

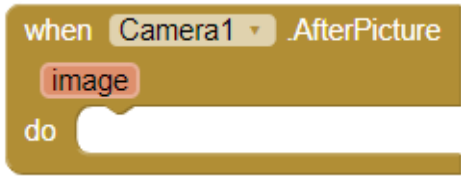
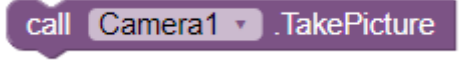
其他元件 – Camera 照相機介紹

作者：蕭志翔

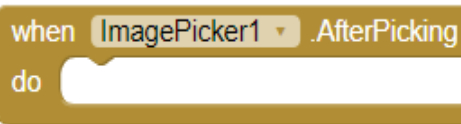
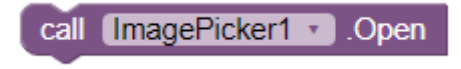
本章介紹

本章主要介紹如何使用 Camera 照相機元件，進行照片拍攝功能，讓使用者可以透過本章的教學範例，快速掌握 Camera 照相機元件，並結合各式指令，設計一個專屬自己 App。

Camera 元件常用介紹：

事件	功能
	當拍照完成後呼叫此事件，並透過字串參數找到照片儲存位置，呼叫它來使用此圖片。
方法	功能
	啟動 Android 裝置的照相機進行拍照。

ImagePicker 元件常用介紹：

事件	功能
	當點選 ImagePicker 中的某項目完成後，呼叫此事件。
方法	功能
	用來呼叫 ImagePicker。

本章除了 Camera 照相機元件之外，還有使用一些 ImagePicker 圖片選擇器元件的指令，在下面的範例中會簡單的做說明及使用，而關於 ImagePicker 圖片選擇器元件會在之後的單元做更詳細的介紹。

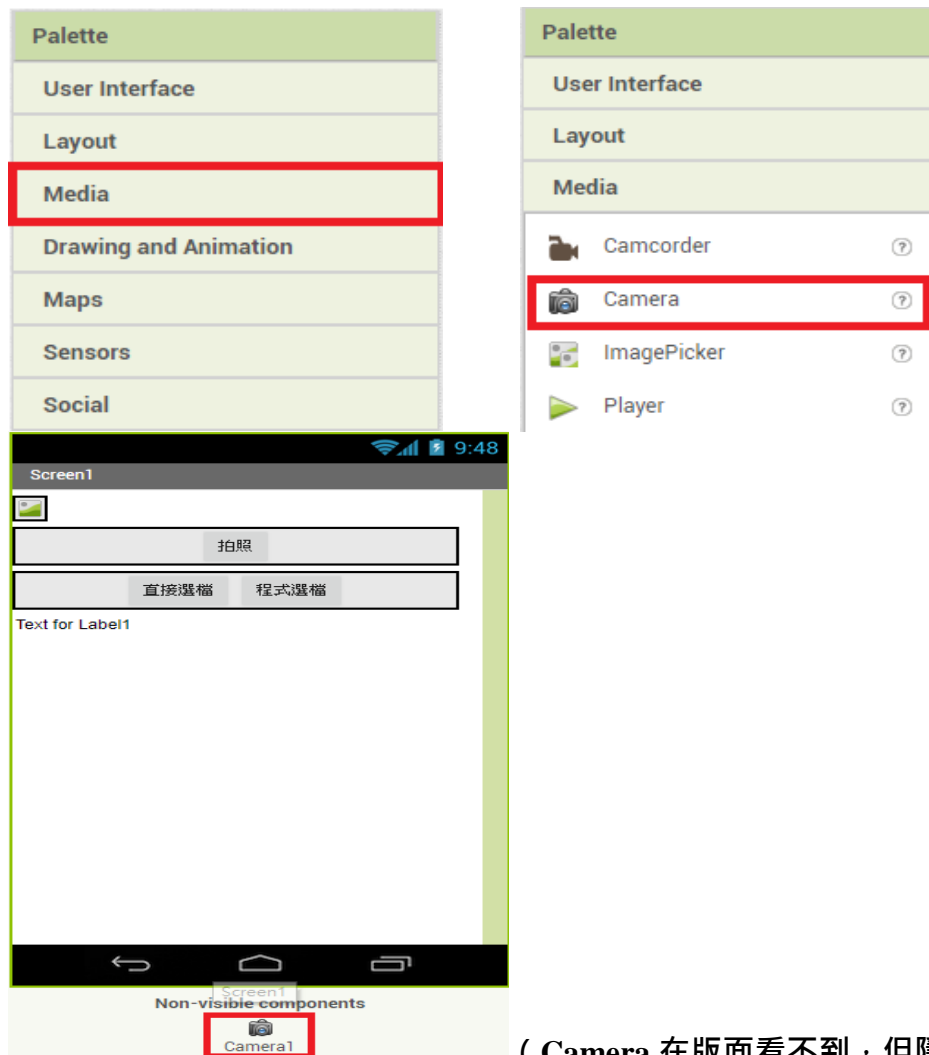
Camera 元件介紹

本章 Camera 元件指令包括 **when Camera.AfterPicture do** 及 **call Camera** 兩種，而下面讓我們用文字與圖片來認識它們。



when Camera.AfterPicture do 主要用於拍照完成後，透過字串參數來呼叫所拍下的照片，而字串參數主要用來儲存照片剛拍攝後存在裝置上所存放的位置。
call Camera 則是主要用於呼叫照相機，讓使用者可以進入拍照模式。

而不管是 **when Camera.AfterPicture do** 還是 **call Camera**，你都必須要新增 Camera 元件才能開始使用他們，而 Camera 元件的位置在版面配置元件的 Media (多媒體) 區。Camera 元件新增後並不會直接顯示在手機螢幕上，而是會隱藏在最下方，所以在新增後要注意是否新增到元件。



