Unit 2 常用元件與運算

作者：林煜衡

元件

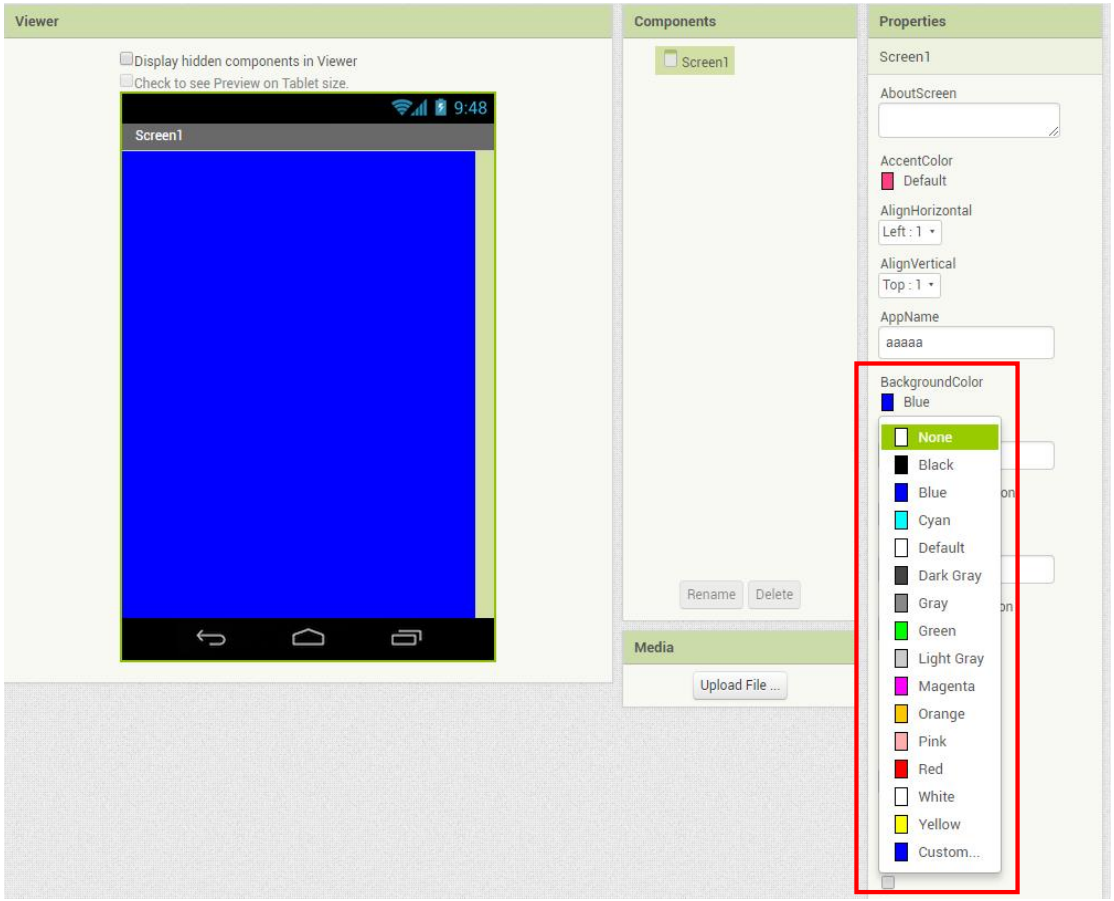
螢幕元件（Screen）

當新增專案時，MIT App Inventor 2 網站會自動新增一個名稱為「Screen1」的螢幕元件，螢幕元件是其他元件的容器。如果需要多個螢幕元件，按下「Add Screen」按鈕即可新增，按下「Remove Screen」按鈕即可移除螢幕元件。

