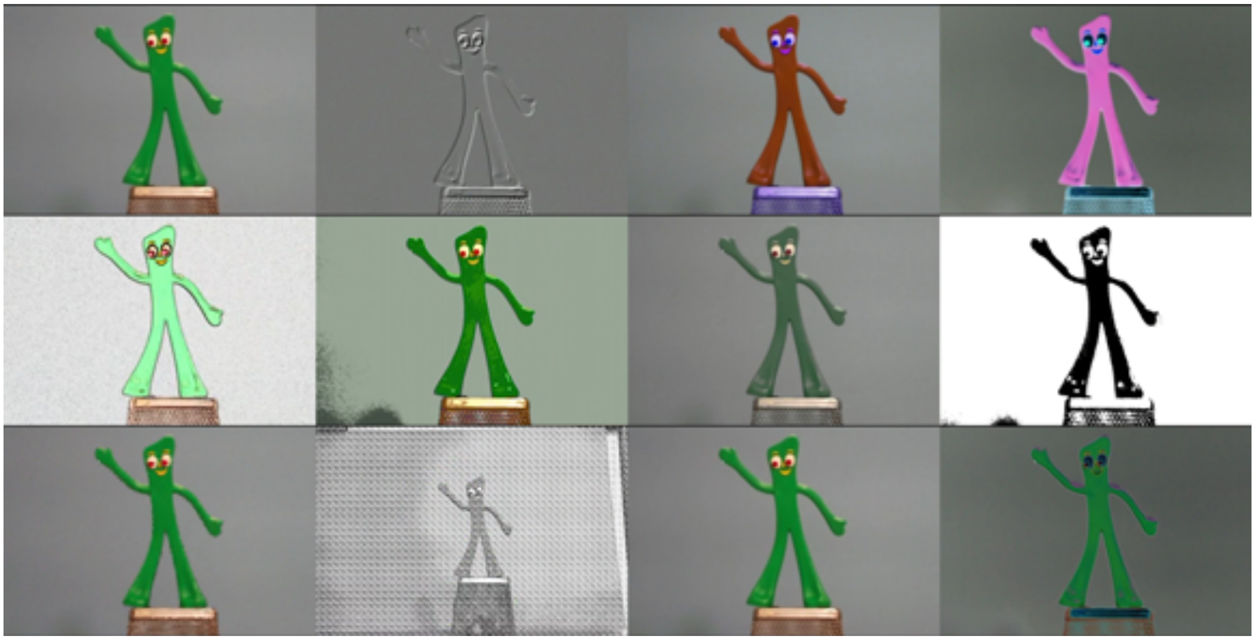
Project: Taking photos with Picamera -Multiple pictures at once

You might want to take a series of photos and have all of them saved. Here is how you can do that.

```
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview()

for i in range(5):
  sleep(5)
  camera.capture('/home/pi/Desktop/image%s.jpg' % i)

camera.stop_preview()
```

Notice that we create a for loop with i in range of 5. We also use sleep to determine how much time is between each photo. We then use %s token to name each photograph taken. The %s will be replaced with whatever we place after % in the file name. In this case, we set for 5. That means we are going to see image0.jpg, image1.jpg, and so on.

Project: Taking photos with Picamera -Image Effects



You can actually change the effects of the photos you take. These are cool filters to spice up your photos. Here is a list of the filters,

```
none, negative, solarize, sketch, denoise, emboss,
oilpaint, hatch, gpen, pastel, watercolor, film, blur,
saturation, colorswap, washedout, posterise, colorpoint,
 colorbalance, cartoon, deinterlace1, deinterlace2
```

You would add one line of code to change the effect of your photo. Here is a simple picture code to add a 'colorswap' effect,

```
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview()

camera.image_effect = 'colorswap'

camera.start_preview()
sleep(5)
camera.capture('picture1.jpg')
 camera.stop_preview()
```

We added `camera.image_effect = 'colorswap'`. You can change the effect by choosing from the list above, or you can have the camera take one picture using each effect.

Project: Taking photos with Picamera -Image Effects 2

```
from picamera import PiCamera
from time import sleep
camera = PiCamera
camera.start_preview()
sleep(5)
for effect in camera.IMAGE_EFFECTS:
  filename = "image_%s.jpg" % effect
  camera.image_effect = effect
  camera.capture(filename)
  sleep(1)
camera.stop_preview()
```

**MAKE SURE TO PUT FILENAME IN DOUBLE QUOTES!
This code will take a picture and use each one of the effects.
They will be labeled by effect, so you know which ones you like.

Camera Settings

Sometimes, you are going to want to tweak the settings for the camera. Here are a list of settings that can be adjusted,

- camera.brightness = 50 (0 to 100)
- camera.sharpness = 0 (-100 to 100)
- camera.contrast = 0 (-100 to 100)
- camera.saturation = 0 (-100 to 100)
- camera.iso = 0 (automatic) (100 to 800)
- camera.exposure_compensation = 0 (-25 to 25)
- camera.exposure_mode = 'auto'
- camera.meter_mode = 'average'
- camera.awb_mode = 'auto'
- camera.rotation = 0
- camera.hflip = False
- camera.vflip = False
- camera.crop = (0.0, 0.0, 1.0, 1.0)

The last few are the ones you are most likely to use for your picamera. You might need to flip/rotate the camera because of the way your project is set up with the camera. You

can do that by changing the `False` to `True` or adjust the rotation from `0` to the degree that meets your needs.
You would add this line after `camera = PiCamera()`

Project: Recording Videos with Picamera

The Picamera can do more than just take photos. It can record video. Enter in the following code,

```
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview()
camera.start_recording('/home/pi/video.h264')
sleep(10)
camera.stop_recording()
camera.stop_preview()
```

You will notice that the code is very similar to the code for taking a picture. We changed camera.capture to `camera.start_recording` and we changed the file to end with `.h264`. Sleep is used to determine how long the video will record. We stop the recording and the preview when we are done.
To play the video, you need to go to the terminal and type in

```
omxplayer video.h264
```

You can add all of the same effects to a video recording that you can for picture taking. If you are having stabilization issues, add this line of code,

```
camera.video_stabilization = True
```

Challenge

- How could you use the camera and Python to create a time-lapse photo sequence?