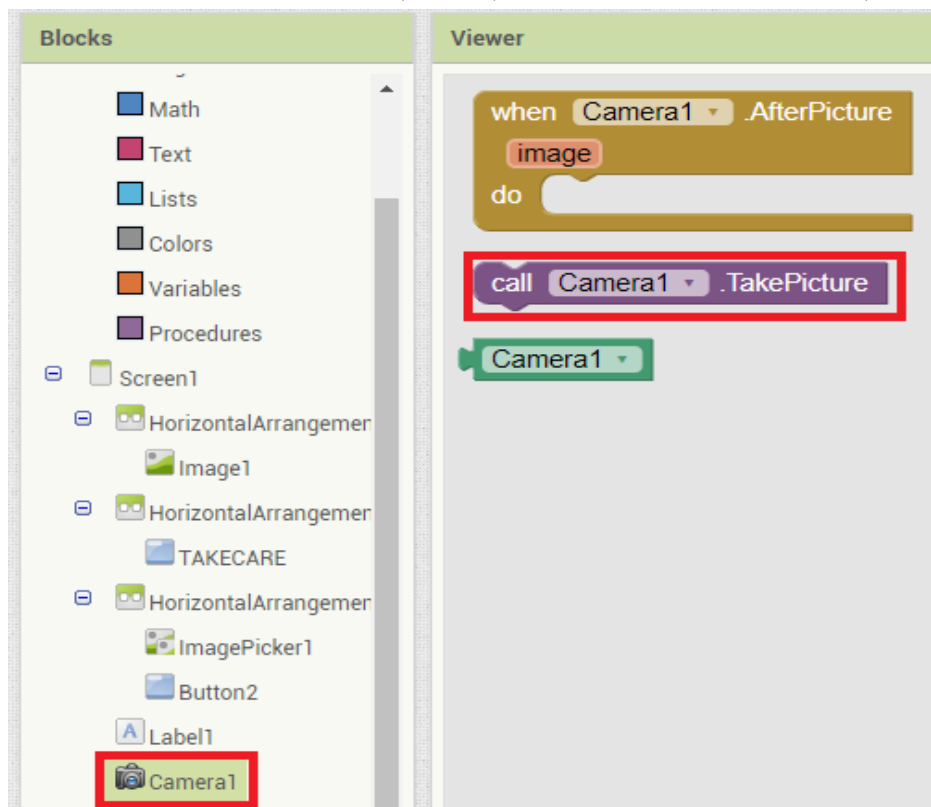
(Camera 在版面看不到，但隱藏最下方)

呼叫 Camera 元件方塊

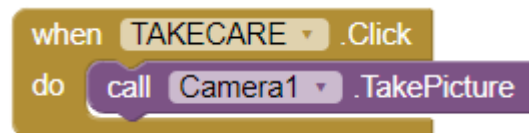
本章的重點在於照相機的應用，所以接下來要介紹如何在手機 App 上開起照相機功能進行拍攝。要讓手機 App 開起照相機功能，必須使用 **call Camera** 方塊才能開啟相機，**call Camera** 方塊主要的功能是用來呼叫手機的照相功能，所以我們可以透過此方塊與 **Button** 按鈕元件的結合，就能做出只要按下按鈕就會開啟拍照功能的程式。

call Camera1 .TakePicture

call Camera 方塊在內建方塊(Blocks)項目中，點選照相機元件(Camera)。



而你能透過與 **Button** 按鈕元件結合，做出觸發拍照模式的按鈕，讓你可以應用在 app 上，方便的開啟拍照模式。

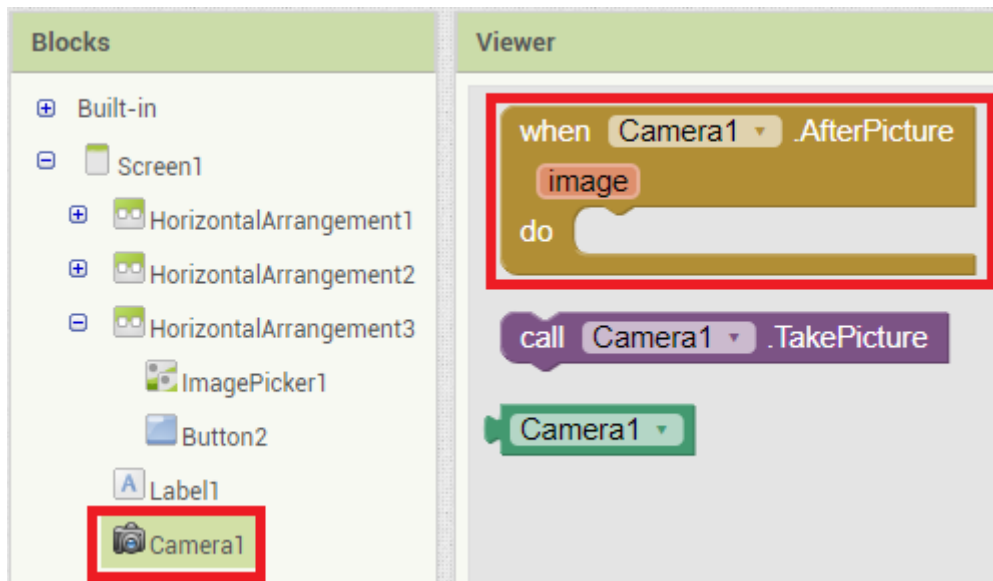


(觸發拍照模式的按鈕)

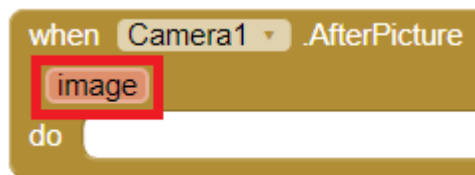
接下來下面會介紹在拍照完成後，所會觸發的程式事件 **Camera.AfterPicture**，而我們可以利用 **Camera.AfterPicture** 來製作完成拍攝後的圖片顯示，或者讓使用者決定要不要重新拍照的畫面及文字顯示。