Screen 元件的常見屬性說明：

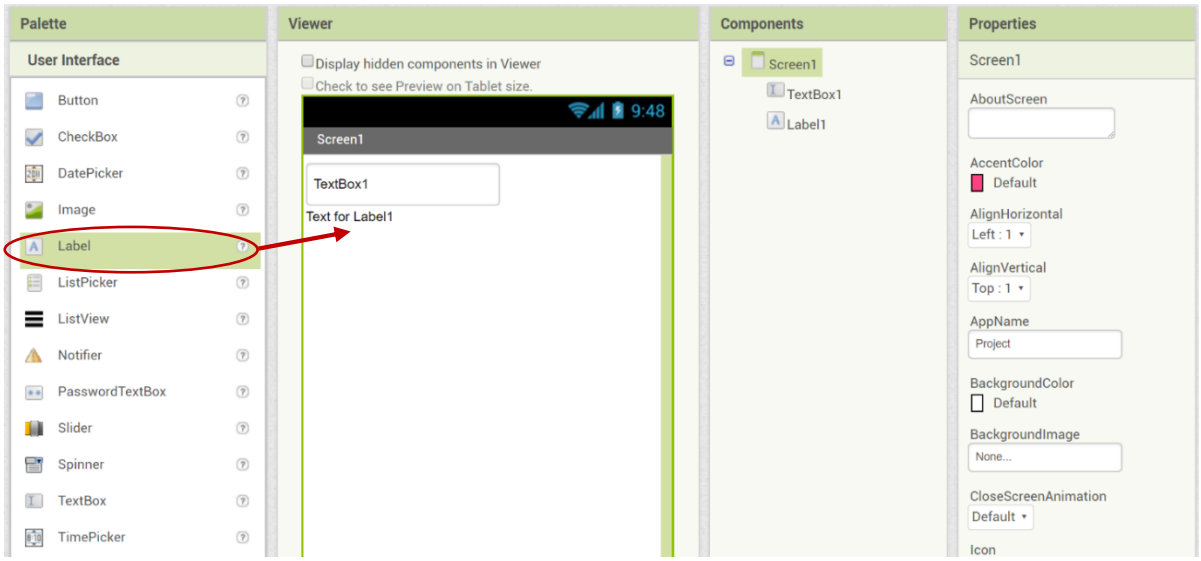
屬性名稱	屬性說明
AlignHorizontal	水平對齊方式，包含 Left、Center 與 Right 來靠左、置中或靠右對齊
AlignVertical	垂直對齊方式
BackgroundColor	背景顏色
BackgroundImage	背景圖案
Icon	安裝於手機上之後，於所有程式選單中的 App 小圖示
OpenScreenAnimation	開始螢幕動畫
CloseScreenAnimation	關閉螢幕動畫，共有預設 Default、淡出 Fade、縮小 Zoom、水平滑出 SlideHorizontal、垂直滑出 SlideVertical 以及無動畫 None 等選項。
ScreenOrientation	螢幕方向，包含了未指定 Unspecified、垂直 Portrait、橫放 Landscape、感測器 Sensor 與使用者自定 User 等選項。
Scrollable	畫面是否可以捲動
Title	標題列內容

