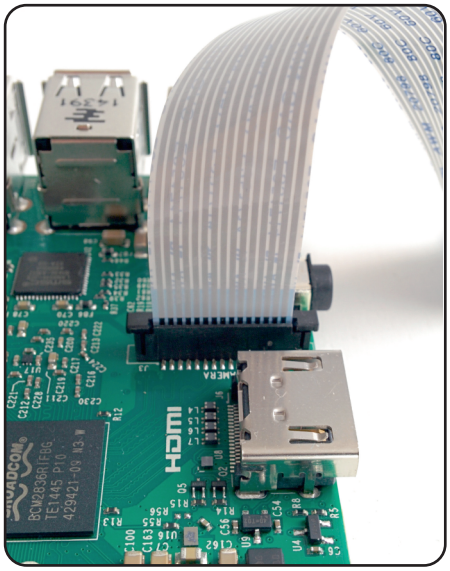
CAMERA WORKSHOP

This workshop guides you through setting up the Raspberry Pi camera module, taking pictures and video using the Python picamera module, connecting a physical button with the GPIO pins and programming it to control the camera.

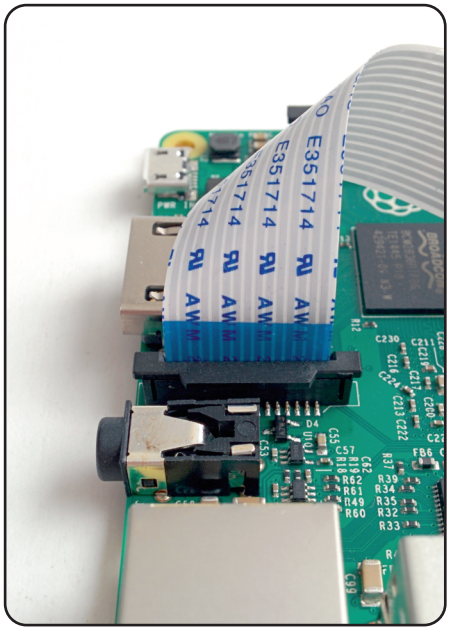
INSTALLATION

1. The camera port is the thin black port next to the Ethernet port
2. Lift the tab on the top of the connector
3. Place the camera's connecting strip in the connector with the blue side facing the Ethernet port
4. Ensuring the strip stays in the connector, push the tab back down



STATIC WARNING!

The camera is a static sensitive component. Try to avoid touching the circuit board itself (although holding it at the edges is fine). Also, whilst the camera is active, try to avoid letting the camera board come into contact with the Pi (or conductive parts of the Pi like the surrounds of the USB connectors).



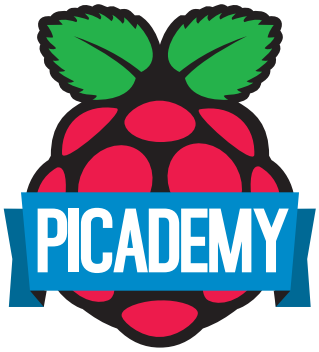
ACTIVATION

You won't need to do this during picademy, but if you're dealing with a new Pi, this is how to activate the camera module:

1. In LXTerminal enter the command `sudo raspi-config`
2. In the menu that appears, move down to `Enable Camera` and press Enter
3. Move right to `Enable` and press Enter again
4. Move right to `Finish` and press Enter
5. Select `Yes` to reboot the Pi

NOTE

It is not necessary to switch off the Pi when installing or removing the camera module. However, you must ensure that the camera is not active when removing it.



CAMERA WORKSHOP

TESTING

After you have rebooted and logged in once more, it's time to test that the camera is working! Start LXTerminal and enter the command: `raspistill -o image.jpg`

If everything is working correctly you should see a red LED light up on the camera module, and a live view of what the camera is looking at should appear on the monitor. After a short delay the camera should capture its view and save it as image.jpg before shutting down again.

CAPTURING AN IMAGE

Now you've got your camera working, it's time to write some code of your own to control it. Start Python with **Menu > Programming > Python 3**. In the Python window, select **File > New Window** and in the window that appears, enter the following script (remember that case and indentation are important!):