Camera.AfterPicture 方塊設定

Camera.AfterPicture 方塊要在新增完 Camera 照相機元件後才會出現，新增完後你能在 Screen1 中點選 Camera 找到 Camera.AfterPicture。

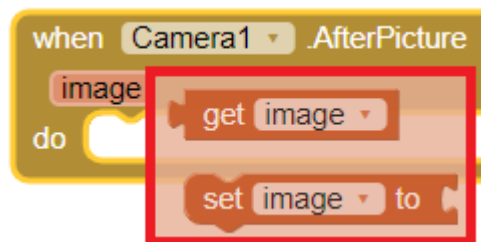


方塊中的 image 字串為一個參數，此參數主要是用來儲存照片剛剛拍攝完後，照片所存放的儲存位置，利用此參數就能夠將剛所拍攝的照片呼叫出來。

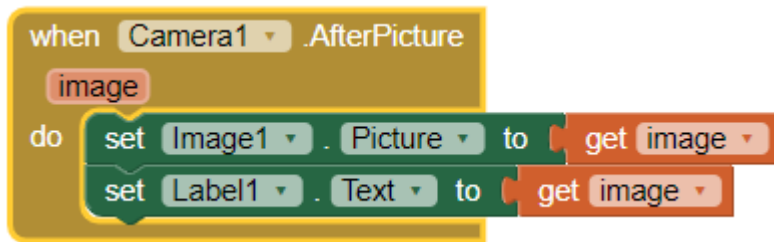


(image 字串參數)

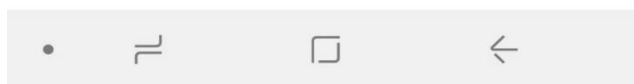
當你點選 image 字串參數後，會跑出 **get image** 和 **set image to** 兩個方塊可以選取新增。**get image** 主要是用來取得 image 字串參數，也就是取得照片位置，藉此呼叫照片顯示出來。而 **set image to** 主要用來設定 image 字串參數，你能透過它將參數內的值進行設定。



(點選 image 字串參數後)



如上圖你可以將 Image 與 Label 都設定為 **get image**，這樣 Image 與 Label 就會將 image 字串參數裡的儲存位子給呼叫出來。



(上方顯示圖片 下方顯示路徑)

由於元件的不同，顯示出來的結果就不同。Image 元件用來顯示圖片，所以它在取得 image 字串參數後會顯示出路徑中的照片，而 Label 用來顯示文字所以取得 image 字串參數後，則是顯示參數內儲存照片位置的路徑。

範例 - 照片拍攝及儲存

請跟著範例做出一個能開啟拍攝模式的按鈕(Camera & Button)並在拍攝完成後能顯示照片(Image) , 且顯示照片當前路徑(Label)的 App 。

步驟一.

- 首先我們先將版面設置好 , 將上面所需要的物件拉至版面排版 , 而版面的配置依個人喜好編排即可 。

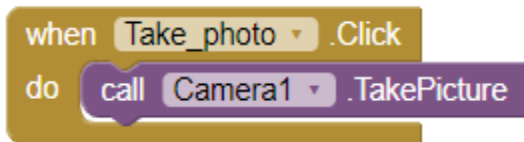


步驟二.

- 先做出能開啟拍照模式的按鈕 Take_Photo 。

利用 **Button.Click** 結合 **call Camera** 製作出能開啟拍照模式的按鈕 , 當我們按下按鈕時就能開啟拍攝模式 。

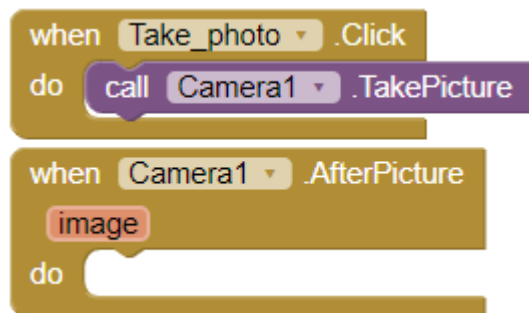
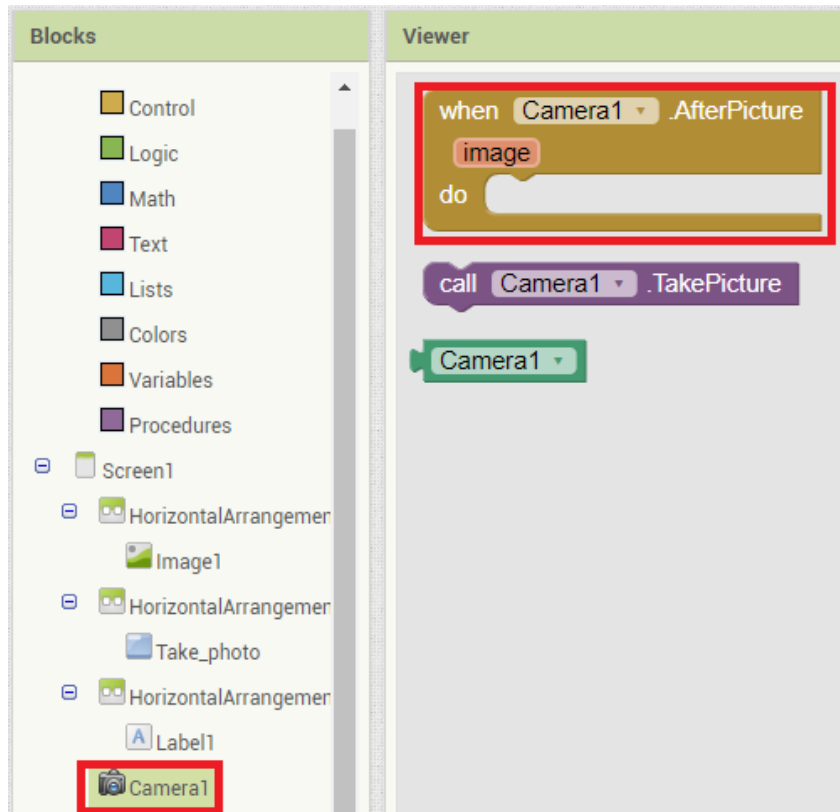
when Take_photo .Click
do



(觸發拍攝模式的 Take_Photo 按鈕)

步驟三.

