

# R-Pi

Team Emertxe



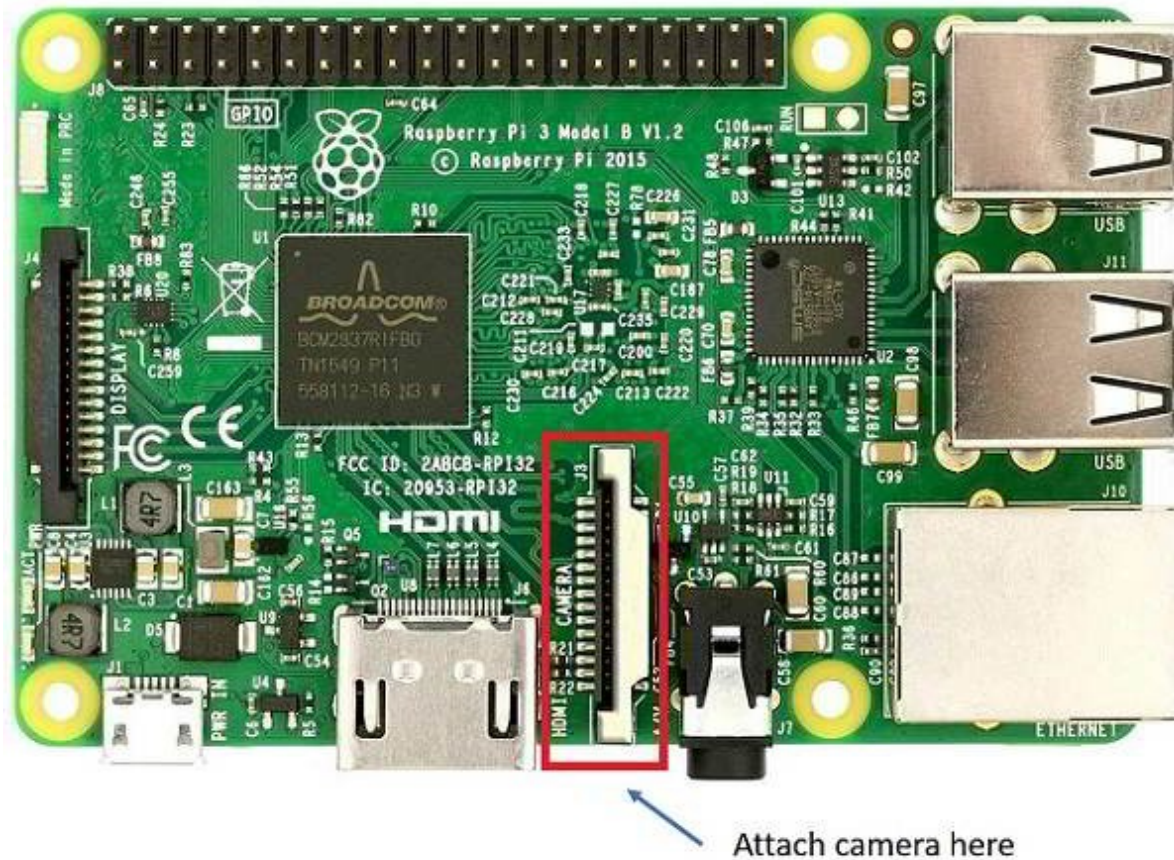
# Interfacing

Camera



# Interfacing: Camera

## Step-1: Connecting to R-Pi

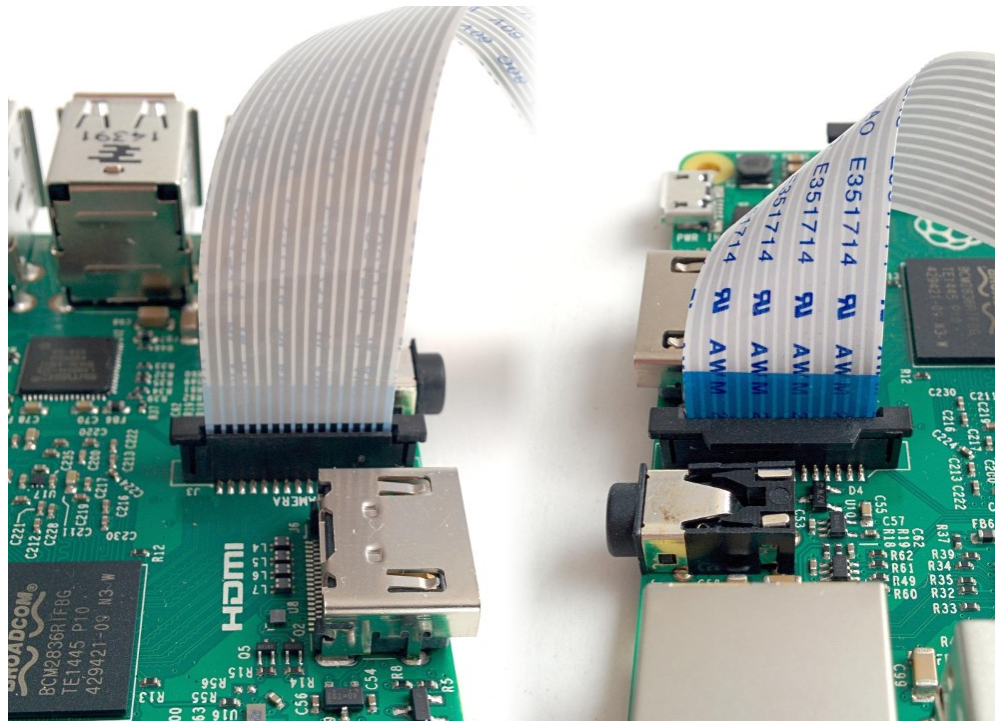


Note: To connect the camera, make sure the r-pi is in switch off state

# Interfacing: Camera

## Step-1: Connecting to R-Pi

1. Gently pull up on the edges of the plastic clip.
2. Insert the camera ribbon; make sure it is the right way round
3. Push the plastic clip back into place