下圖將 Screen1 的 BackgroundColors 設為藍色。



標籤元件（Lable）

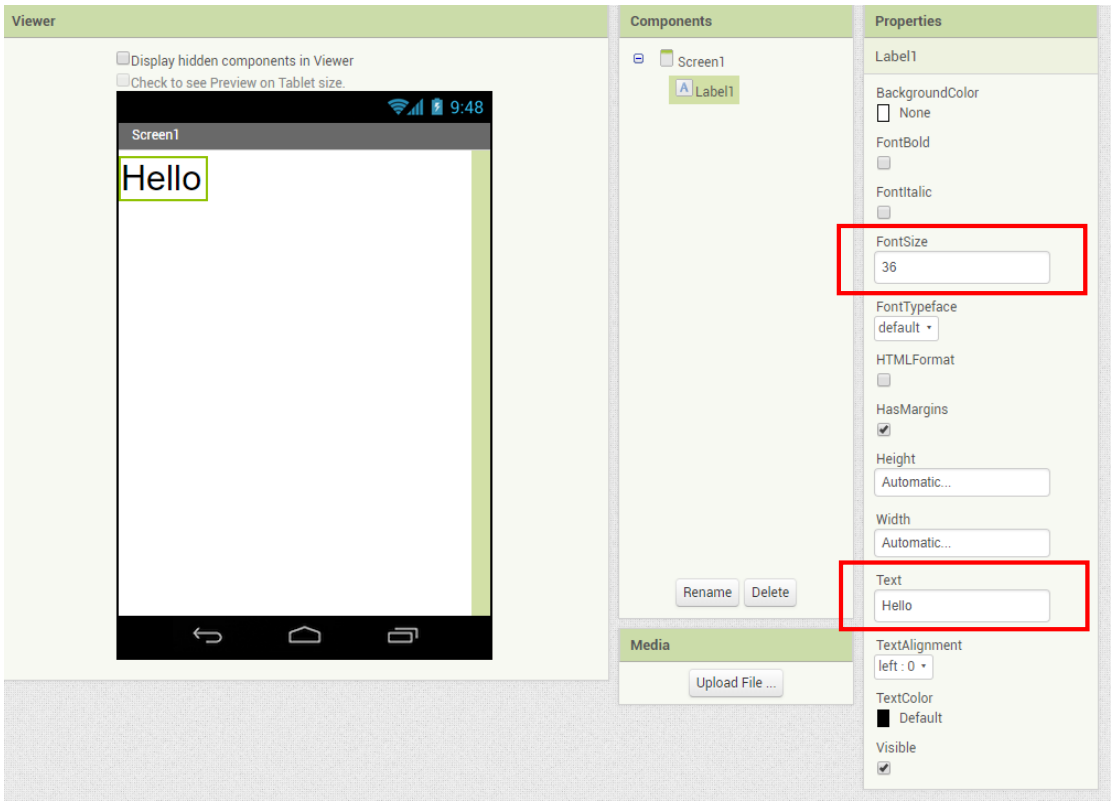
標籤元件的主要功能是用來顯示文字。



Lable 元件的常見屬性說明：

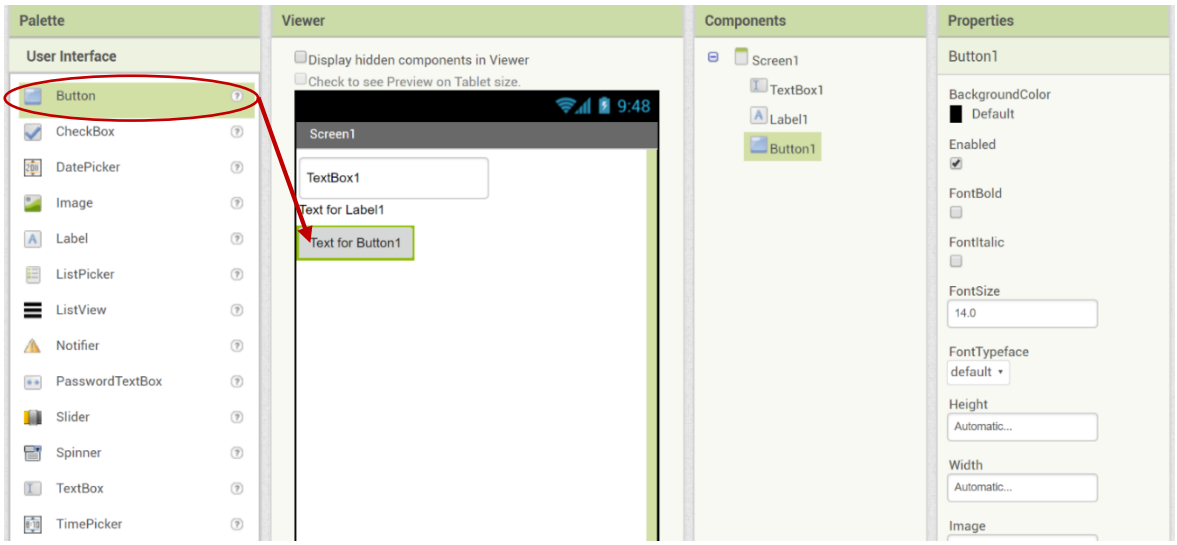
屬性名稱	屬性說明
BackgroundColor	標籤文字方塊背景顏色
FontBold	粗體文字
FontItalic	斜體文字
FontSize	標籤文字大小
FontTypeface	標籤文字字體
Text	標籤文字方塊的文字
TextAlignment	標籤文字方塊內的文字對齊位置（靠左、置中、靠右）
TextColor	標籤文字顏色
Visible	是否顯示標籤文字方塊
Width	標籤文字方塊的寬度
Height	標籤文字方塊的高度

