

# Raspberry Pi 4

## 簡介單板電腦



# 你需要的設備

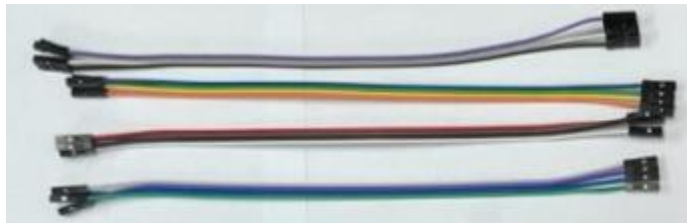
- PC / Mac 電腦
  - USB A 插頭
  - Chrome / Safari / Edge 瀏覽器
- 如沒有 PC / Mac 電腦，你可以使用 iPad + PowerBank 代替

# 設備

Card reader



Jumper



OLED 顯示屏



Logitech Webcam



Raspberry Pi 外殼



Raspberry Pi



SD 卡



行動電源及 USB-C 線



# 設備

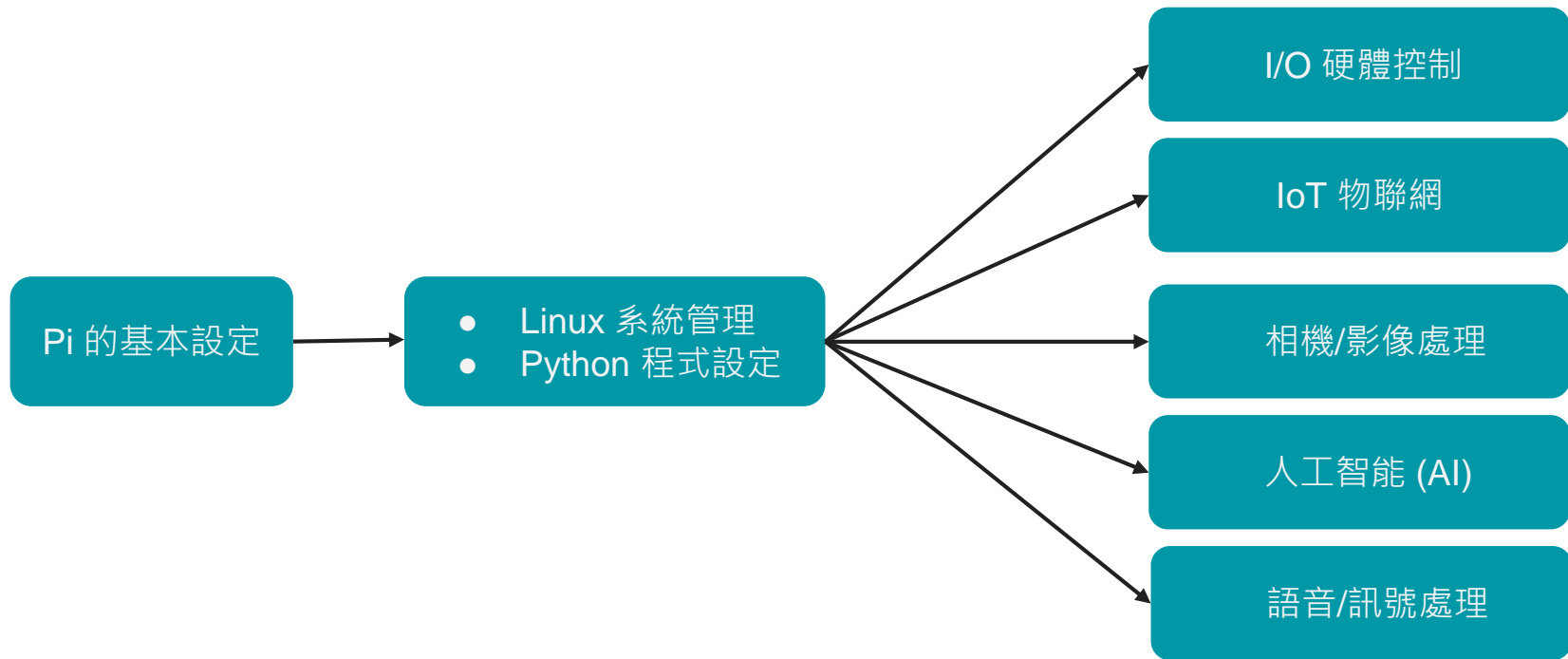


# 安全措施

- 請勿將Raspberry Pi暴露於水
- 請勿將Raspberry Pi暴露於任何熱源
- 避免在Raspberry Pi通電時對其進行處理
  - 包括但不限於拔出 SD Card
- 僅握住邊緣



# 如何學Raspberry Pi?



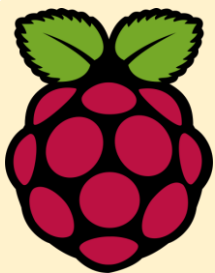
# 本課學習重點

- 什麼是 Raspberry Pi ?
- Raspberry Pi 的用途
- 拆解 Raspberry Pi
- Raspberry Pi 的操作系統
  - 什麼是操作系統 [DSE 必修部分 B.b]
  - Linux 與開源項目 [DSE 必修部分 E.c]
  - 延伸：其他軟件授權制度 [DSE 必修部分 E.c]

## 1.1 什麼是 Raspberry Pi?









# 各種不同種類的電腦



\$8,000+



\$3,000+



\$1,000+



\$1,000+



\$800,000,000+



# 電腦



輸入

處理器

輸出

# 微電腦 / 單版電腦



輸入



處理器

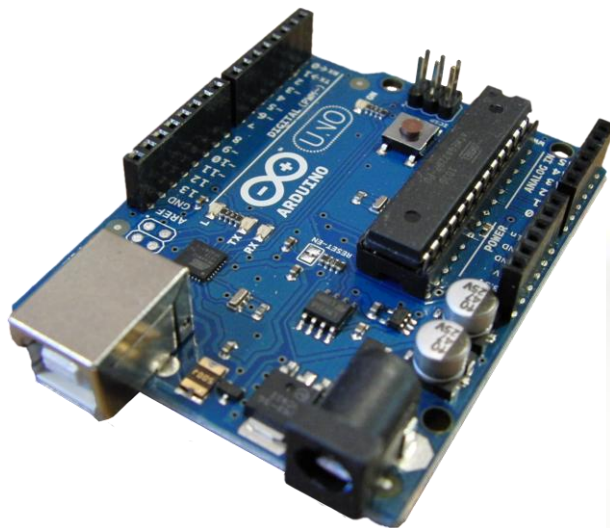


輸出

# 微電腦的種類



Raspberry Pi



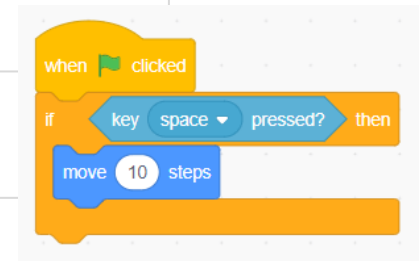
Arduino



Micro:bit

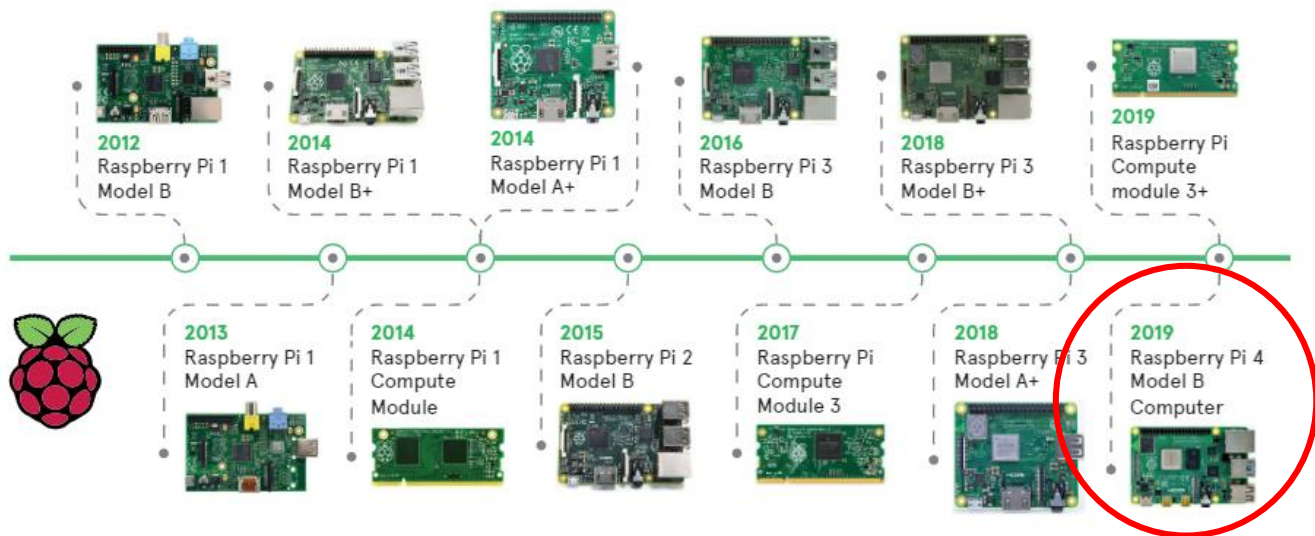


Raspberry Pi	Arduino	micro:bit
微電腦	微控制器	
支援多種程式語言開發	Java, C	Blockly
中高階產品	中階產品	入門產品
\$350+	\$200+	\$150+