Select **File > Save** from the menu and give your script a name like `camera1.py`. Now select **Run > Run Module** from the menu and all being well your camera should do the same thing as

```
from time import sleep
from picamera import PiCamera

camera = PiCamera()
camera.start_preview(alpha=192)
sleep(5)
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

when you tested it above; the red LED should light up, the preview should appear on the screen and after 5 seconds the camera should capture the scene to `image.jpg`.

NOTE

You can experiment with the script to try taking a picture without the preview running, or adjust the delay before the picture is captured.

TRANSPARENT?

You may being wondering why the preview produced by the script above is semi-transparent. This is the result of the `alpha=192` parameter; remove this and the preview will be opaque, but in this case you may find it difficult to shut down the script if you made an error!

CAMERA WORKSHOP

PUSH BUTTON CAMERA

The next step is to connect our camera to a push button so that we can control precisely when the image is captured, rather than relying on a delay. Use **File > Save As...** to save your script with a new name like `camera2.py` then modify it to match the code below:

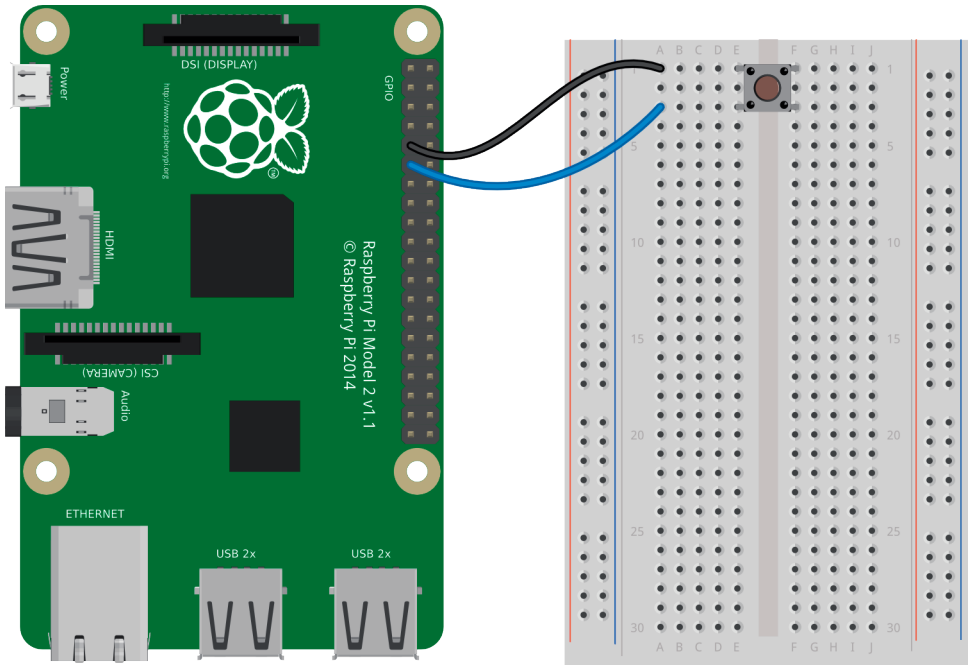
```
from time import sleep
from picamera import PiCamera
from gpiozero import Button

camera = PiCamera()
button = Button(17)
camera.start_preview(alpha=192)
button.wait_for_press()
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

GPIO NUMBERING

In previous sessions you have been using the GPIO pins in BOARD mode, which numbers the pins 1, 2, 3, 4, etc. In this session we are using the GPIO pins in BCM (Broadcom) mode. This mode numbers the pins by their connection to the processor, hence the numbers are non-sequential. You may need to flip your pin template accordingly.

Now place a push button on your breadboard, and connect one side of it to GPIO pin 17, and the other to a GPIO ground pin as shown in the diagram. When you run the script you should find that the preview starts, but the script then waits for you to push the button before capturing an image.



CAMERA WORKSHOP

SELFIE CAMERA

With a simple modification, we can change our push-button camera into a selfie camera. Once again, use **File > Save As...** to save your script as something else like `camera3.py`. Now add a delay after detecting the button push, but before the capture:

```
from time import sleep
from picamera import PiCamera
from gpiozero import Button

camera = PiCamera()
button = Button(17)
camera.start_preview(alpha=192)
button.wait_for_press()
sleep(5)
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

NOTE

In this workshop we're using a button to trigger the camera. However, there's no reason you couldn't use:

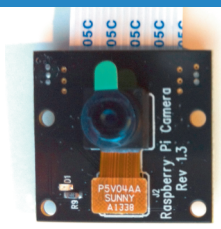
- A pressure pad
- An infra-red motion sensor
- An ultra-sonic distance sensor
- A temperature sensor
- A block in Minecraft (when it's hit)
- Anything else you can connect to your Pi!

EXTRAS

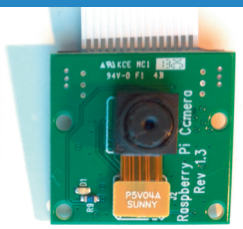
Can you extend the script to take a burst of three pictures when the button is pressed?

NIGHT VISION!

The Pi's camera module is available in two variants: the regular camera module and the PiNoIR. The PiNoIR lacks the infra-red filter usually found on cameras, which means it can see into the near infra-red range (<950nm). This isn't enough for heat detection, but it is good enough for night-vision with appropriate illumination (e.g. 850nm LEDs).



PiNoIR camera module



Regular camera module

CAMERA WORKSHOP

CAMCORDER

The Pi's camera is also capable of recording video. Once again, use **File > Save As...** to save your script under a new name like `camera4.py`. Now use the `start_recording` and `stop_recording` methods instead of capture, as shown in the script below:

```
from time import sleep
from picamera import PiCamera
from gpiozero import Button

camera = PiCamera()
button = Button(17)
camera.start_preview(alpha=192)
button.wait_for_press()
camera.start_recording('/home/pi/Desktop/video.mp4', format='h264')
sleep(5)
camera.stop_recording()
camera.stop_preview()
```

EXTRAS

Can you make a slow-motion video using some of the faster framerates in the table below?

After recording your video you can double click on it to view it in the Pi's web-browser, or play it from the command line with `omxplayer video.mp4`. Just as with capturing images, you can trigger video recording with almost anything you can think of: pressure pads, infra-red motion sensors; with a bit of effort you can even use the camera itself to perform motion detection.

You can also record to things other than files. For example, you can stream video to other machines over the network, or record to a circular buffer in memory!

FIND OUT MORE

You can find out lots more about the camera library, including many code recipes at:

- picamera.readthedocs.org
- github.com/waveform80/picamera
- github.com/waveform80/picamera_demos

SPEED LIMITS

The Pi's camera module is based on the Omnivision 5647 sensor. This is the "little brother" of the sensor used in the iPhone 4 (without auto-focus or LED flash). The camera's video recording framerate limits at various resolutions are given in the table to the right.

* At the highest resolution only MJPEG videos are supported, not MP4.

Resolution	Framerate
2592x1944 (~5Mp)*	15fps
1920x1080 (1080p)	30fps
1280x720 (720p)	49fps
640x480 (VGA)	90fps (!)