USING THE CAMERA MODULE

7. CONNECTING THE CAMERA MODULE

WARNING: The Raspberry Pi must be **shut off** before connecting the camera module. *Failing to do so will destroy both the RPi and the camera.* Ensure the power cable is disconnected (after safely shutting down the Pi).

a. Locate the camera port and connect the camera (highlighted in the image). Carefully raise the lock with your fingertips. Feel free to ask for help.

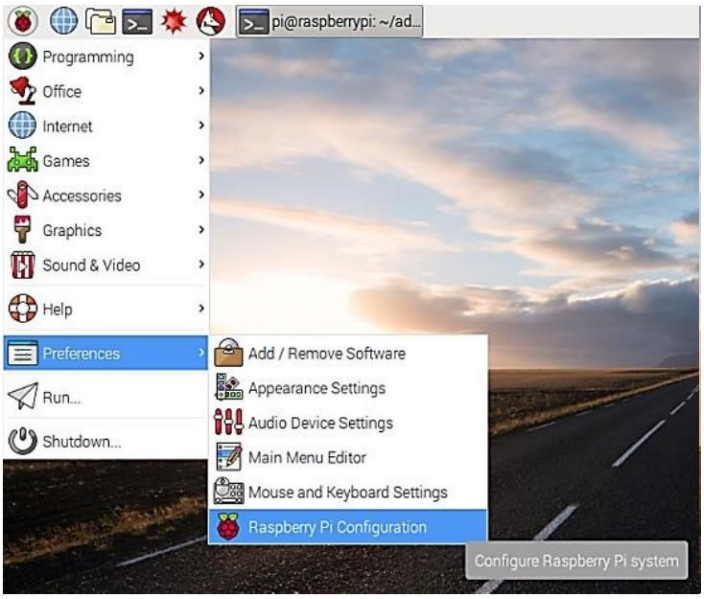
b. Feed the connections into the slot. Make sure the metal contacts on the cable line up with the contacts inside the slot. Lower the black lock back once the camera cable is in place.

2. CONFIGURING THE RASPBERRY PI

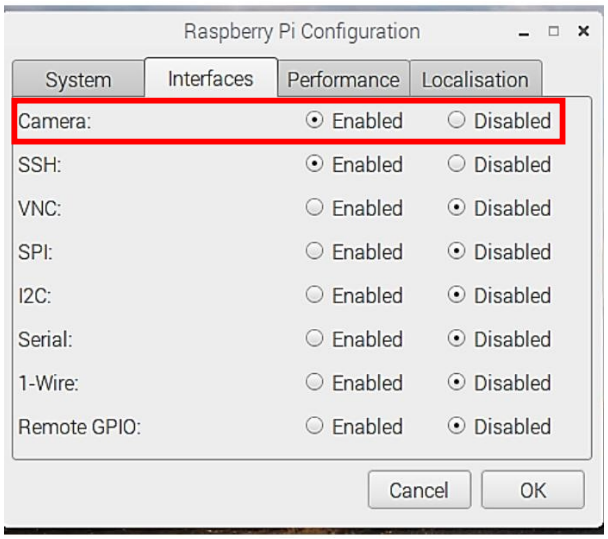
a. Power on the Raspberry Pi (by plugging in the power cable)

b. Open the **Raspberry Pi Configuration Tool** from the main menu:

Start > Preferences > Raspberry Pi Configuration



c. In the Interfaces tab, ensure the camera is enabled. Then, hit OK and restart the Pi.