# Raspberry Pi 的起源

- 由 Eben Upton 和 Pete Lomas 等人成立於 2008 年
- 旨在降低人們學習電腦和編程的成本







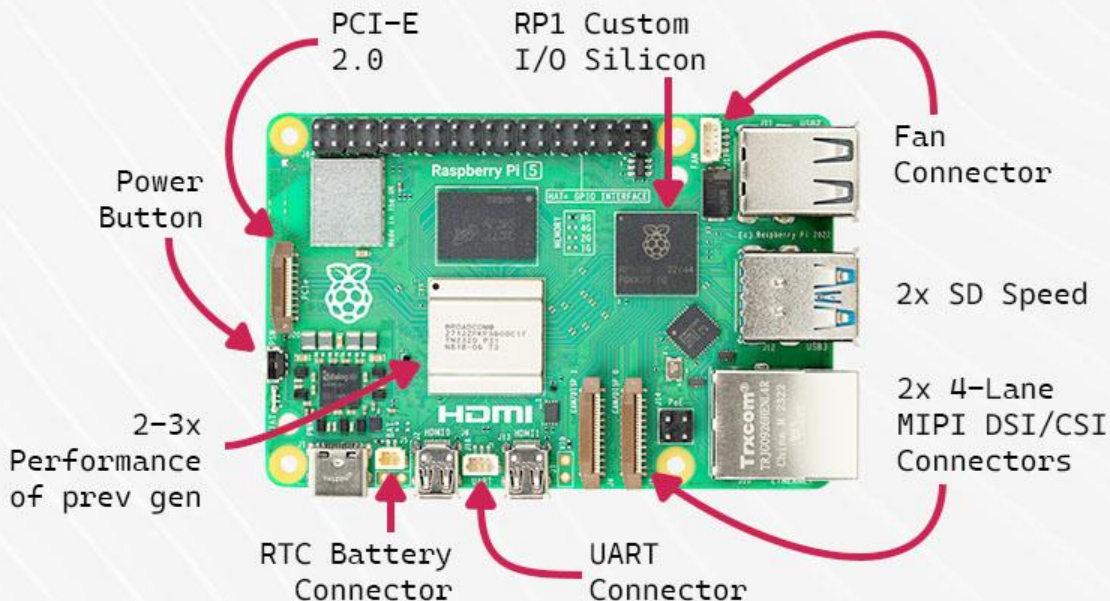
**Raspberry Pi** **5**

# Raspberry Pi 的最新發展

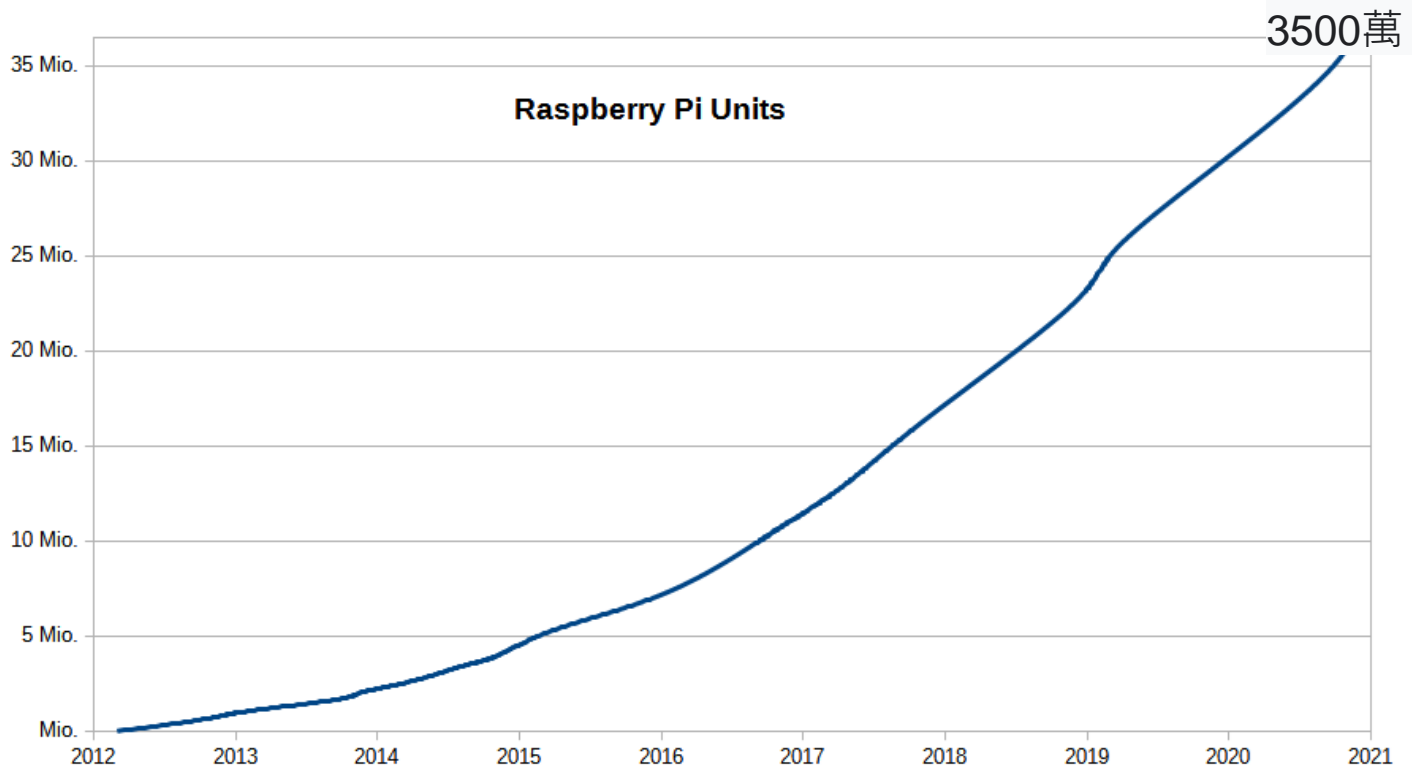


## Raspberry Pi 5

another leap forward  
for the most popular  
single-board computer



# Raspberry Pi 於2012 至2020年間的銷售量



# Maker's Choice



## 1.2 Raspberry Pi 的用途



## (1) 媒體中心





## (2) 遊戲機

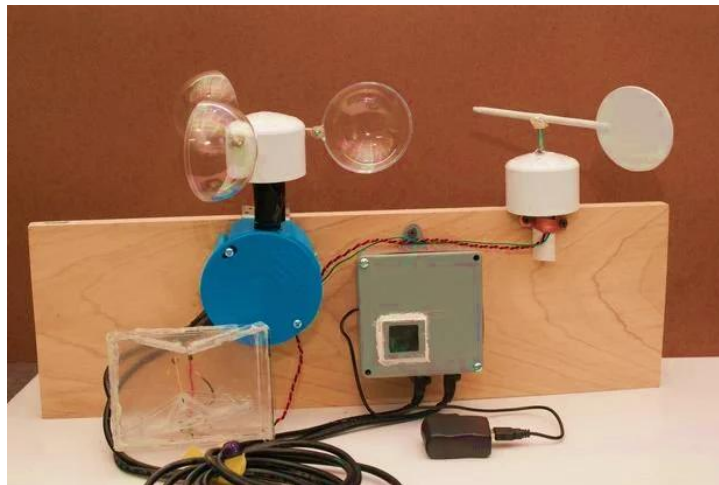


### (3) 閉路電視

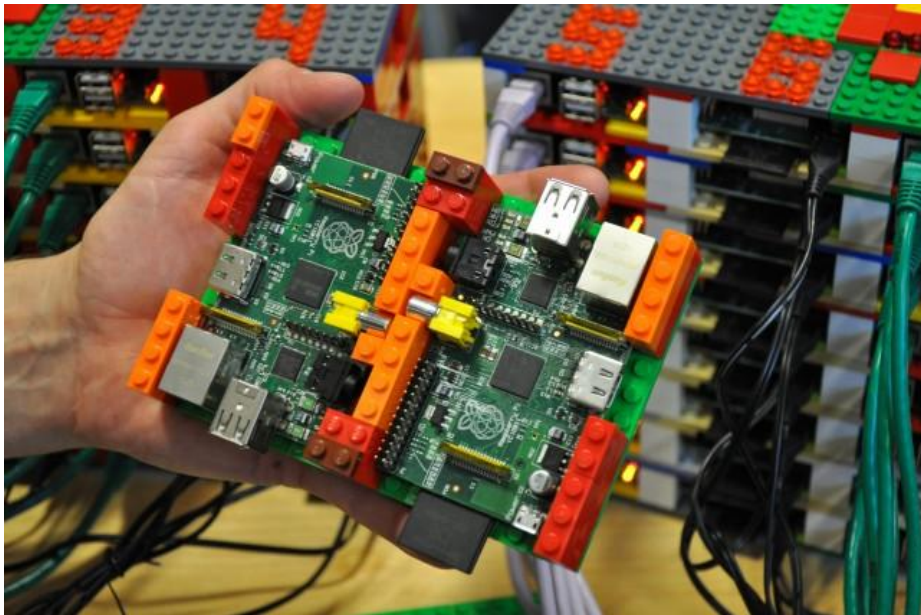




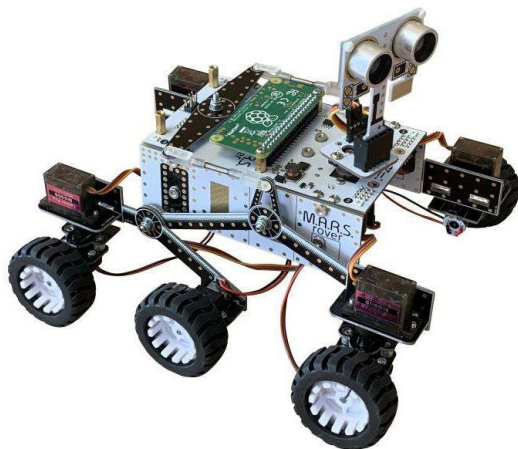
## (4) 氣象站



## (5) 超級電腦



## (6) 機器人項目



## (7) 各種互聯網服務？



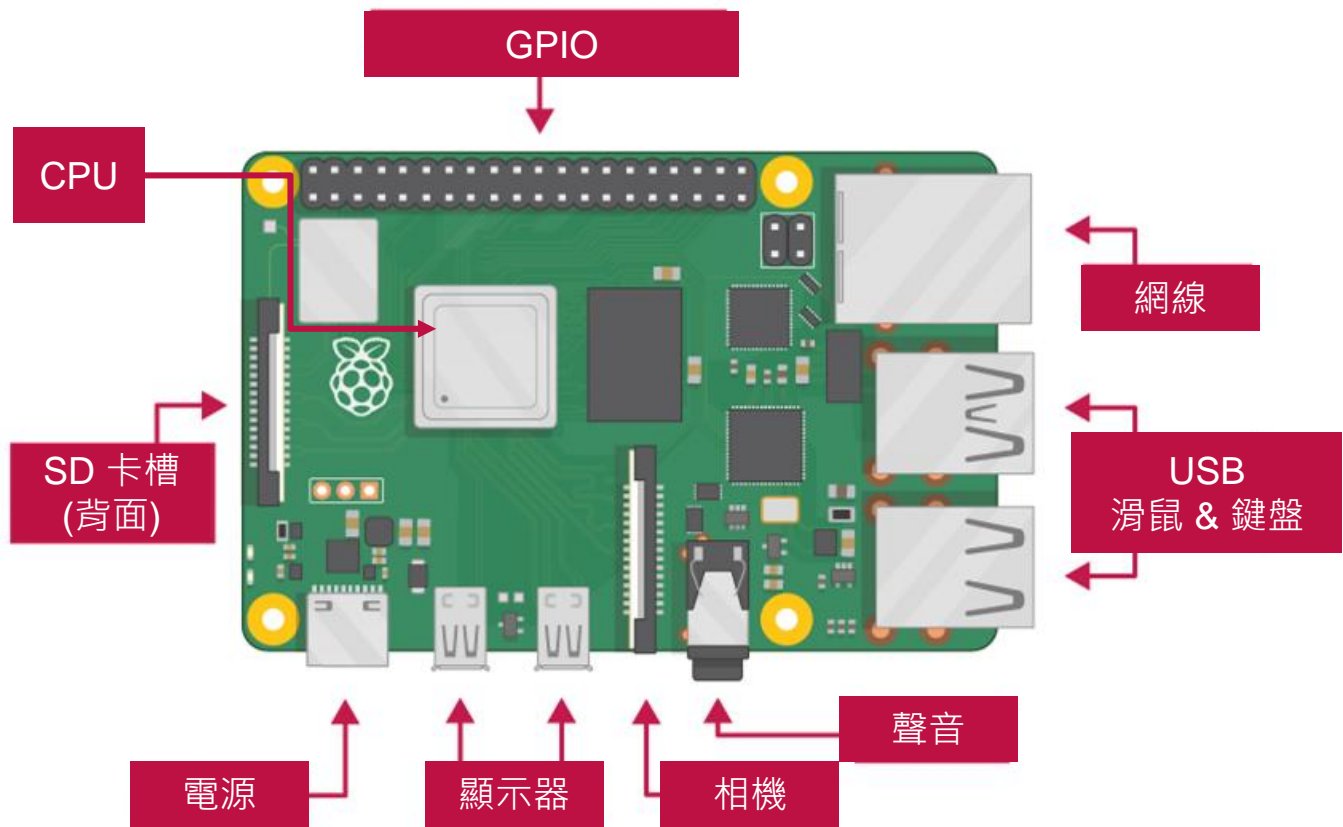
**NETFLIX**



## 1.3 拆解 Raspberry Pi



# 拆解 Raspberry Pi



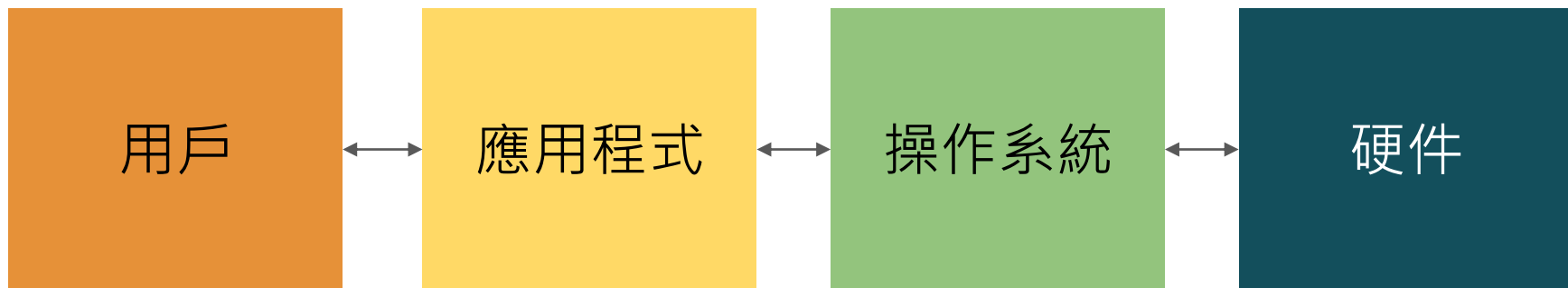
## 1.4 Raspberry Pi 的操作系统





# 操作系統 Operating System (OS)

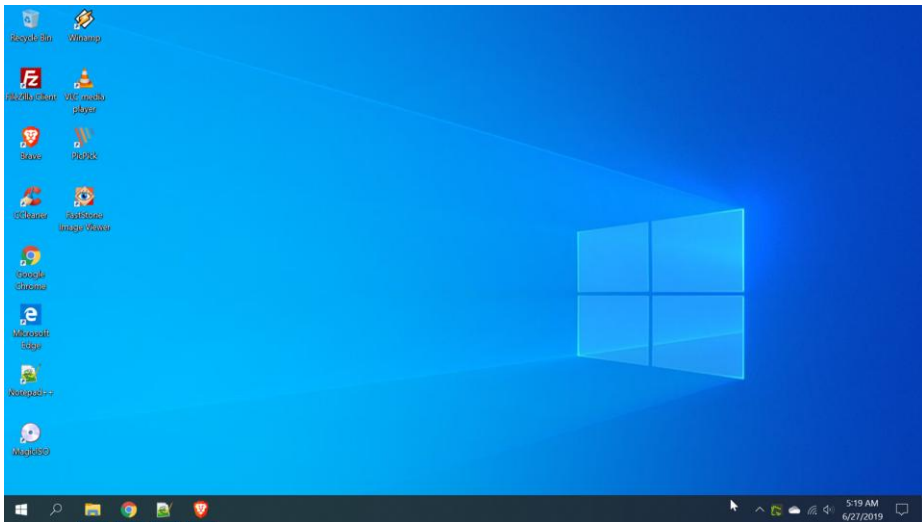
- 使用者和應用程式 (Application Software) 與電腦硬件溝通的橋樑
- 提供讓應用程式可以執行的平台
- 提供一系列的實用程式 (Utility Software) 予使用者監控電腦的硬件





課題：電腦系統基礎 - 系統軟件

# 常見的操作系統



Windows

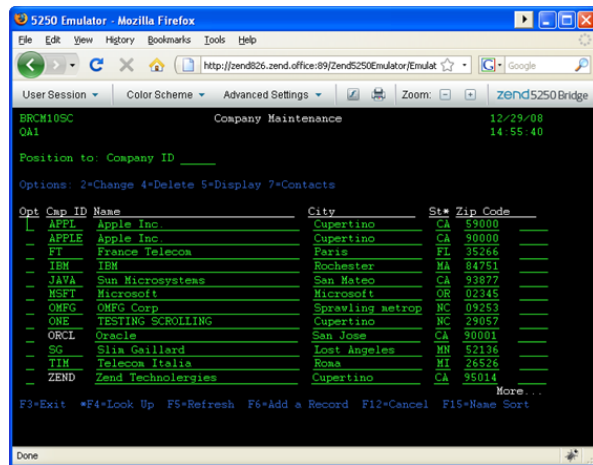
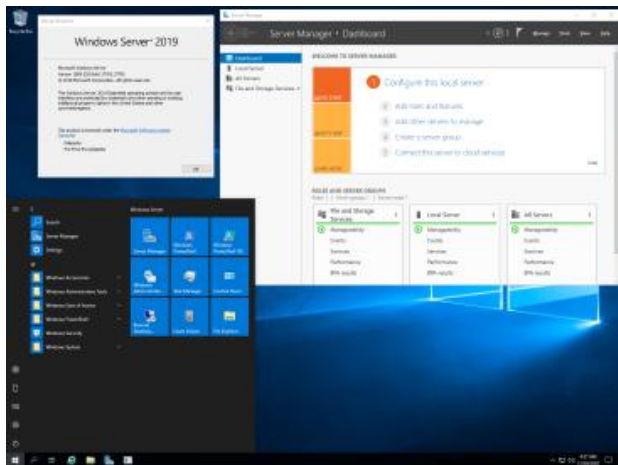