- 我們要設定在拍攝完成後，顯示拍攝完成的照片及顯示照片路徑，首先在內建方塊(Blocks)項目中，點開 Camera 並新增 **Camera.AfterPicture** 方塊。



(新增 **Camera.AfterPicture** 方塊)

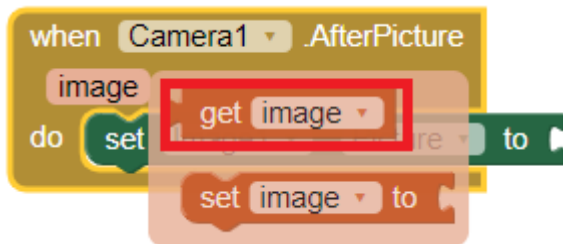
步驟四.

- 在 **Camera.AfterPicture** 方塊中，設定顯示拍攝圖片及顯示照片路徑，也就是將 **set Image.Picture to** 及 **set Label.Text to** 與 **Camera.AfterPicture** 組合起來。在內建方塊(Blocks)項目中，點開 Image 並新增 **set Image.Picture to** 方塊。



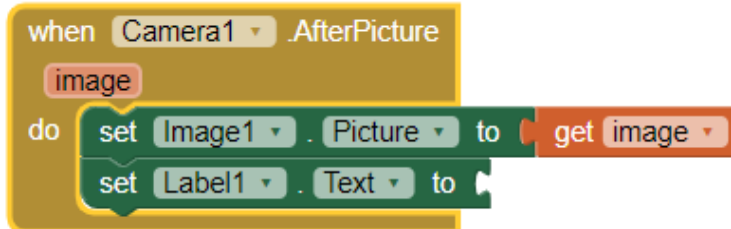
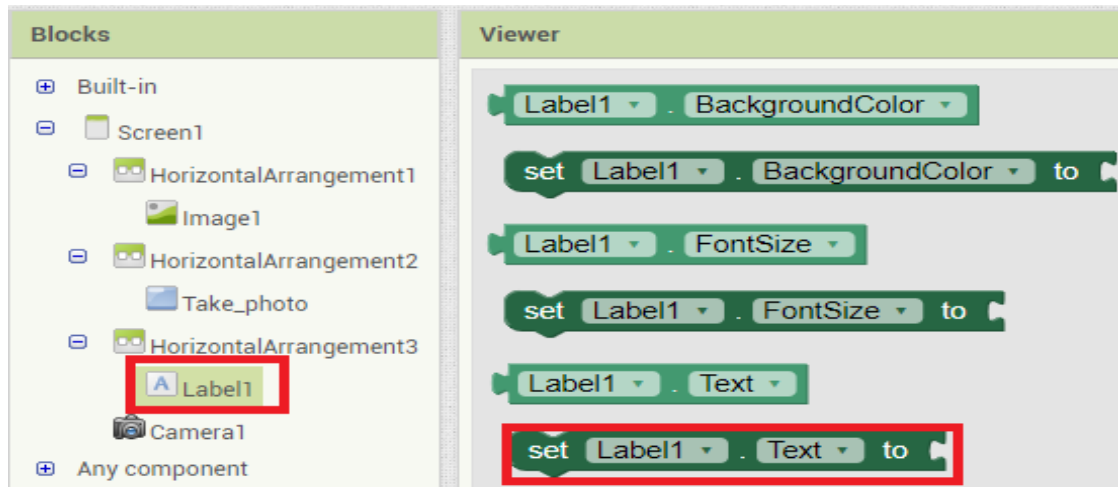
步驟五.

- 在 set Image.Picture to 方塊後新增 Camera.AfterPicture 的 get image 參數，讓拍攝完成後，可以觸發顯示照片的事件。



步驟六.

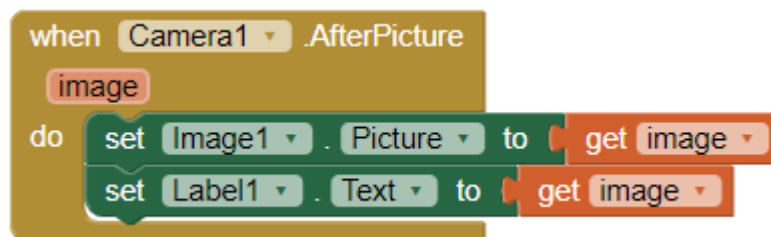
- 接下來用一樣的方式設定顯示路徑的方塊，點選內建方塊(Blocks)，從 Label 中新增 **set Label.Text to** 方塊。



(新增 set Label.Text to)

步驟七.

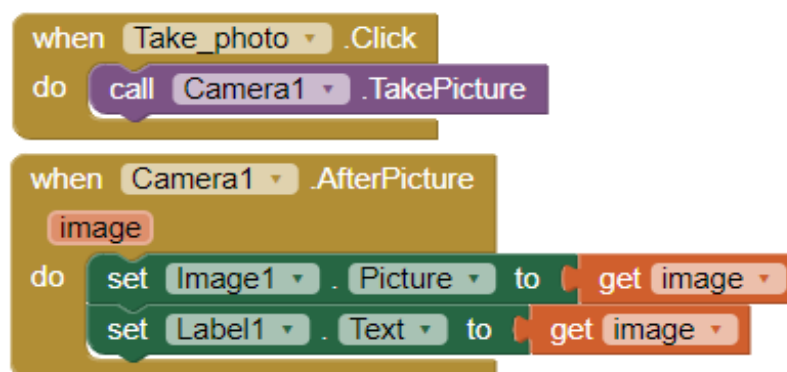
- 在 **set Label.Text to** 方塊後新增 **get image** 方塊，讓我們可以取得 **image** 字串參數的路徑，並顯示在 Label 上。



(get image 字串參數)

步驟八.