下圖將 Label 的 Text 改為 Hello、FontSize 改為 36。



按鈕元件（Button）

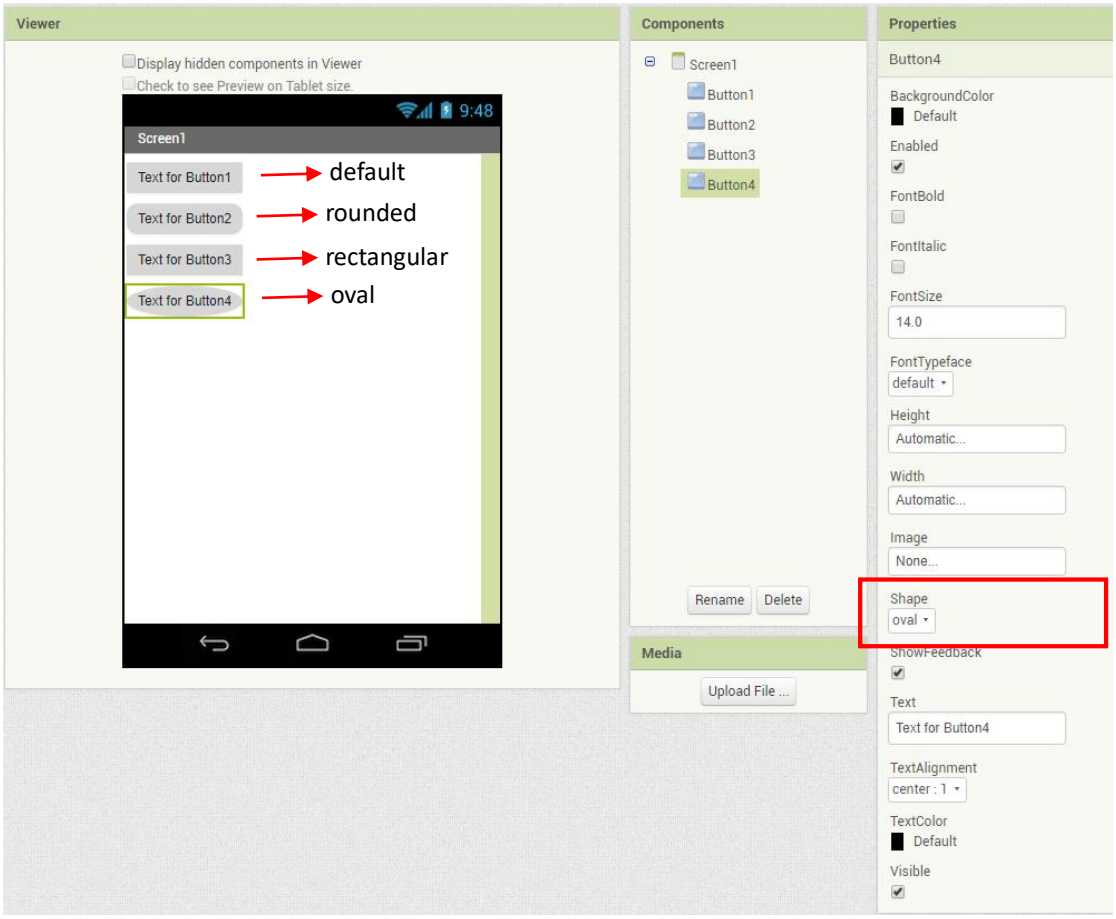
按鈕元件的主要功能是藉由使用者觸發事件（Event），以進行程式的運作。



Button 元件的常見屬性說明：

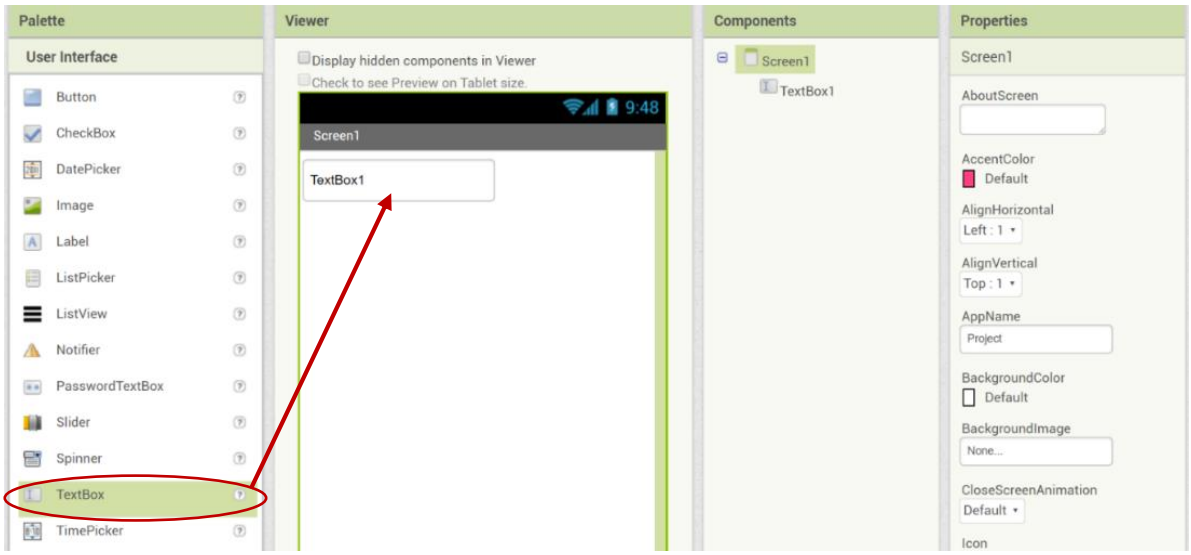
屬性名稱	屬性說明
BackgroundColor	按鈕背景顏色
Enabled	是否使用按鈕
FontBold	粗體文字
FontItalic	斜體文字
FontSize	文字大小
FontTypeface	文字字體
Image	圖片按鈕
Shape	按鈕形狀（預設為矩形，可改為圓形、橢圓形）
ShowFeedback	當使用圖片按鈕時，按下按鈕是否有回饋反應
Text	按鈕的文字
TextAlignment	按鈕的文字對齊位置（靠左、置中、靠右）
TextColor	按鈕文字顏色
Visible	是否顯示按鈕
Width	按鈕的寬度
Height	按鈕的高度