MacOS

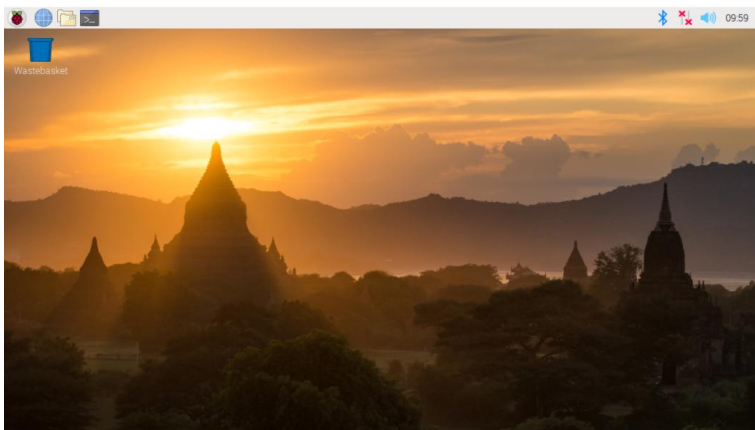
## 其他操作系統 (OS)



# 其他操作系統 (OS)



# Raspberry Pi 的官方操作系統



**Raspberry Pi OS**  
(前稱 Raspbian)

- 基於 Linux 的一個操作系統
  - 什麼是 Linux ?
- 用家亦可以自行安裝其他操作系統

# Linux

- 由Linus Torvalds 於1991年開發出來
- 開放源碼 (**Open Source**) 的操作系統
  - 用GPL (General Public License) 的方式發行這份程序



Linus Torvalds

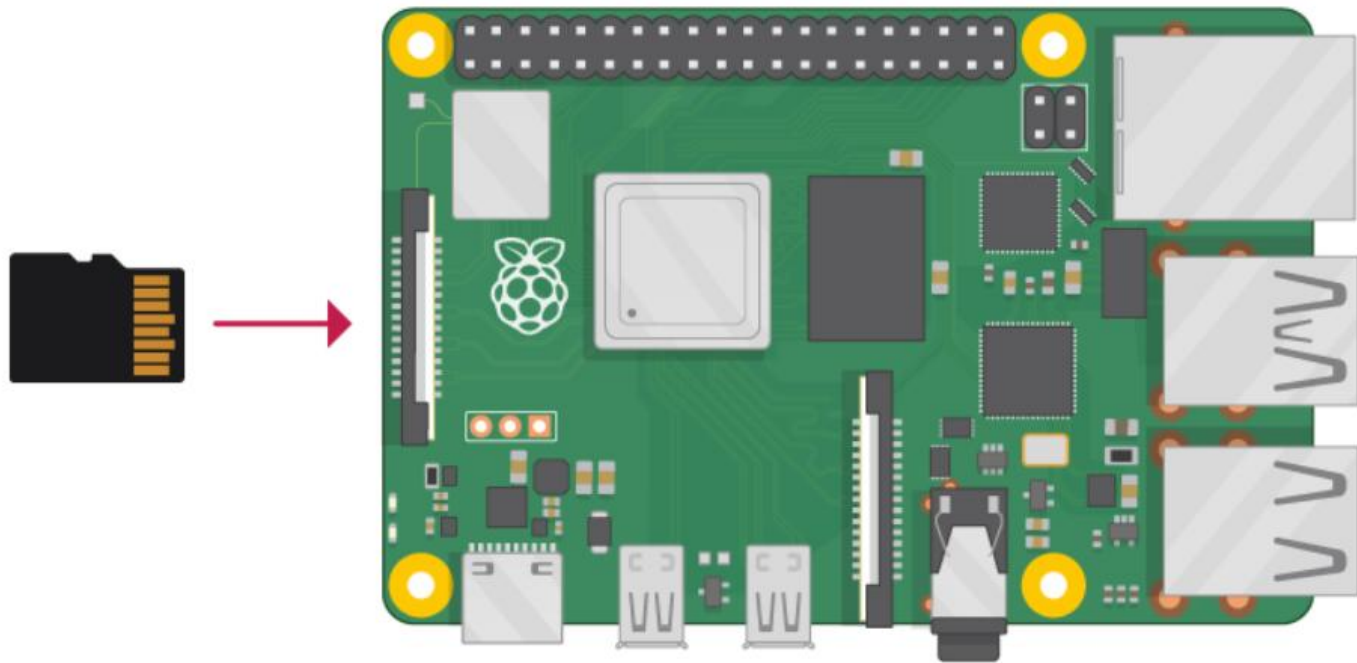
# 開放源碼軟件

- 任何人都可以自由地：
  - 使用該軟件作任何用途
  - 瀏覽程式碼
  - 更改程式碼
  - 發佈更改後的程式碼
- **唯獨不可以**將更改後的程式據為己有並發佈為任何非開源軟件
  - 程式碼必須公開

# 插入 SD Card 及啟動 Raspberry Pi

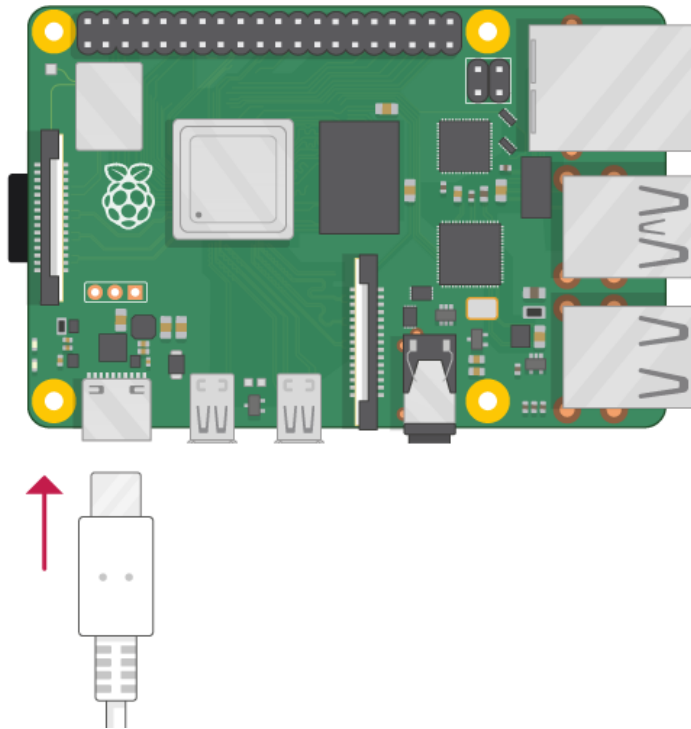


## (1) 將SD卡插入Raspberry Pi



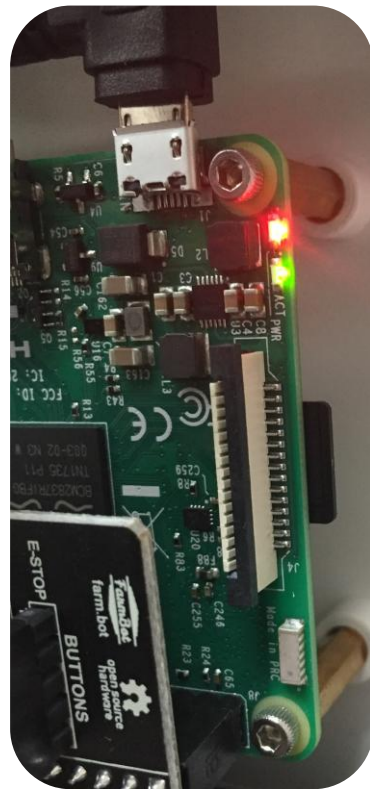


## (2) 將行動電源與Raspberry Pi 連接，並供電給 Raspberry Pi



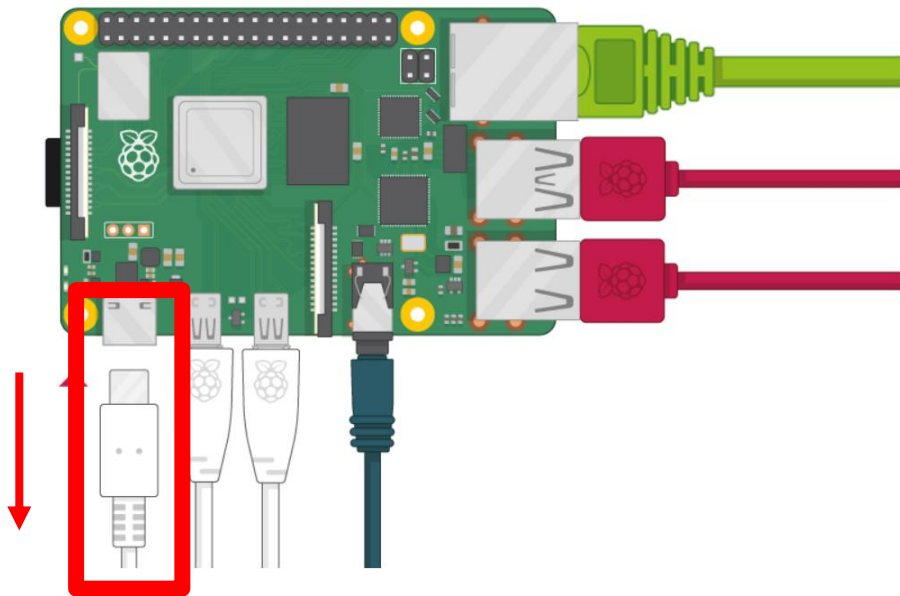
### (3) 留意Raspberry Pi 的訊號燈

- 實色紅燈 → Raspberry Pi 已連接電源
- 綠燈閃爍 → SD卡正在活動



# 關機方法

- 直接拔出電源線



# Raspberry Pi 4



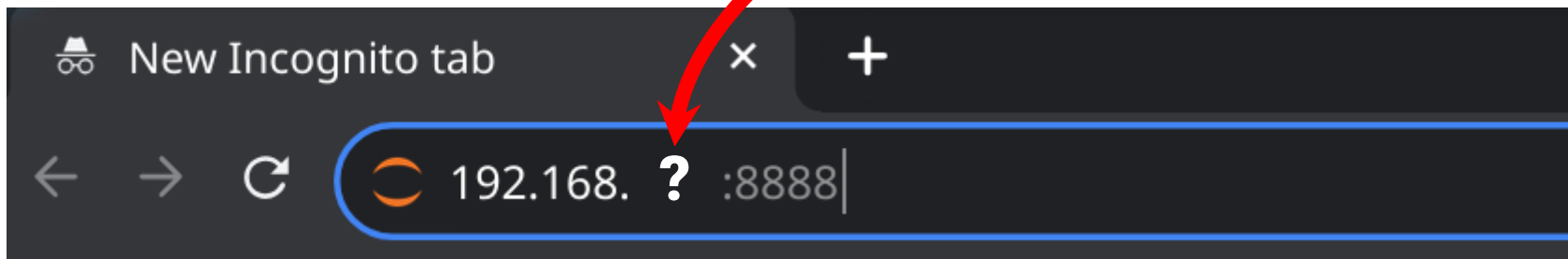
探索 Raspberry Pi 的大腦

# Jupyter Lab 的基本操作



# JupyterLab

打開 Chrome, 在地址列輸入 192.168.xxx.yyy:8888 進入JupyterLab



# 為何不能連接到 Jupyter Lab

- 沒有輸入自己的 IP 地址
- 打錯 IP 地址
- 將 :8888 打成 .8888
- 沒有將 PC 連接到 Raspberry Pi 相同的網路



# 為何不能連接到 Jupyter Lab



為什麼是 :8888 而不是 .8888 ?  
:8888 的意義是什麼 ?  
它是IP的一部份嗎 ?

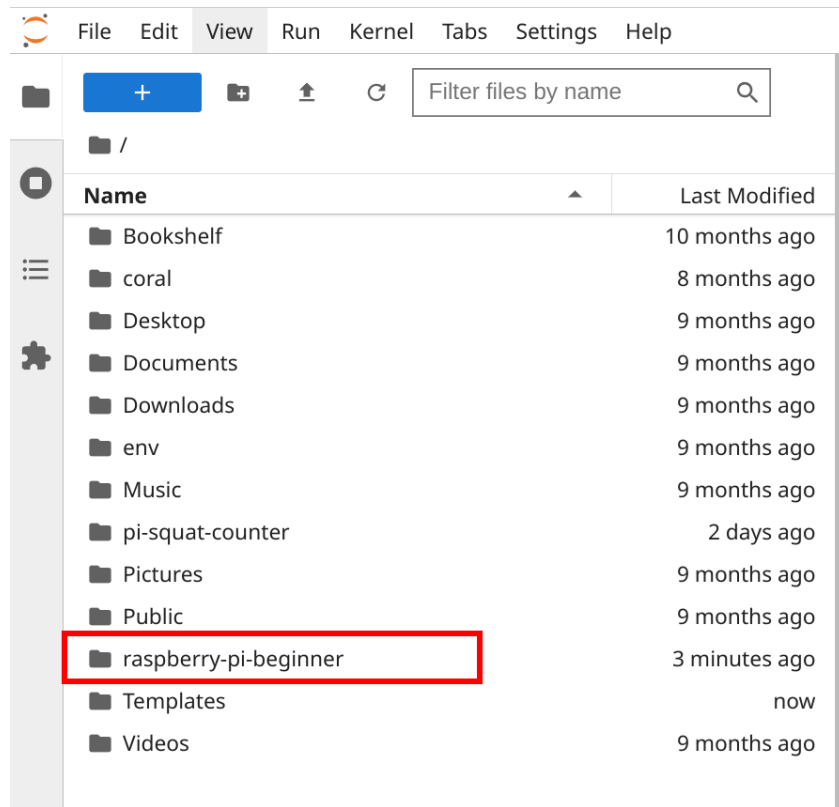
# 通訊埠編號

- 讓電腦能夠區分不同種類的通訊
- 有 65,536 個通訊埠 (0-65535)
  - 這個數字有什麼特別？  $2^{16} = 65,536$
- 當中，有些是有官方指定的功用的，例：
  - 連接埠 21：FTP (File Transfer Protocol)
  - 連接埠 80：HTTP (HyperText Transfer Protocol)