- 完整程式碼



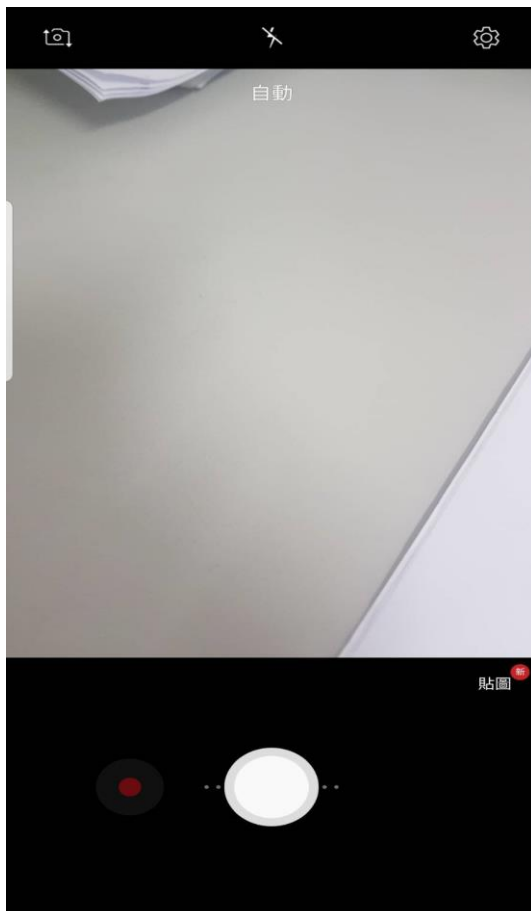
(完整程式碼)

步驟八.

- 執行結果



(手機 App 介面)



(開啟拍照模式)



(顯示圖片及路徑)

活動

請修改此 App，試著拍一張自拍照，在拍照完成後，將顯示路徑變為顯示自己的名字。

ImagePicker 元件介紹

在拍完許多照片後，我們常常會在某些時候需要挑選滿意的照片，上傳到電子履歷或社群軟體，這時候我們就可以使用 **ImagePicker 圖片選擇器元件**，來達到這樣選擇照片的功能。



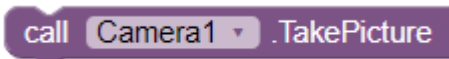
when ImagePicker.AfterPicking do 是當圖片選擇器被點選後，完成某個項目後所會觸發的事件。

call ImagePicker 則是主要用於呼叫 **ImagePicker 圖片選擇器**，讓使用者可以開啟圖片選擇功能。

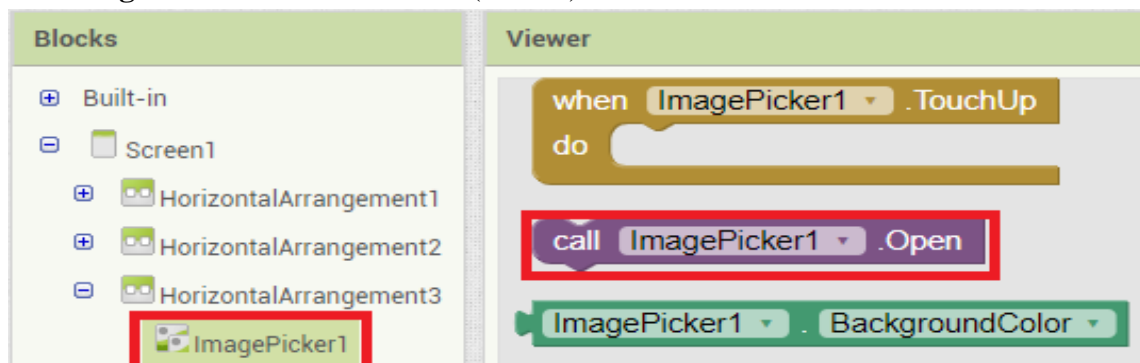
而 **when ImagePicker.AfterPicking do** 與 **call ImagePicker** 也是要新增完 **ImagePicker 圖片選擇器元件** 才會出現在方塊(Blocks)項目中。

呼叫 ImagePicker 元件方塊

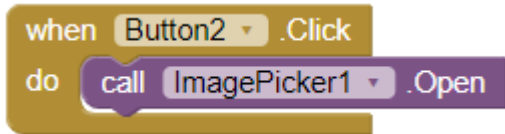
ImagePicker 圖片選擇器可以與本章的重點 **Camera 照相機** 進行結合運用，而要使用 ImagePicker 圖片選擇器的話，我們就需要用 **call ImagePicker** 來呼叫它。