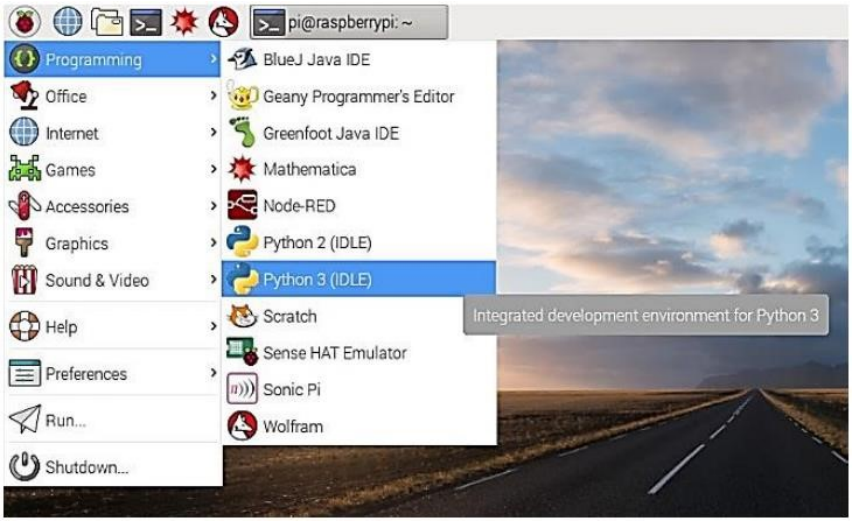


3. Displaying The Camera Preview

We can write a python program to display a preview of what the camera sees. To do this, we use the `PiCamera` library (already installed)

a. Open the IDLE IDE for **Python 3** in the main menu. Open a new file and save it.

Start > Programming > Python 3 (IDLE)



WARNING: Do NOT save it as `picamera.py`. (This will overwrite the library file.)

b. Enter the following code and execute it (by pressing F5):

```
from picamera import PiCamera          # for accessing the camera
from time import sleep                 # allow the program to wait

# Initialises the camera and loads the preview for 5 seconds
camera = PiCamera()                    # instantiate a PiCamera object
camera.start_preview()
sleep(5)                               # Camera will stay on for 5 seconds
camera.stop_preview()
```

c. Try modifying the code above by adding the following lines:

You can rotate the image by changing the rotation attribute of the camera object. This must be done before calling `camera.start_preview()`.

```
camera.rotation = 180                  # Rotates 180 degrees clockwise
```

You can change the transparency of the preview by setting the alpha level when calling `camera.start_preview()`:

```
camera.start_preview(alpha=200)        # 0=transparent, 255=opaque
```

4. Taking Still Pictures

We can use the capture function to save still images to a location of our choice.

a. Add the following line to your code to capture an image.

```
camera.capture('/home/pi/Desktop/image.jpg')
```

Important Notes:

- It is important to **let the camera sleep for at least 2 seconds before capturing**. This gives the sensor time to set its light levels.
- All camera.XXXX() function calls must be made between calls to `camera.start_preview()` and `camera.stop_preview()`
- When specifying a path, it must be enclosed in single quotes. Don't forget the first "/"
- The image file **silently** appears on the Desktop. Double-click it to view.
- ***If you get stuck in camera preview:*** (1) press ctrl + alt + t (2) type blindly "pkill python3"

b. Modify your code to take multiple pictures in a row:

```
#the top of your code stays the same as before…
camera.start_preview(alpha=200)
for i in range(5):
    sleep(5)
    camera.capture('/home/pi/Desktop/image%s.jpg' % i)
      # The "%s" is used to name each image differently using the "% i",
      # "%s" is replaced by the loop's index (0 to 4) in the file name.
camera.stop_preview()
```

5. Creating a GIF from Still Images (Optional)

We can create a gif out of these pictures by launching an application from the Terminal.

a. Launch the terminal (click the terminal icon on the taskbar)

b. Type the following two commands and hit Enter after each one:

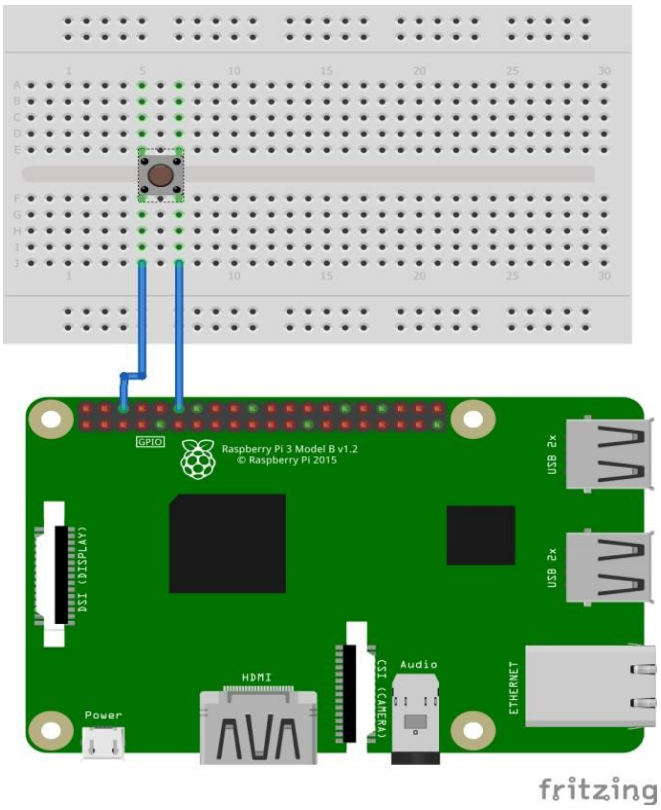
```
sudo apt-get install imagemagick
cd Desktop
convert -delay 10 -loop 0 image*.jpg animate.gif
```

- The first line installs the required software (already done)
- `-loop 0` means the gif loops forever.
- `-delay 10` makes the delay one 10th of a second.
- Saves to `animate.gif` on the Desktop. Be patient, as this will take some time.

c. View the GIF by opening the file on the desktop once this is complete.

6. MINI-PROJECT: DIY CAMERA

In a new Python file, write a program which lets the user view the camera preview and capture the scene by pushing a button. The program should exit once the photo is taken. Use the circuit diagram and code below to get started. You can also draw on code from previous sections.




```
#The following code defines GPIO pins for the button…
from picamera import PiCamera
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
capturePin = 12
GPIO.setup(capturePin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

Hints:

- Use `GPIO.input(pinNumber)` to check the value at the pin. (`True = 1`, `False = 0`)
- When the buttons are pushed down, the Pi will receive a `0` at the corresponding pin. (Since the buttons will connect the pin to ground)
- Use a `while` loop and an `if` statement to repeatedly check for input from the button.

7. Recording Video

Recording video is similar to taking a still, but we replace `capture` with `start_recording` and `stop_recording`. Try the following code which records for 10 seconds:

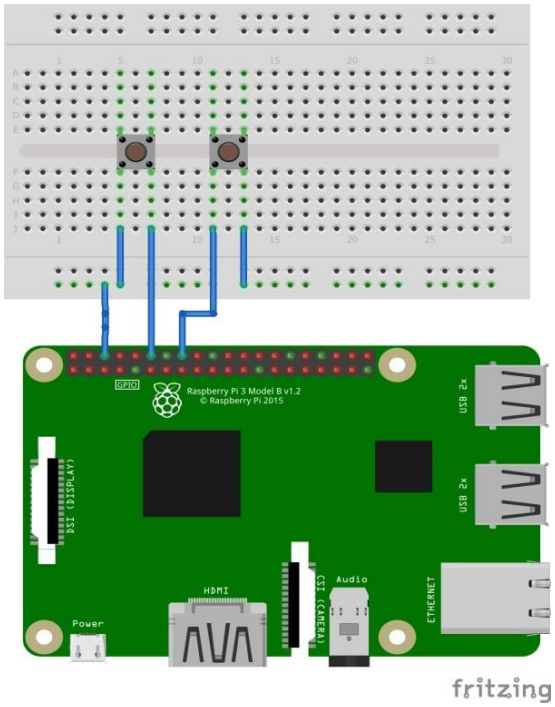
```
camera.start_preview(alpha=200)
camera.start_recording('/home/pi/Desktop/video.h264')
sleep(10)                          # Records for 10 seconds
camera.stop_recording()
camera.stop_preview()
```

To play the video, you'll need to switch to the terminal window to launch a video player. Use the following command:

```
cd Desktop
omxplayer video.h264
```

8. Mini-Project: DIY Camcorder

In a new Python file write a program to let the user push one button to start recording video and another to stop the recording. Use the circuit diagram below to get started. You can also draw on code from previous sections. Feel free to expand on this idea!



Hint: Extend the code from the previous project and set up another pin (here it's pin 16) for the second button.