- IP 位址 → 地址第一行 (街道、門牌號碼)      192.168. \_ . \_
- 通訊埠編號 → 地址第二行 (樓層、單位)      :8888

# Jupyter Notebook 基本教學

- 雙擊 “raspberrypi-beginner”



# Jupyter Notebook 基本教學

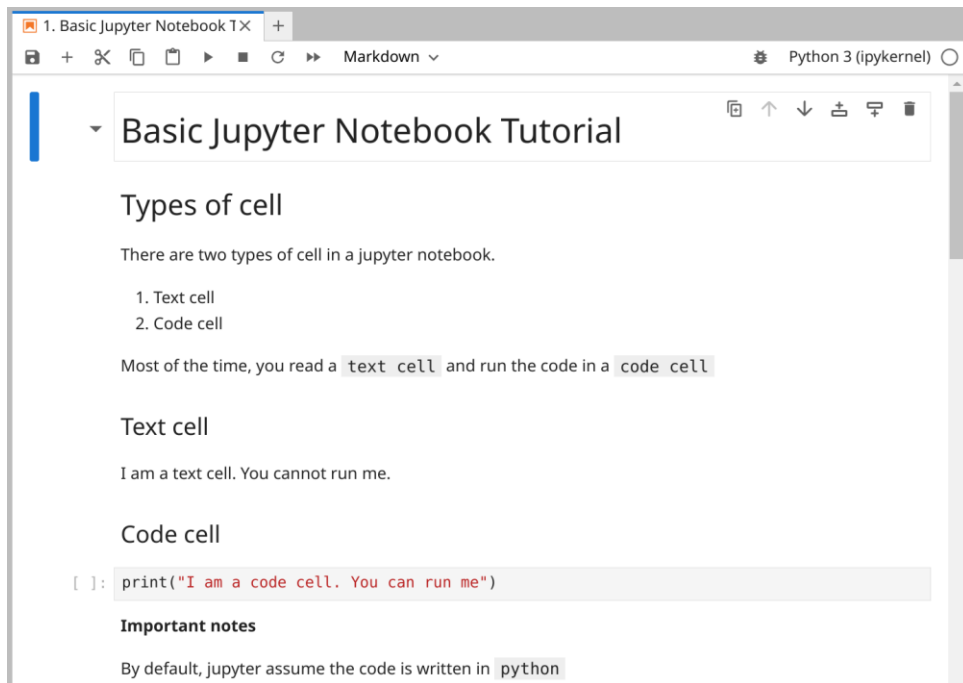
- 打開 “3.1 Basic Jupyter Notebook Tutorial”

📁 / raspberry-pi-beginner /

Name	Last Modified
📁 admin	3 個月前
📁 camera_output	4 個月前
📁 images	3 個月前
📄 3.1 Basic Jupyter Notebook Tutorial.ipynb	4 個月前
• 📄 3.2 Linux Commands.ipynb	6 分鐘前
📄 4.1 Hello World.ipynb	3 個月前
📄 5.1 World Clock.ipynb	3 個月前
📄 6.1 Taking a picture.ipynb	3 個月前
📄 6.2 Tune camera focus.ipynb	30 分鐘前
📄 6.3 Save camera image to disk.ipynb	3 個月前
📄 6.4 Your time lapse camera project.ipynb	3 個月前
📖 README.md	4 個月前

# Jupyter Notebook 基本教學

- 打開後會看到這個介面



# 使用指令關機



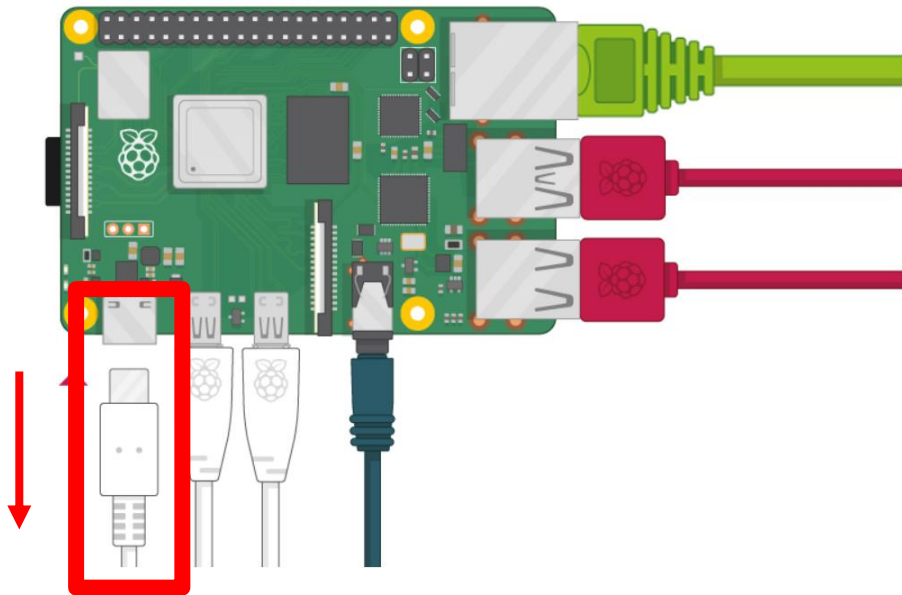
# 使用指令關機

- 你知道如何關機嗎？
  - `sudo poweroff`
  - 關機後 OLED 的時間有更新嗎？
  - 為什麼關機後 OLED 還在顯示資料？
  - 何時才可拔掉電源線？
    - 需等待綠燈停止



# 關機方法

- 直接拔出電源線



# Raspberry Pi 4



使用 Python 編程 Hello World 程式

# 學習重點

- 學習 Python
  - `print()`
  - 認識程式集
- 使用 OLED 顯示屏
  - 在OLED顯示屏上顯示「Hello World」
  - 更改程式碼以顯示您的名字
  - 利用程式顯示屏幕大小
  - 更改文字顯示的位置
  - 清除顯示屏上的內容
  - 將顯示屏填滿白色

# 學習 Python

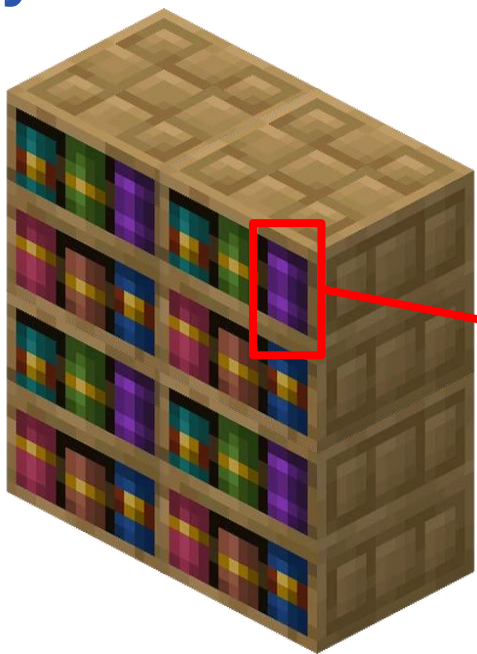


# Jupyter Notebook

- 打開 4.1 Hello World.ipynb 的這個 Jupyter Notebook

/ raspberry-pi-beginner /		
Name		Last Modified
admin		2 小時前
camera_output		4 個月前
images		2 小時前
• 3.1 Basic Jupyter Notebook Tutorial.ipynb		3 小時前
• 3.2 Linux Commands.ipynb		3 小時前
• 4.1 Hello World.ipynb		39 分鐘前
• 4.2 OLED Hello World.ipynb		1 小時前
• 5.1 World Clock.ipynb		3 個月前
• 6.1 Taking a picture.ipynb		3 個月前
• 6.2 Tune camera focus.ipynb		4 小時前
• 6.3 Save camera image to disk.ipynb		1 小時前
• 6.4 Your time lapse camera project.ipynb		3 小時前
▼ README.md		4 個月前