call ImagePicker 方塊在內建方塊(Blocks)項目中，點選圖片選擇器元件。

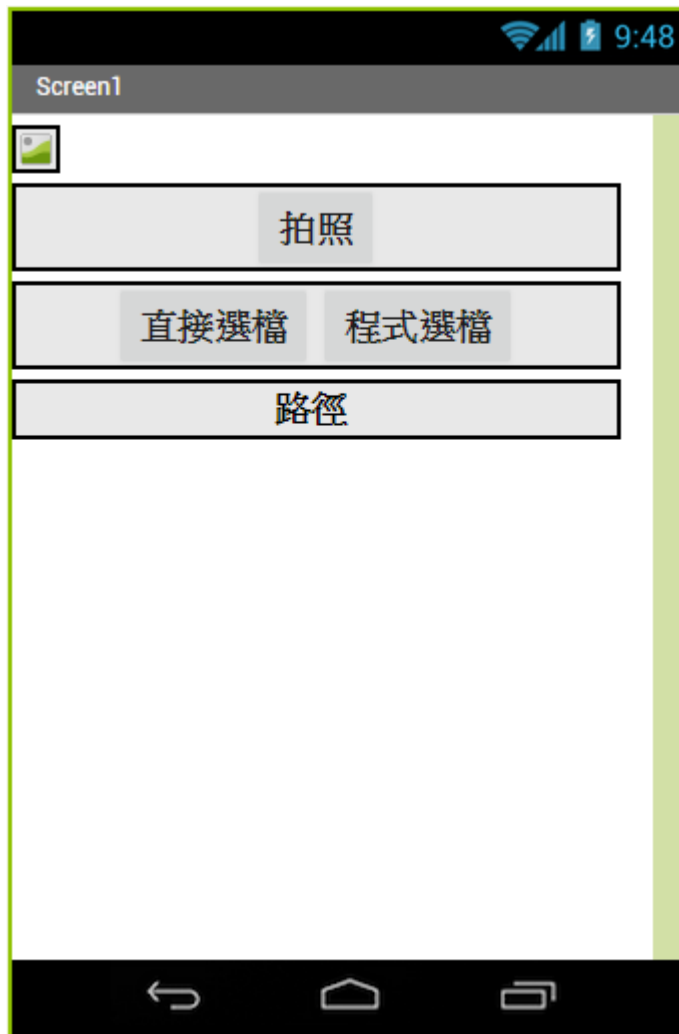


而 ImagePicker 圖片選擇器本身就能直接當一個元件按鈕使用，但我們也能使用 Button 按鈕元件與 **call ImagePicker** 方塊結合製作出開啟圖片選擇器的按鈕。



(開啟圖片選擇器的按鈕)

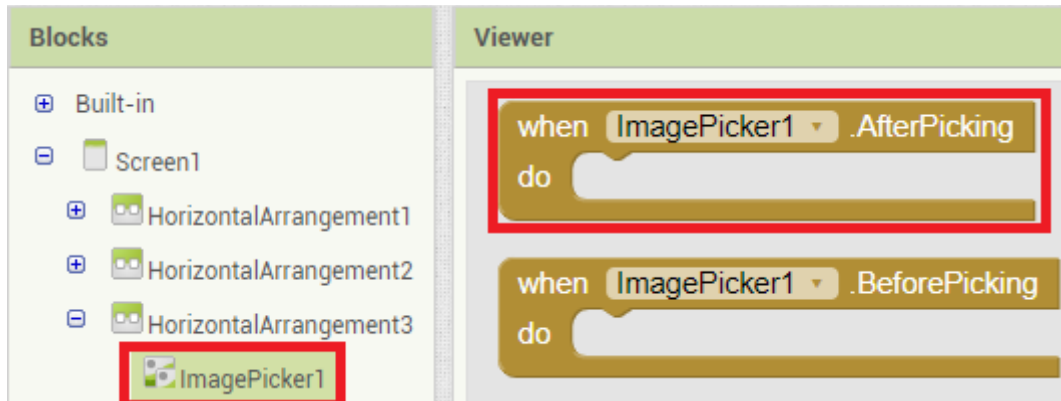
下面圖為 App 介面圖，左方直接選檔為 **ImagePicker** 圖片選擇器元件，而右方程式選檔則為 Button 按鈕元件與 **call ImagePicker** 方塊結合，我們可以看出來兩者在介面的顯示上並無差異。



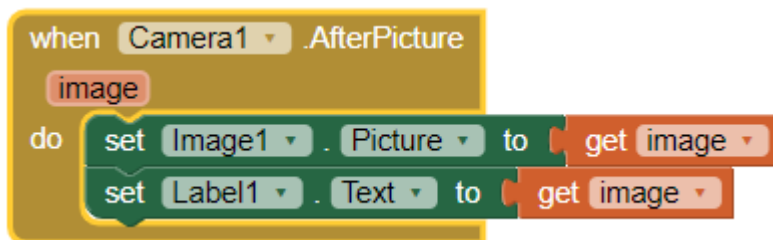
(App 介面圖)

ImagePicker.AfterPicking 方塊設定

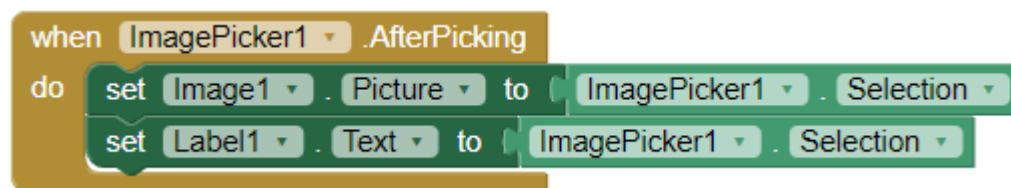
ImagePicker.AfterPicking 方塊一樣點選內建方塊(Blocks)項目，並選取圖片選擇器中的 **when ImagePicker.AfterPicking do** 新增。



與 **Camera.AfterPicture** 相同，我們能透過設定，顯示出我們使用 **ImagePicker** 選擇的照片及照片路徑，但 **Camera.AfterPicture** 使用的是 **Image** 字串參數來記錄圖片位置，而 **ImagePicker.AfterPicking** 是直接抓取檔案位置來開啟圖片。



(**Camera.AfterPicture** 元件方塊)



(**ImagePicker.AfterPicking** 元件方塊)