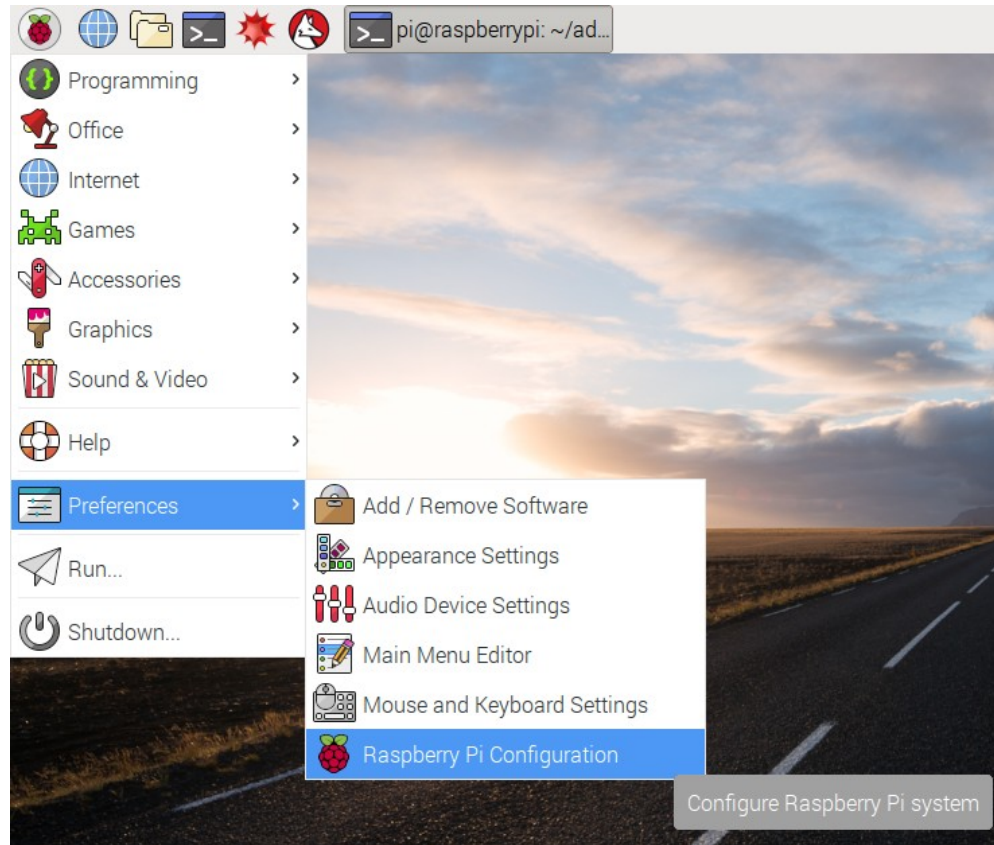


Note: To connect the camera, make sure the r-pi is in switch off state

# Interfacing: Camera

## Step-2: Enable the Camera

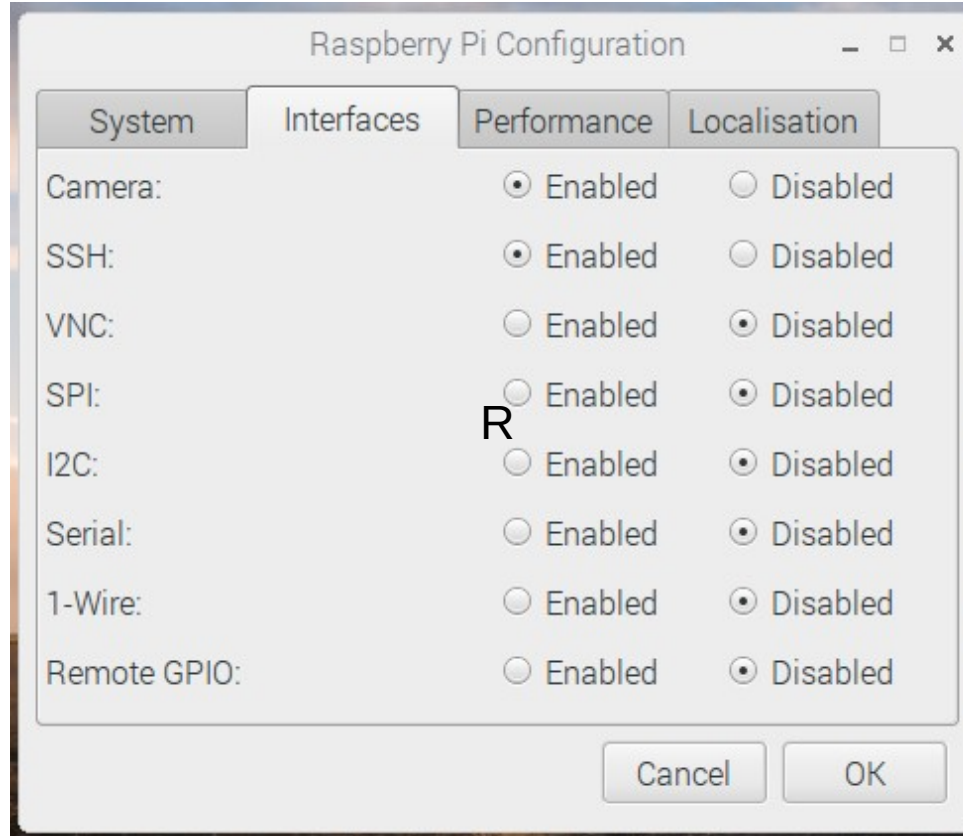
1. Start up the Pi.
2. Open the Raspberry Pi Configuration Tool from the main menu