# Library



**random**  
程式集 Library



**randint(a, b)**  
函數 Function

傳回一個 a 和 b 之間 (包含 a 和 b) 的隨機整數

# OLED Hello World





# Raspberry Pi 4

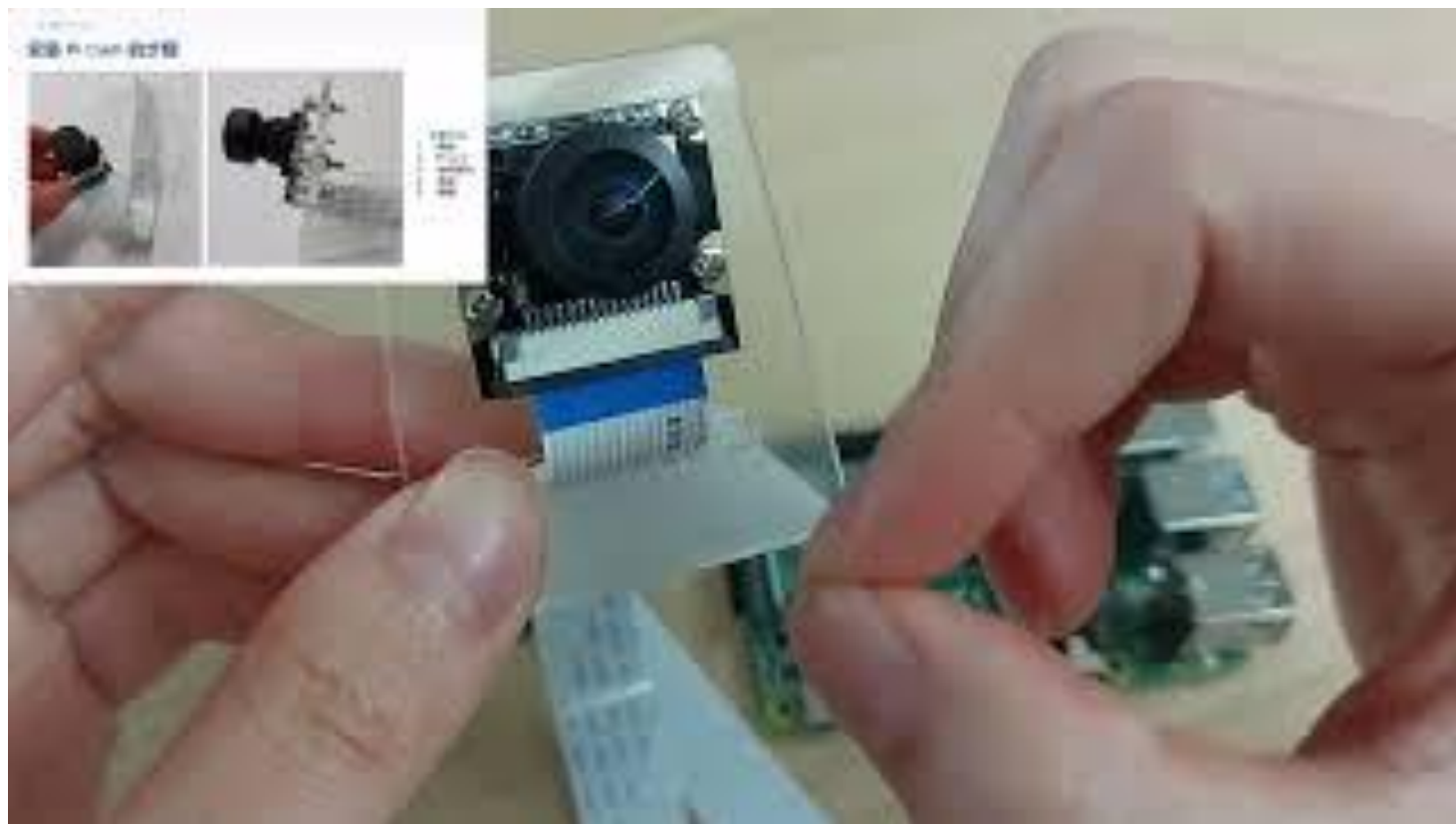


連接網路攝影機及圖像處理

# 安裝 Raspberry Pi 鏡頭



# 教學影片



## 安裝 Pi Cam

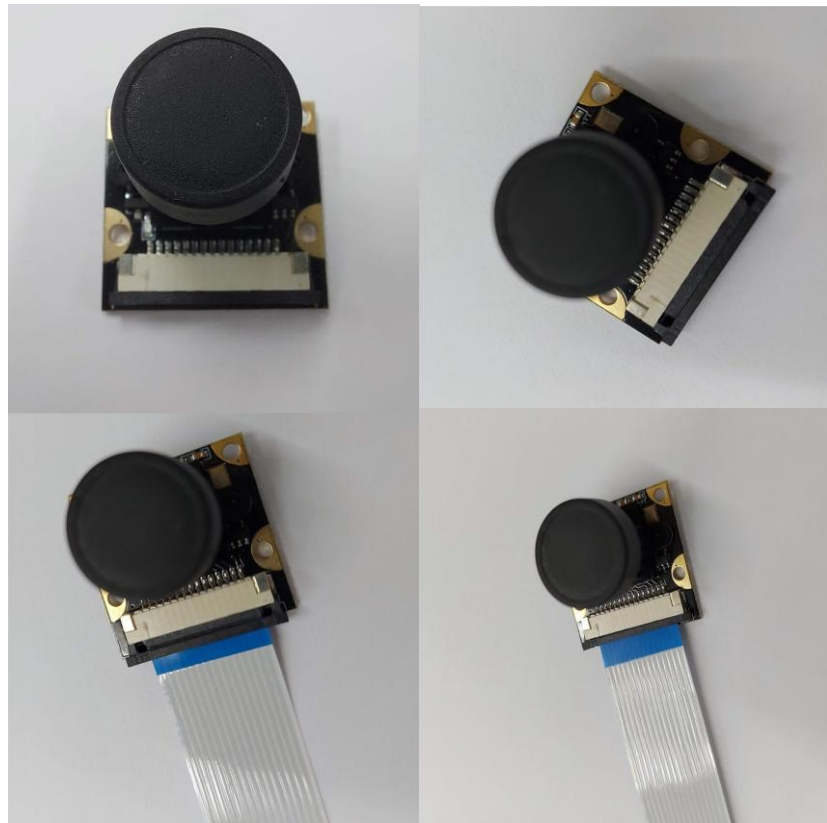


## 安裝 Pi Cam



## 連接數據線

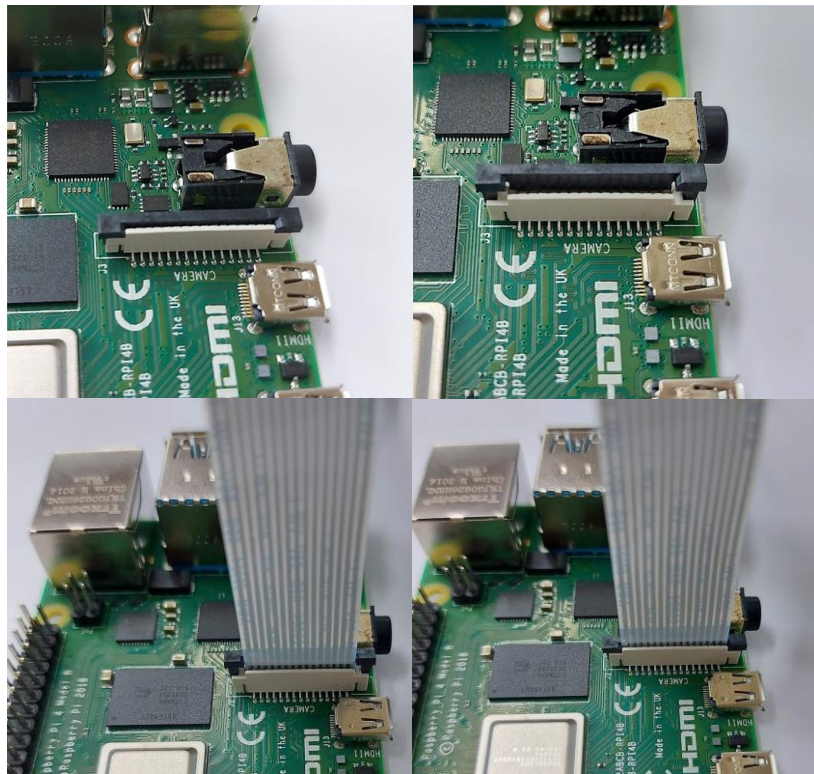
1. 打開鏡頭的卡榫。這個卡榫要兩邊同時施力拉起，注意不要太大力，只能拉起來 1-2 mm
1. 將相機排線插入 (藍色向外)
1. 將卡榫蓋回即可，注意排線要平整的插入，不可以左右不一樣高





# 連接數據線至 Raspberry Pi

1. 打開相機埠的卡榫
1. 將相機排線插入 (藍色向著USB槽)
1. 將卡榫蓋回即可，注意排線要平整的插入，不可以左右不一樣高



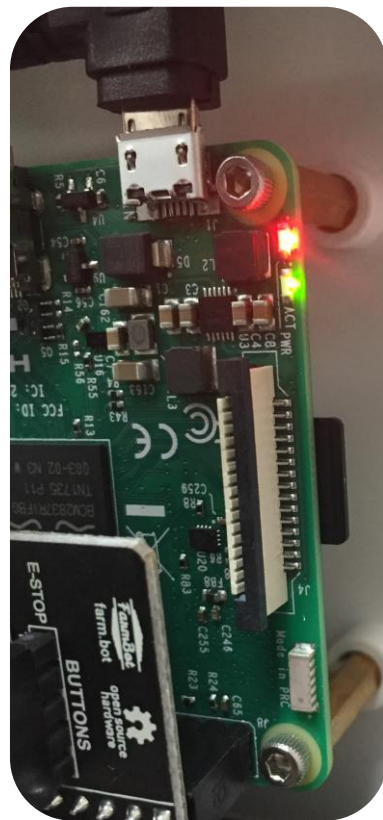


## 安裝鏡頭支架

- 使用雙面貼將鏡頭貼在支架上



將電源線與 Raspberry Pi 連接，並供電給 Raspberry Pi

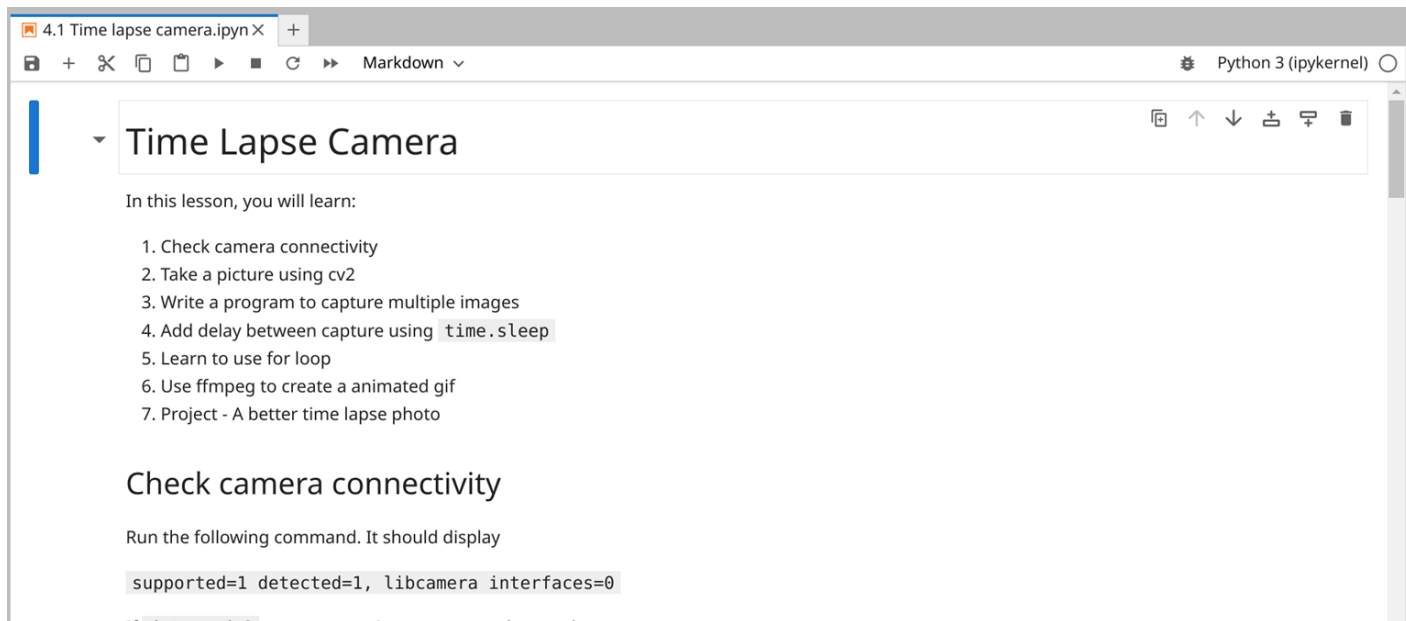


使用程式拍攝一幅圖片



# 拍攝一幅圖片

- 打開 4.1 Taking a picture
  - 試執行練習 1 的程式



# 解釋程式碼

```
: import cv2
  from IPython.display import display, Image

# Start the camera
cap = cv2.VideoCapture(0) ①

# Capture a single frame
ret, frame = cap.read() ②

# Save the captured image
if ret:
    flipped_frame = cv2.flip(frame, 0) ③
    display(Image(data=cv2.imencode('.jpeg', flipped_frame)[1].tobytes())) ④

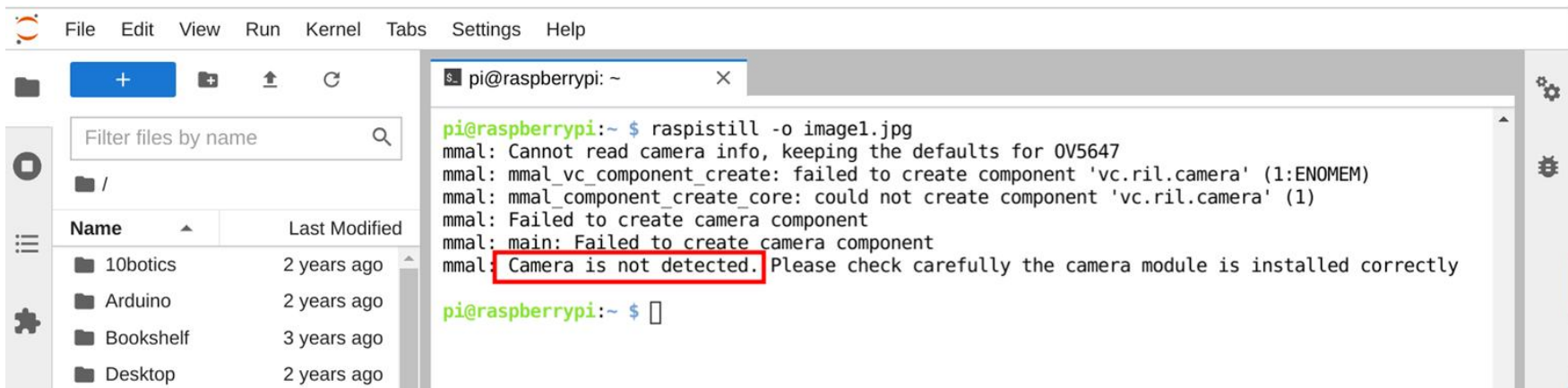
# Release the camera
cap.release()
```

排除錯誤



# 數據線連接錯誤

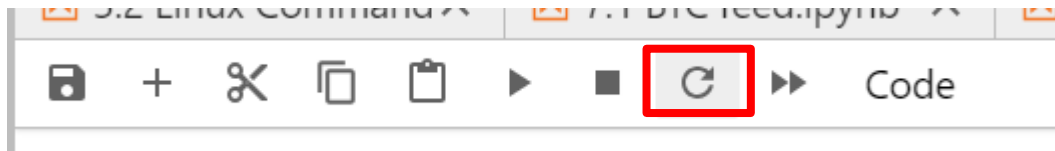
- Camera is not detected
  - 檢查數據線連接是否正確



## 數據線連接錯誤

```
[ WARN:0@31.382] global cap_v4l.cpp:997 open VIDEOIO(V4L2:/dev/video0): can't open camera by index  
[ERROR:0@31.393] global obsensor_uvc_stream_channel.cpp:159 getStreamChannelGroup Camera index out of range
```

- Can't open camera by index
  - 檢查鏡頭是否已連接好
  - 確保沒有其他程序正使用鏡頭
  - 重啟 Jupyter 內核 (Kernel)



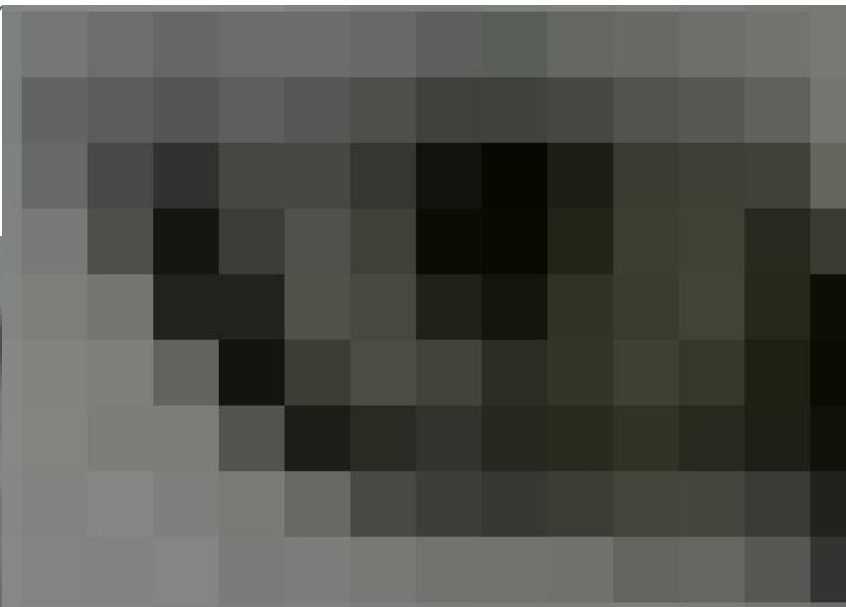
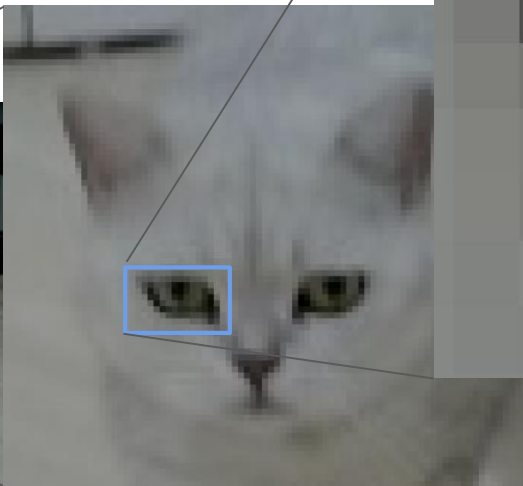
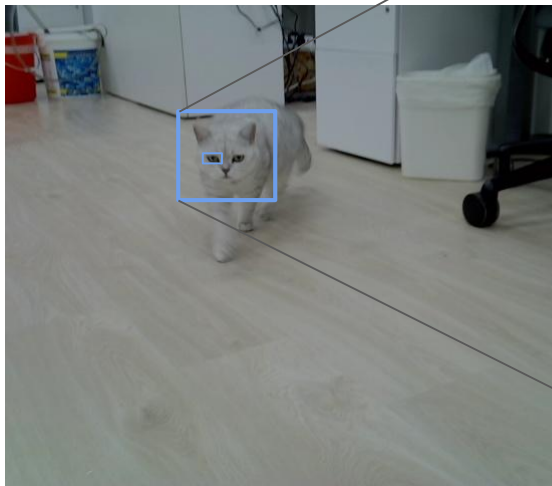