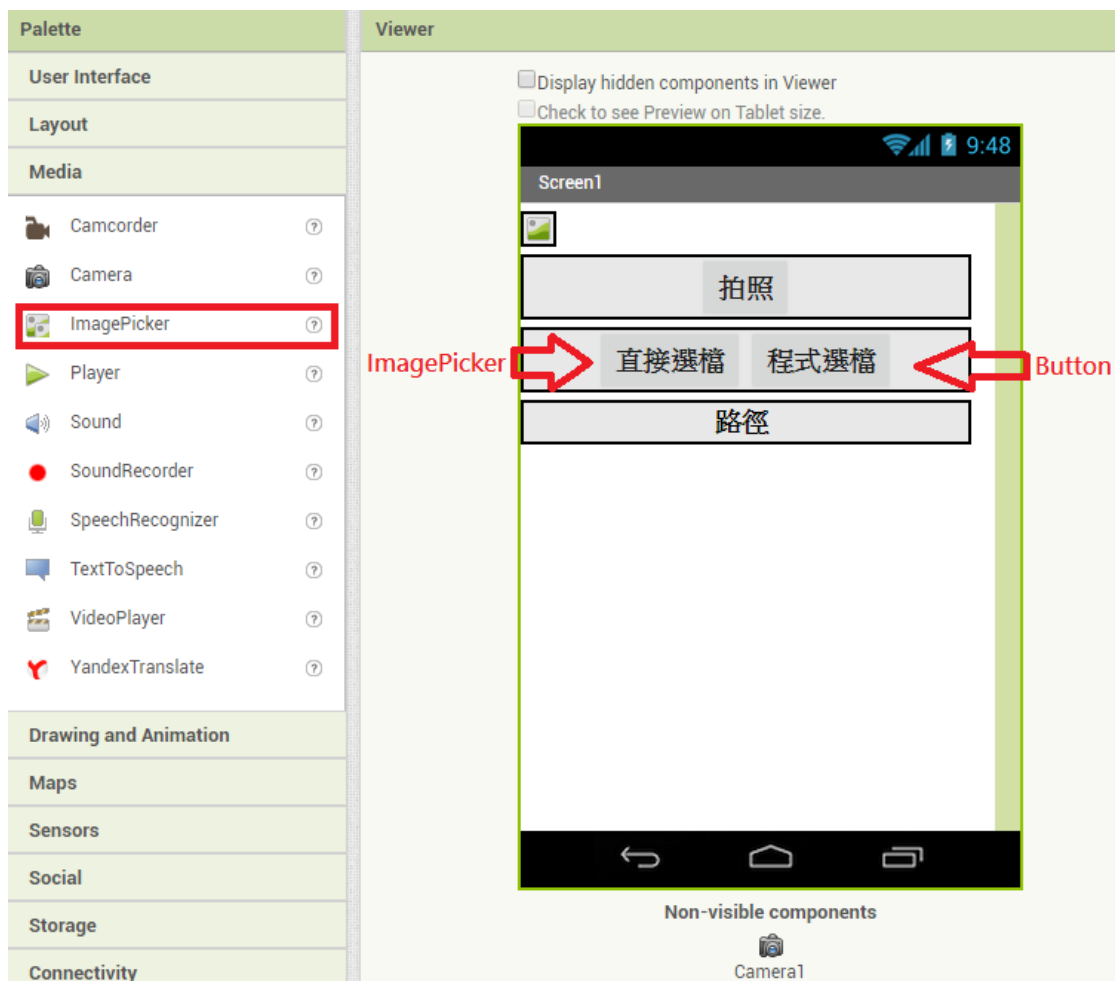
上圖你可以看出 **Camera.AfterPicture** 及 **ImagePicker.AfterPicking** 兩者的不同，**Camera** 是取得 **Image** 字串參數來顯示，而 **ImagePicker** 是用 **Selection**，也就是選取的項目來開啟圖片及其圖片的位置。

範例 - 照片選擇及開啟

請延續上一個範例，我們在範例上面新增一個 **ImagePicker** 圖像選擇器元件，以及一個與 **call ImagePicker** 方塊結合的 **Button** 按鈕，並透過圖像選擇器來選擇一張圖片並顯示它的路徑。

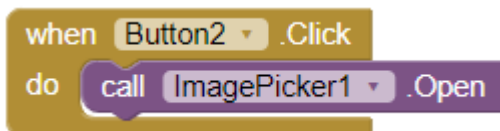
步驟一.

- 我們先將版面設置好，將上面所需要的物件拉至版面排版，而版面的配置依個人喜好編排即可。



步驟二.

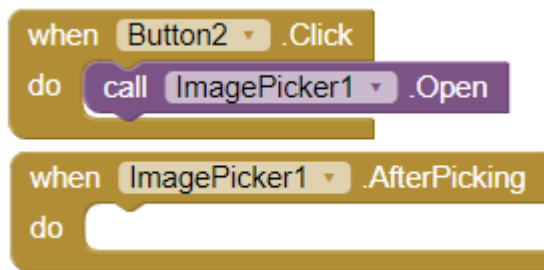
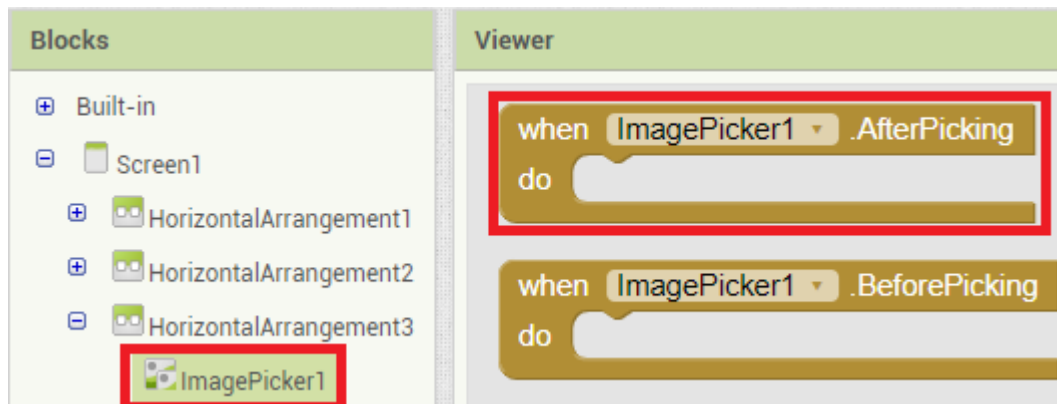
- 利用 **Button.Click** 結合 **call ImagePicker** 製作圖片選擇器開啟按鈕。



(觸發拍圖片選擇器開啟按鈕)

步驟三.