# Interfacing: Camera

## Step-2: Enable the Camera

3. Ensure the camera software is enabled.
4. Reboot the pi.





# Interfacing: Camera

## Step-3: Coding + Running



1. vi camera.py

```
from picamera import PiCamera
from time import sleep

camera = PiCamera()

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

2. Run the code

```
Python3 camera.py
```

# Interfacing

Coding-2: Rotation





# Interfacing: Camera

## Coding: Rotation



1. vi camera.py

```
from picamera import PiCamera
from time import sleep

camera = PiCamera()

#Image rotation by 180 degree
camera.rotation = 180

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

Note: You can rotate the image by 90, 180, or 270 degrees, or you can set it to 0 to reset.

# Interfacing

Coding-3: Transparency



# Interfacing: Camera

## Coding: Transparency



1. vi camera.py

```
"""
alpha can be any value between 0 and 255.
"""
from picamera import PiCamera
from time import sleep

camera = PiCamera()

camera.start_preview(alpha=200)
sleep(10)
camera.stop_preview()
```

Note: alpha can be any value between 0 and 255.

# Interfacing

Coding-4: Still Pictures



# Interfacing: Camera

## Coding: Still Pictures



1. vi camera.py

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

Note:

It's important to sleep for at least 2 seconds before capturing, to give the sensor time to set its light levels.

# Interfacing: Camera

## Coding: 5 Still Pictures



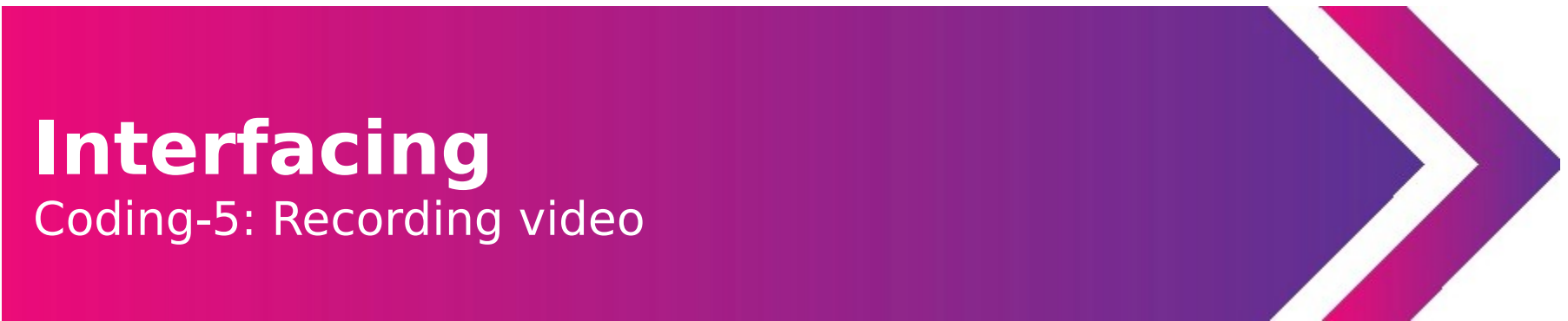
1. vi camera.py

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

The variable `i` contains the current iteration number, from 0 to 4, so the images will be saved as `image0.jpg`, `image1.jpg`, and so on.

# Interfacing

Coding-5: Recording video





# Interfacing: Camera

## Coding: Recording videos



1. vi camera.py

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

2. Type the following command and press Enter to play the video

```
omxplayer video.h264
```

It will record 10 seconds of video and then close the preview.