將圖像數碼化



## 電腦是如何儲存和顯示圖像的？

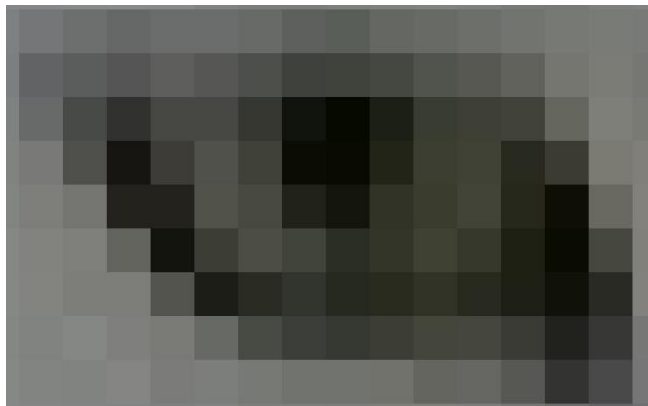
- 讓我們將拍到的照片不斷放大



## 電腦是如何儲存和顯示圖像的？

- 這種一格一格顏色組成的圖像名為**點陣式圖像 Bitmap Image**
  - 相反為向量圖形，以算式表示圖形，例：字體
- 當中一格顏色方塊稱為**像素 Pixel**

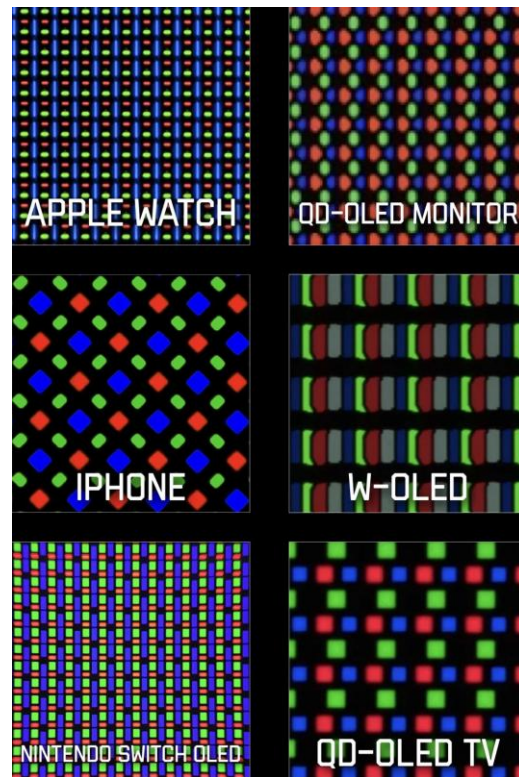
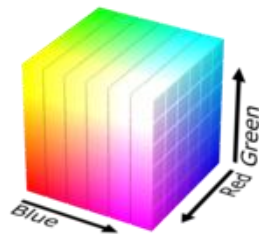
那麼，電腦由是如何儲存每格像素的資訊？



# 電腦是如何儲存和顯示圖像的？

- 你有試過用放大鏡看電子屏幕嗎？
- 以紅綠藍三原色構成不同顏色
  - 簡稱：RGB

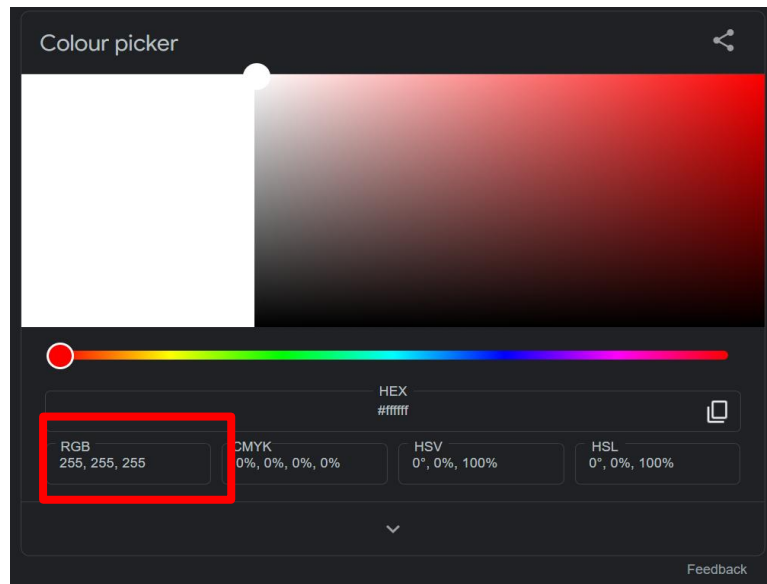
54	17	6	28	57
55	20	9	31	60
49	13	0	22	49
64	11	9	34	60
65	12	11	36	62
57	4	0	23	49
72	33	20	49	58
73	35	22	51	60
65	24	11	38	46



## 電腦是如何儲存和顯示圖像的？

- 標準RGB (又稱全彩) 顯色共可顯示多少顏色？
- 試在 [Google Color Picker](#) 尋找答案
- 提示：留意R G B數值範圍，再計算可行組合數

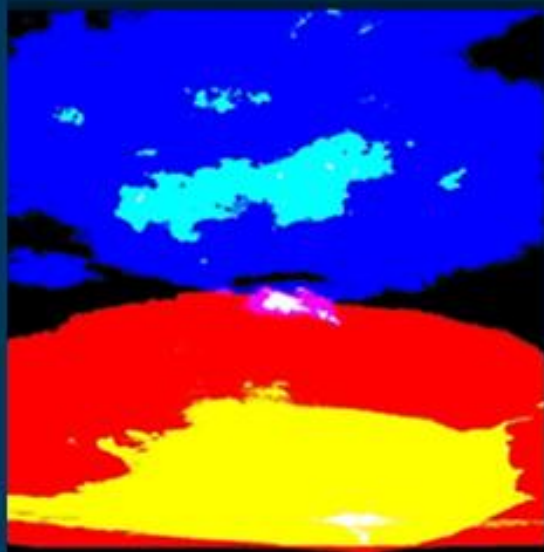
R, G, B 數值各可為 0 - 255，共 256 個組合  
 $256 * 256 * 256$   
 $\approx 16,800,000$  種顏色



# 計算圖片大小

- 首先，一個標準RGB像素儲存的數據有什麼？
  - R (0-255), G (0-255), B (0-255)
  - 一個 0 - 255 的數字佔多少位元？(多少個 1 / 0 才可組成 256 個組合？)
  - 8 位元
  - 一個像素的大小 = 8 位元 \* 3 種顏色 = 24 位元 Bits (3 位元組 Bytes)
- 我們這樣可以得知標準RGB的色深 **Colour Depth** 是 24 位元 Bits
  - 色深 = 表達一隻顏色所有的儲存空間
  - 色深越高，顏色越細緻