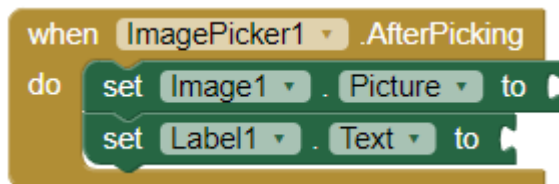
- 我們要設定使用圖片選擇器後，顯示出所選擇的圖片及其圖片路徑，點選 ImagePicker 新增 **ImagePicker.AfterPicking**。



(新增 **ImagePicker.AfterPicking** 方塊)

步驟四.

- 將 **ImagePicker.AfterPicking** 與 **set Image.Picture to** 及 **set Label.Text to** 組合起來，使用 **ImagePicker.Selection** 達到顯示所選取的圖片及其路徑的功能。



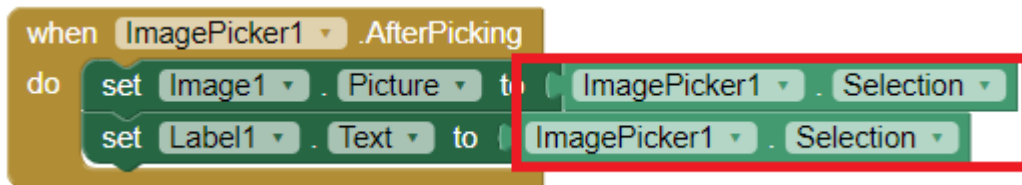
(組合 **set Image.Picture to** 及 **set Label.Text to**)

步驟五.

- 點選 ImagePicker 新增 **ImagePicker.Selection**，並將它連接到 **set Image.Picture to** 及 **set Label.Text to** 後面。



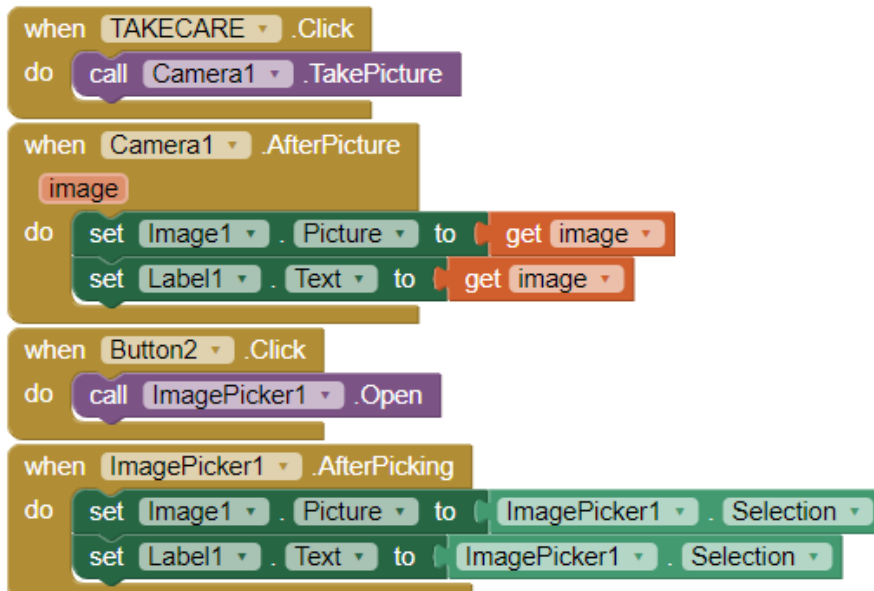
(選取 **ImagePicker.Selection**)



(新增 **ImagePicker.Selection** 方塊)

步驟六.

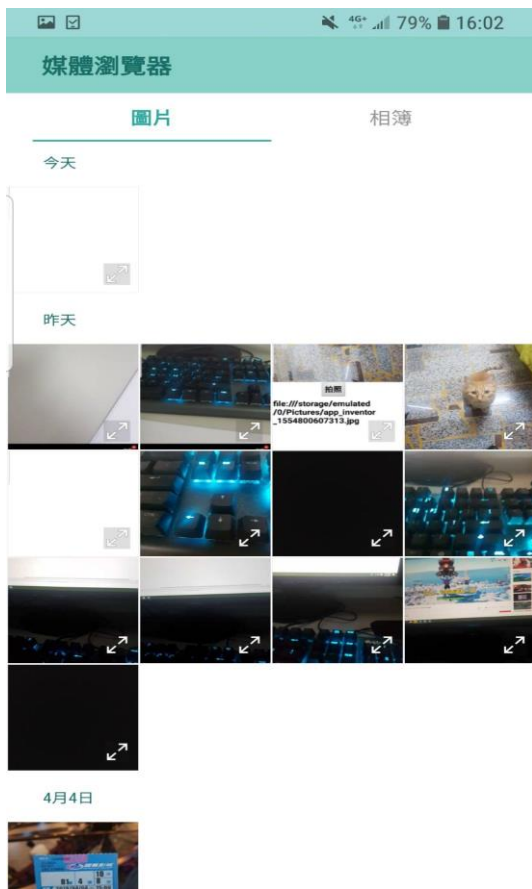
- 完整程式碼



(完整程式碼)

步驟七.

- 執行結果



(開啟圖像選擇器)



(顯示選取圖片及路徑)

活 動

請試著使用看看 **ImagePicker** 按鈕與結合 **call ImagePicker** 的按鈕，兩者所產生的結果是否有所不同。