# Interfacing

Coding-6: Effects



# Interfacing: Camera

## Coding: Effects



1. The resolution of the capture is configurable.
2. By default it's set to the resolution of your monitor, but the maximum resolution is 2592 x 1944 for still photos and 1920 x 1080 for video recording.
3. The minimum resolution allowed is 64 x 64. Try taking one at that resolution.

```
camera.resolution = (2592, 1944)
camera.framerate = 15
camera.start_preview()
sleep(5)
camera.capture('/home/pi/Desktop/max.jpg')
camera.stop_preview()
```

Set the frame rate to 15 to enable this maximum resolution

# Interfacing: Camera

## Coding: Effects



### 1. Adding text to the image with annotate\_text

```
camera.start_preview()  
camera.annotate_text = "Hello world!"  
sleep(5)  
camera.capture('/home/pi/Desktop/text.jpg')  
camera.stop_preview()
```

### 2. You can set the annotation text size with the following code:

```
camera.annotate_text_size = 50
```

# Interfacing: Camera

## Coding: Effects



1. Can alter the brightness setting, which can be set from 0 to 100. The default is 50.

```
camera.start_preview()
camera.brightness = 70
sleep(5)
camera.capture('/home/pi/Desktop/bright.jpg')
camera.stop_preview()
```

# Interfacing: Camera

## Coding: Effects



1. Adjusting the brightness in a loop, and annotating the display with the current brightness level:

```
camera.start_preview()
for i in range(100):
    camera.annotate_text = "Brightness: %s" % i
    camera.brightness = i
    sleep(0.1)
camera.stop_preview()
```

# Interfacing: Camera

## Coding: Effects



1. Adjusting the contrast in a loop, and annotating the display with the current brightness level:

```
camera.start_preview()
for i in range(100):
    camera.annotate_text = "Contrast: %s" % i
    camera.contrast = i
    sleep(0.1)
camera.stop_preview()
```

Valid sizes are 6 to 160. The default is 32.



# Interfacing: Camera

## Coding: Effects



1. can also alter the annotation colours. First of all, ensure that Color is imported by amending your import line at the top:

```
from picamera import PiCamera, Color
camera.start_preview()
camera.annotate_background = Color('blue')
camera.annotate_foreground = Color('yellow')
camera.annotate_text = " Hello world "
sleep(5)
camera.stop_preview()
```

Valid sizes are 6 to 160. The default is 32.

# Interfacing: Camera

## Coding: Effects



1. can use `camera.image_effect` to apply a particular image effect.
2. The options are:
  - `none`, `negative`, `solarize`, `sketch`, `denoise`, `emboss`, `oilpaint`, `hatch`, `gpen`, `pastel`, `watercolor`, `film`, `blur`, `saturation`, `colorswap`, `washedout`, `posterise`, `colorpoint`, `colorbalance`, `cartoon`, `deinterlace1`, and `deinterlace2`.
3. The default is `none`.

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

# Interfacing: Camera

## Coding: Effects



1. Try looping over the various image effects in a preview to test them out:

```
camera.start_preview()
for effect in camera.IMAGE_EFFECTS:
    camera.image_effect = effect
    camera.annotate_text = "Effect: %s" % effect
    sleep(5)
camera.stop_preview()
```

# Interfacing: Camera

## Coding: Effects



1. can use `camera.awb_mode` to set the auto white balance to a preset mode to apply a particular effect.
2. The options are: `off`, `auto`, `sunlight`, `cloudy`, `shade`, `tungsten`, `fluorescent`, `incandescent`, `flash`, and `horizon`.
3. The default is `auto`.

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

You can loop over the available auto white balance modes with `camera.AWB_MODES`

# Interfacing: Camera

## Coding: Effects



1. can use `camera.exposure_mode` to set the exposure to a preset mode to apply a particular effect.
2. The options are: `off`, `auto`, `night`, `nightpreview`, `backlight`, `spotlight`, `sports`, `snow`, `beach`, `verylong`, `fixedfps`, `antishake`, and `fireworks`.
3. The default is `auto`.

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

You can loop over the available exposure modes with `camera.EXPOSURE_MODES`.

THANK YOU