3 Bit  
(8 Color)



8 Bit  
(256 Color)



24 Bit  
(16,77,216 Color)



# 深蹲計數器





# 深蹲



深蹲的正確做法？

# HOW TO DO SQUAT

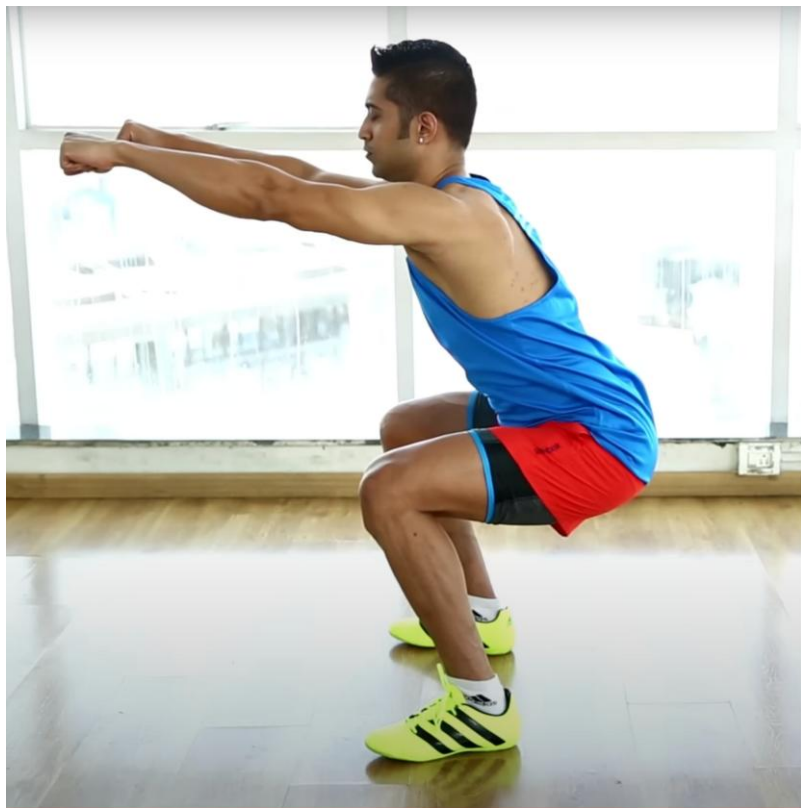


**FITNESS**  
Special

## 練習

- 每兩位同學一組，輪流做 10 下深蹲
  - 負責數數的同學必須大聲喊出數字

# 分析



# 分析

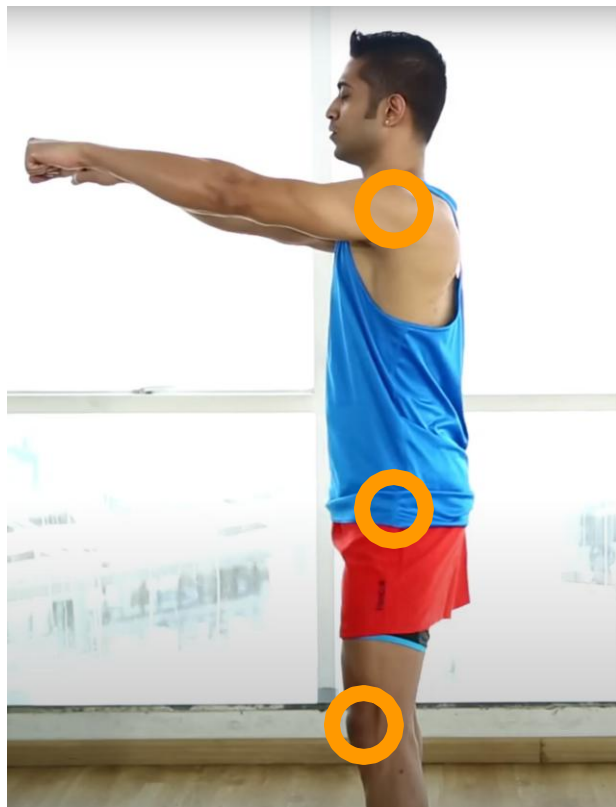


# 設計



如何為之完成一次深蹲？

# 分析

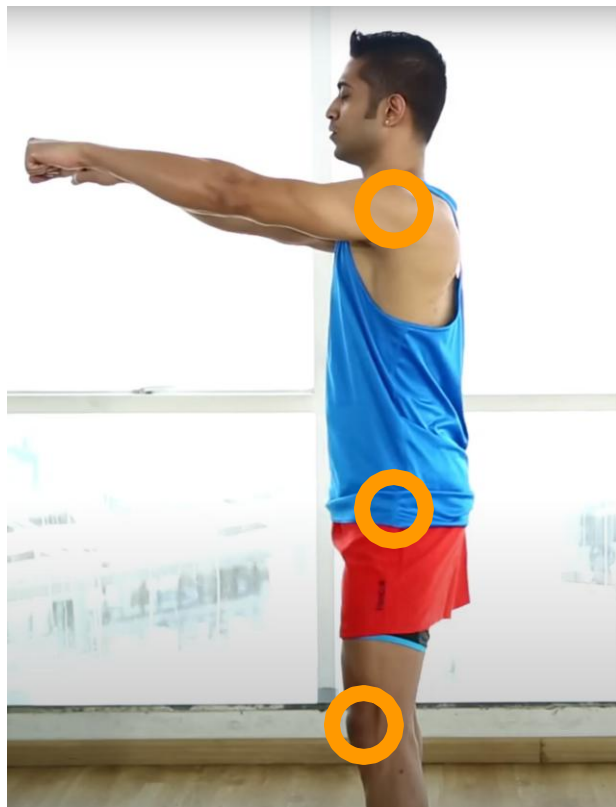


# 分析





# 分析



# 設計



如何設計一個深蹲計數器？

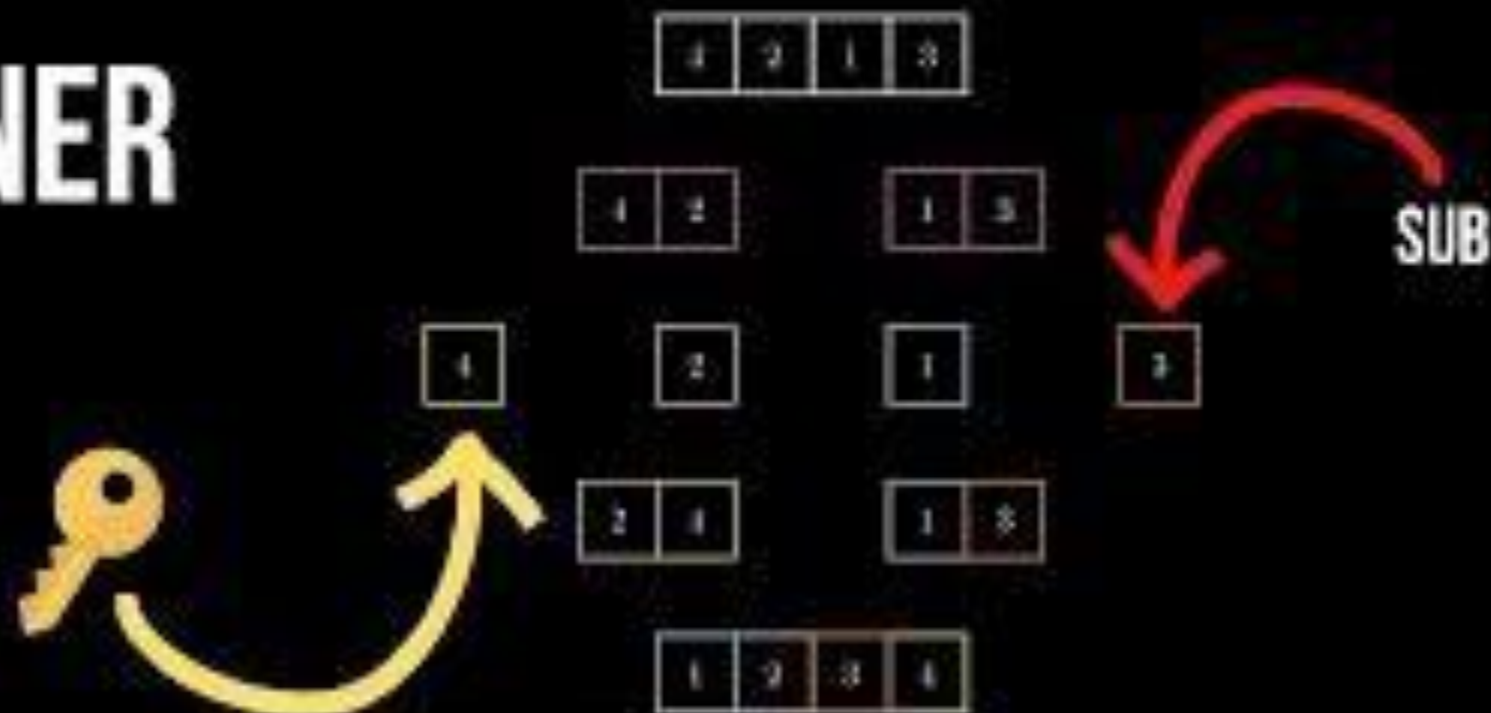
State: s1

CORRECT: 0

INCORRECT: 0



I SHOULD HAVE LEARNED THIS  
SOONER





State: s1

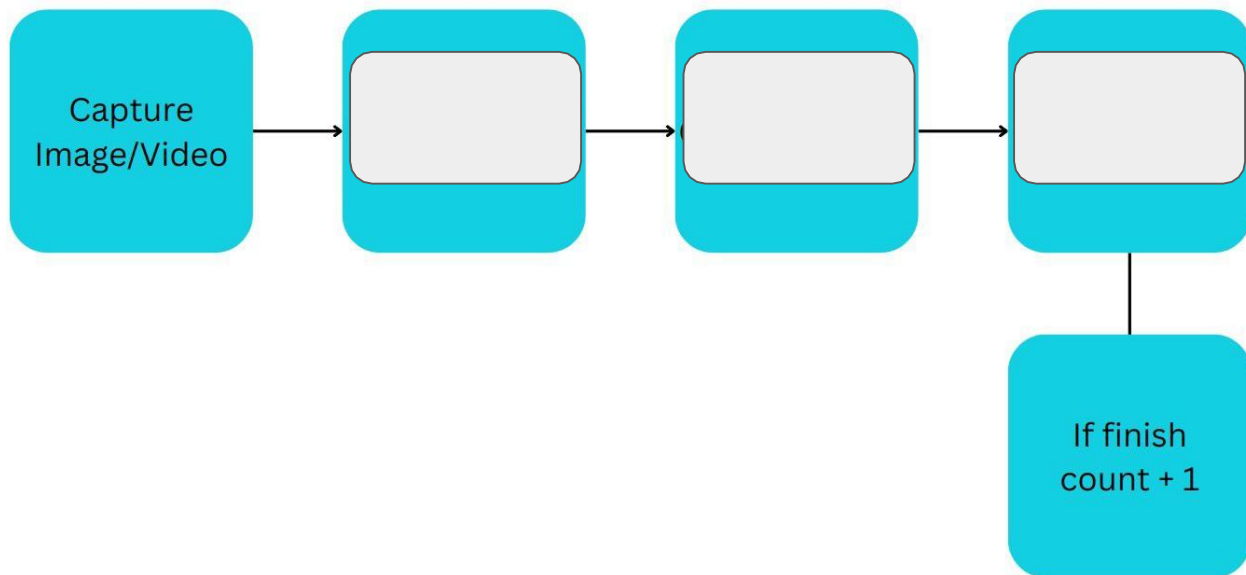
CORRECT: 0

INCORRECT: 0



## 初步設計

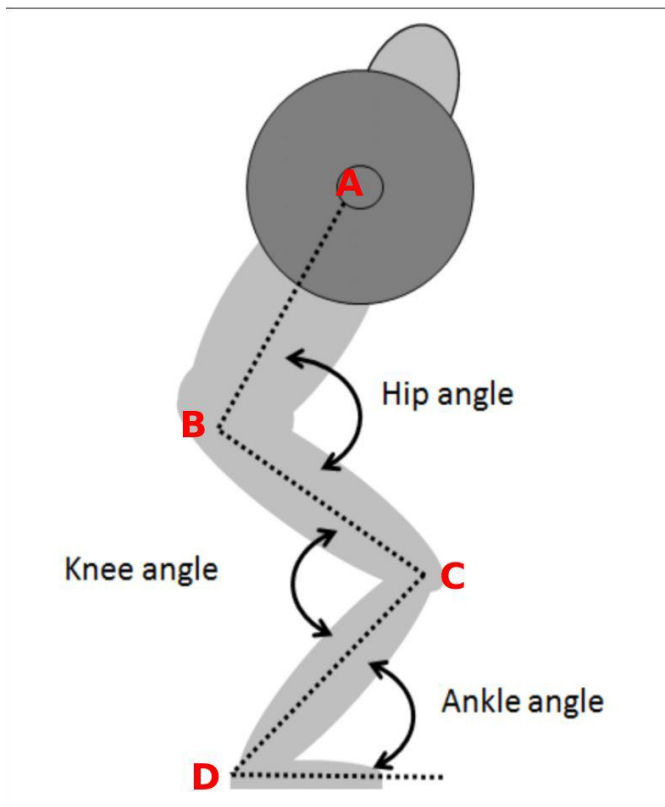
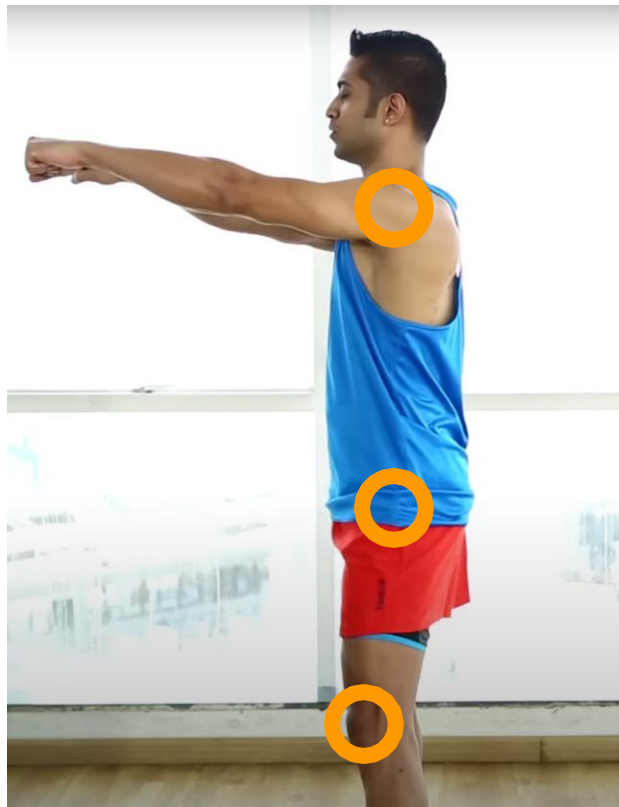
- 3 分鐘思考一下該怎樣設計



# 姿勢偵測模型



# 分析



- $\angle ABC > 150$
- $\angle BCD > 150$



# 什麼是姿勢偵測模型？

- 檢測人體姿勢的計算機視覺技術

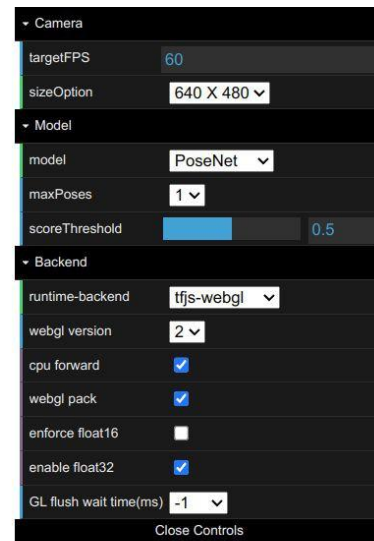


# 有什麼模型可以用？

- [PoseNet](#) / [MoveNet](#)
- MediaPipe BlazePose / Pose Landmark Detection
  - [Introduction on 2020](#)
  - [Project page](#)

# PoseNet / MoveNet

- 五分鐘試玩各個模型 Let's try



# 設計

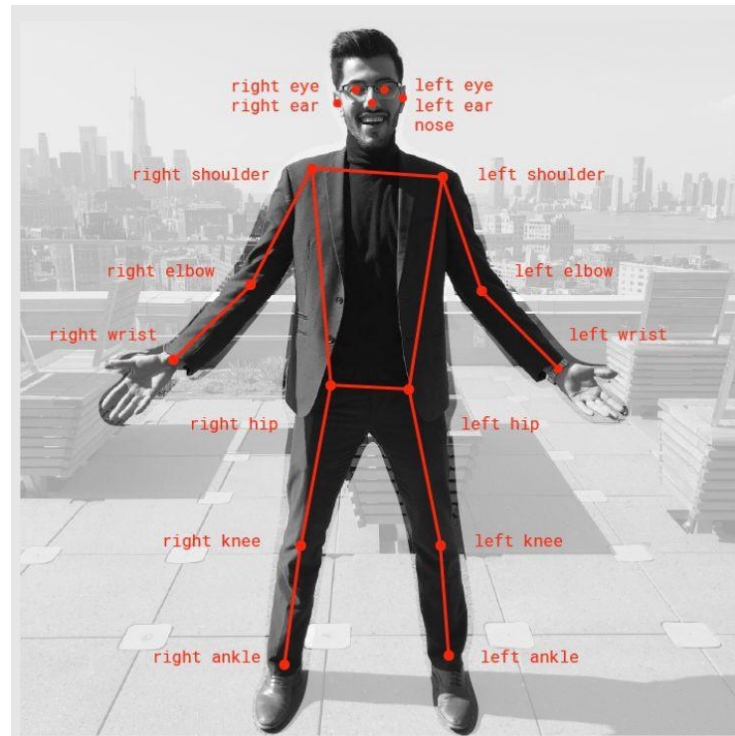


有什麼分別？

# PoseNet / MoveNet

- 17 key points
  - Each keypoint contains x, y, score and name

```
{  
  "score": 0.32371445304906,  
  "keypoints": [  
    { // nose  
      "position": {  
        "x": 301.42237830162,  
        "y": 177.69162777066  
      },  
      "score": 0.99799561500549  
    },  
    ...  
  ]  
}
```



# PoseNet / MoveNet

- PoseNet
  - Previous generation model released in 2017
  - Capable of detecting multiple pose
- MoveNet
  - State-of-the-art pose estimation
  - Capable of detecting single pose only
  - Ultra fast and accurate model
  - [Detailed model card of MoveNet](https://github.com/tensorflow/tfjs-models/tree/master/pose-detection)

# Model Card

- 兩分鐘閱讀一下 Model Card

## MoveNet.SinglePose

### Model Details

A convolutional neural network model that runs on RGB images and predicts [human joint locations](#) of a single person. The model is designed to be run in the browser using [Tensorflow.js](#) or on devices using [TF Lite](#) in [real-time](#), targeting **movement/fitness activities**. Two variants are presented:

- **MoveNet.SinglePose.Lightning**: A lower capacity model that can run >50FPS on most modern laptops while achieving good performance.
- **MoveNet.SinglePose.Thunder**: A higher capacity model that performs better prediction quality while still achieving real-time (>30FPS) speed. Naturally, *thunder will lag behind the lightning, but it will pack more of a punch.*

# PoseNet / MoveNet

- What is mAP?

Model	Size (MB)	mAP	Latency (ms)		
			Pixel 5 - CPU 4 threads	Pixel 5 - GPU	Raspberry Pi 4 - CPU 4 threads
<a href="#">MoveNet.Thunder (FP16 quantized)</a>	12.6MB	72.0	155ms	45ms	594ms
<a href="#">MoveNet.Thunder (INT8 quantized)</a>	7.1MB	68.9	100ms	52ms	251ms
<a href="#">MoveNet.Lightning (FP16 quantized)</a>	4.8MB	63.0	60ms	25ms	186ms
<a href="#">MoveNet.Lightning (INT8 quantized)</a>	2.9MB	57.4	52ms	28ms	95ms
<a href="#">PoseNet(MobileNetV1 backbone, FP32)</a>	13.3MB	45.6	80ms	40ms	338ms



# BlazePose

- 三分鐘試試 BlazePose 模型

- [https://mediapipe-studio.webapps.google.com/demo/pose\\_landmarker](https://mediapipe-studio.webapps.google.com/demo/pose_landmarker)

### Pose Landmark Detection

Identifies key points on a human body in an image or video. This information can be used for a variety of applications, such as human-computer interaction, motion analysis, and virtual reality.

For more information on the model, performance, etc, see the [documentation](#).

The sample parameters below can be changed. See [documentation](#) for more details

Inference delegate: GPU inference

Model selections: Pose Landmarker Lite


Demo num poses: 1 5

Minimum pose detection confidence: 1% 99%

Minimum pose presence confidence: 1% 99%

Minimum tracking confidence: 1% 99%

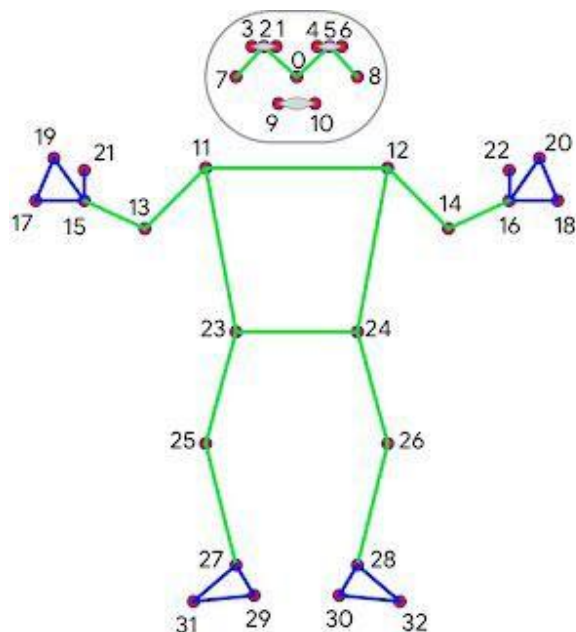
Input Integrated Camera (13d3...)



Inference time (ms): 9.2

# BlazePose

- Returns 33 keypoints



- |                    |                            |
|--------------------|----------------------------|
| 0. nose            | 17. right pinky knuckle #1 |
| 1. right eye inner | 18. left pinky knuckle #1  |
| 2. right eye       | 19. right index knuckle #1 |
| 3. right eye outer | 20. left index knuckle #1  |
| 4. left eye inner  | 21. right thumb knuckle #2 |
| 5. left eye        | 22. left thumb knuckle #2  |
| 6. left eye outer  | 23. right hip              |
| 7. right ear       | 24. left hip               |
| 8. left ear        | 25. right knee             |
| 9. mouth right     | 26. left knee              |
| 10. mouth left     | 27. right ankle            |
| 11. right shoulder | 28. left ankle             |
| 12. left shoulder  | 29. right heel             |
| 13. right elbow    | 30. left heel              |
| 14. left elbow     | 31. right foot index       |
| 15. right wrist    | 32. left foot index        |
| 16. left wrist     |                            |

# BlazePose

- 3D pose estimation



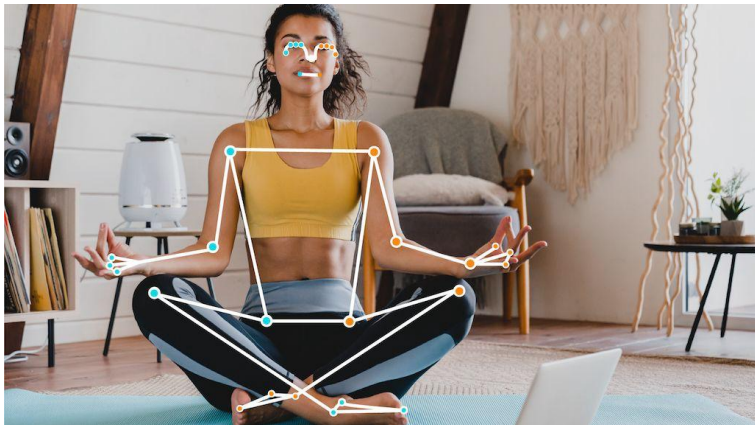
# BlazePose

- keypoints3D

```
[
  {
    score: 0.8,
    keypoints: [
      {x: 230, y: 220, score: 0.9, name: "nose"},
      {x: 212, y: 190, score: 0.8, name: "left_eye"},
      ...
    ],
    keypoints3D: [
      {x: 0.5, y: 0.9, z: 0.06 score: 0.9, name: "nose"},
      ...
    ]
  }
]
```

# BlazePose

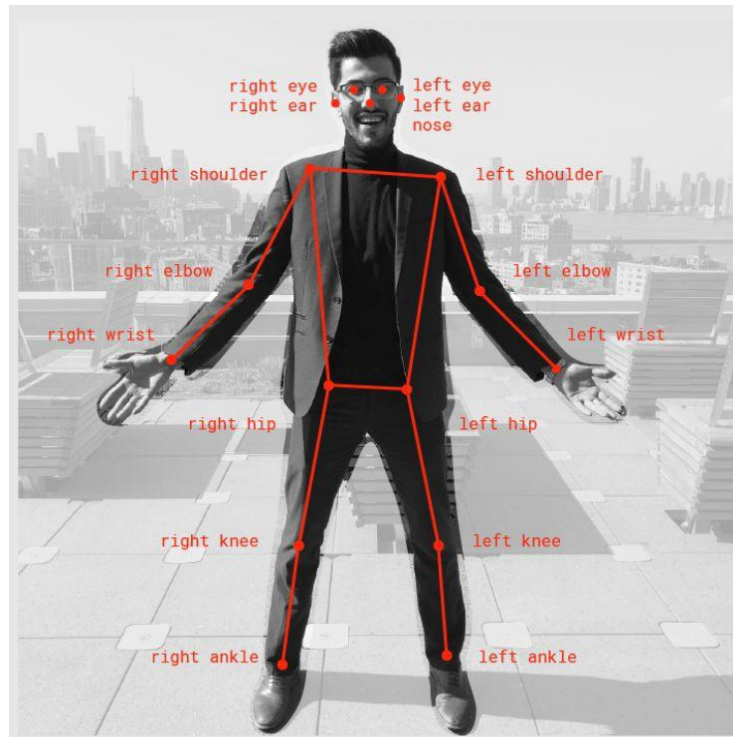
- Segmentation Mask



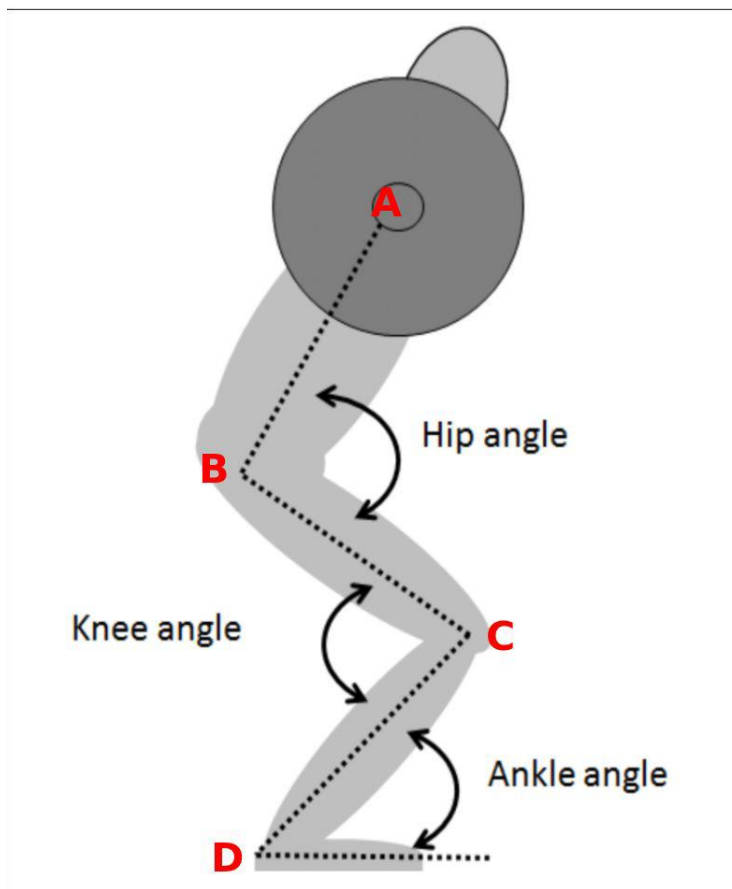
# 計算角度



# 要計算哪些角度？



# 分析



- $\angle ABC > 150$
- $\angle BCD > 150$



# Reference

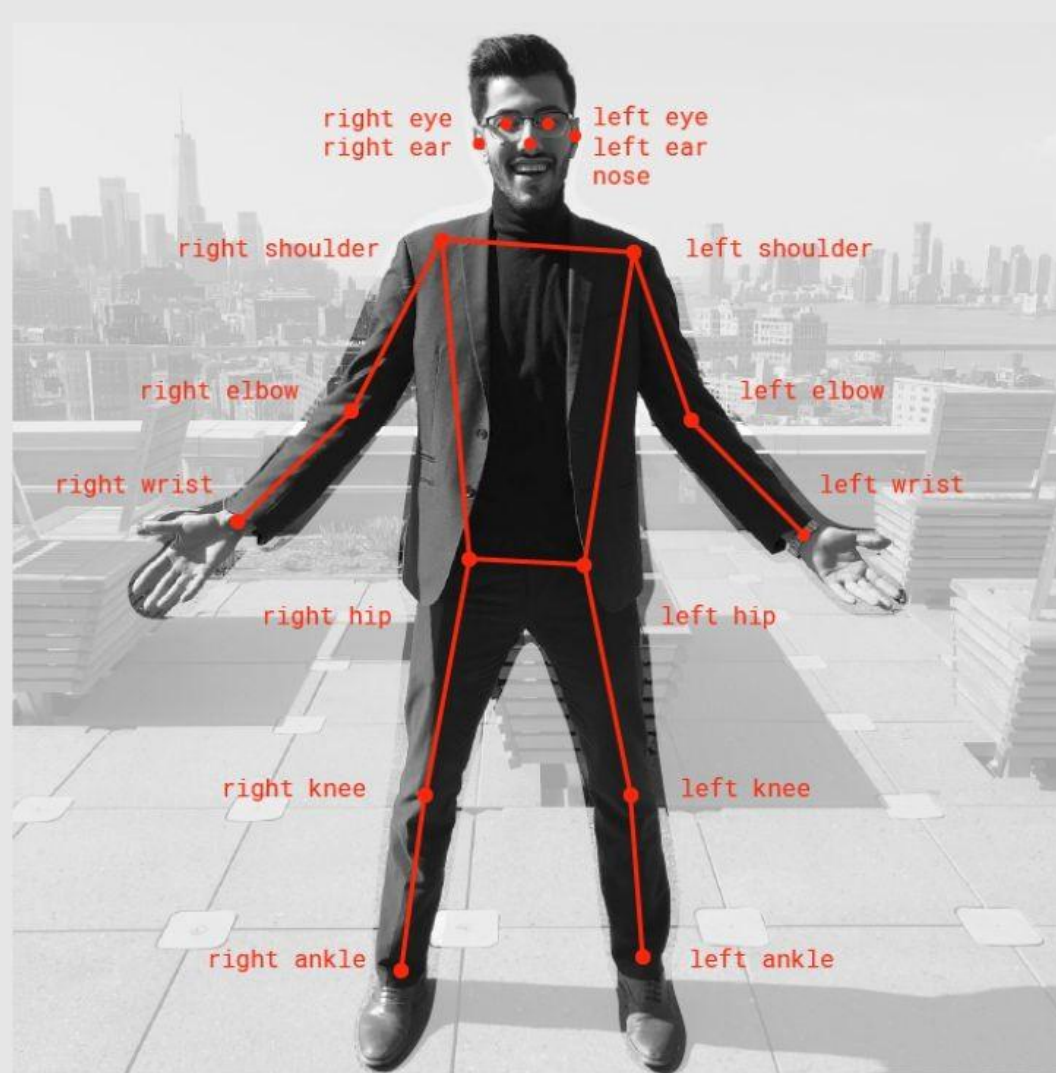
- <https://learnopencv.com/ai-fitness-trainer-using-mediapipe/>



如何在圖片標注圓點？

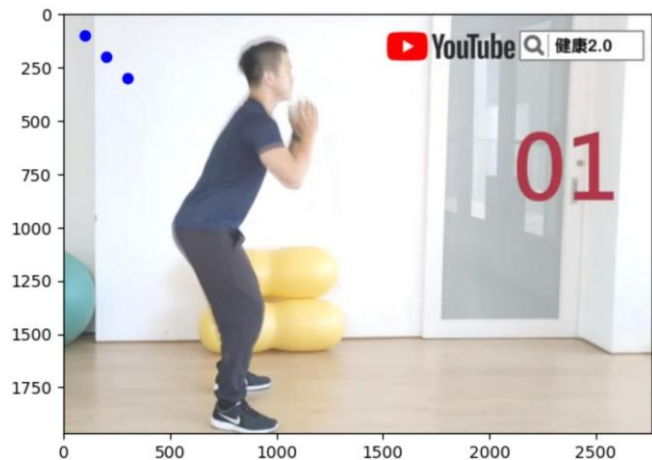


# 例子



# 程式碼

```
def draw_circle():  
    points = [(100, 100), (200, 200), (300, 300)] ① Examl  
  
    x, y = zip(*points) ② npack points into separate list  
    plt.plot(x, y, 'bo') ③
```



# 練習#1

- 將藍點改為紅點
- 參考文件:
  - [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.scatter.html](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.scatter.html)



## 答案

```
def draw_circle():  
    points = [(100, 100), (200, 200), (300, 300)]  
  
    x, y = zip(*points)  
    plt.plot(x, y, 'ro')  
  
# Load and show the image  
image_path = "resource/input_image2.jpeg"  
  
load_and_show_image(image_path)
```

# 圖像大小





## 圖像大小



如何得知圖像的大小?

## 回顧：我們如何讀取圖片？

```
img = mpimg.imread("resource/input_image2.jpeg" )
```

## 回顧：我們如何讀取圖片？



imread matplotlib

All

Images

Videos

Shopping

News

: More

回顧：我們如何讀取圖片？

# matplotlib.pyplot.imread #

`matplotlib.pyplot.imread(fname, format=None)`

Read an image from a file into an array.

## 回顧：我們如何讀取圖片？

### Returns:

`numpy.array`

The image data. The returned array has shape

- (M, N) for grayscale images.
- (M, N, 3) for RGB images.
- (M, N, 4) for RGBA images.

PNG images are returned as float arrays (0-1). All other formats are returned as int arrays, with a bit depth determined by the file's contents.

## 回顧：我們如何讀取圖片？

### Examples

---

```
>>> np.shape(np.eye(3))  
(3, 3)  
>>> np.shape([[1, 3]])  
(1, 2)  
>>> np.shape([0])  
(1, )  
>>> np.shape(0)  
( )
```

---

## 回顧：我們如何讀取圖片？

- 執行右邊的程式碼

```
img = mpimg.imread("resource/input_image2.jpeg" )  
img.shape
```

(1967, 2762, 3)



## 回顧：我們如何讀取圖片？

- 3 是什麼意思？

```
height, width, _ = img.shape  
print(f"width={width}, height={height}, {_}", )  
width=2762, height=1967, 3
```