下圖分別將 Button 的 Shape 屬性不同設定值顯示於 Screen 中。



文字方塊元件（TextBox）

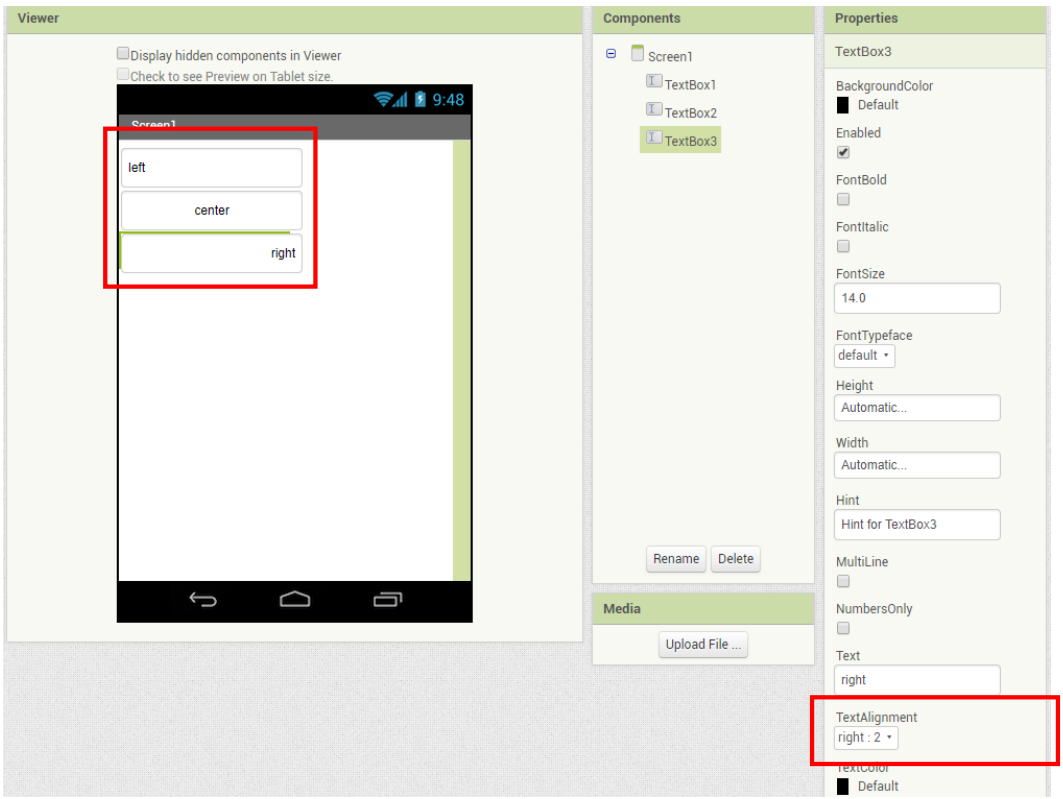
文字方塊是讓使用者輸入文字的元件。



TextBox 元件的常見屬性說明：

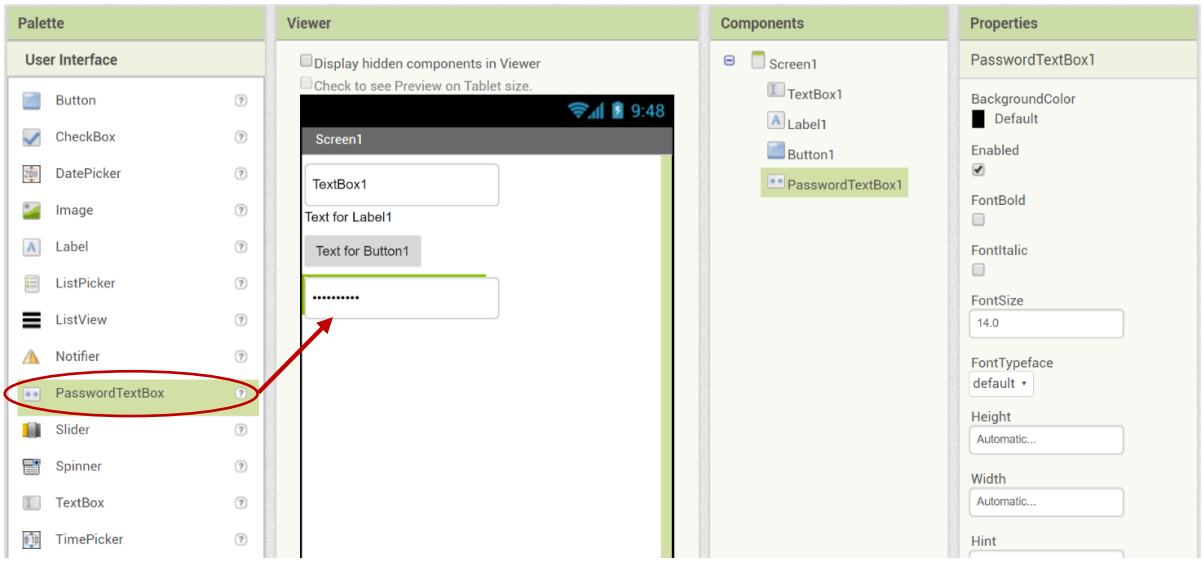
屬性名稱	屬性說明
BackgroundColor	文字方塊背景顏色
Enabled	是否使用文字方塊
FontBold	粗體文字
FontItalic	斜體文字
FontSize	文字大小
FontTypeface	文字字體
Hint	文字方塊的提示文字
MultiLine	是否可以輸入多行文字
NumbersOnly	只能輸入數字
Text	文字方塊的文字
TextAlignment	文字方塊內的文字對齊位置（靠左、置中、靠右）
TextColor	文字顏色
Visible	是否顯示文字方塊
Width	文字方塊的寬度
Height	文字方塊的高度

下圖將 TextBox 的 TextAlignment 屬性的不同設定值顯示於 Screen 中



密碼文字方塊元件（PasswordTextBox）

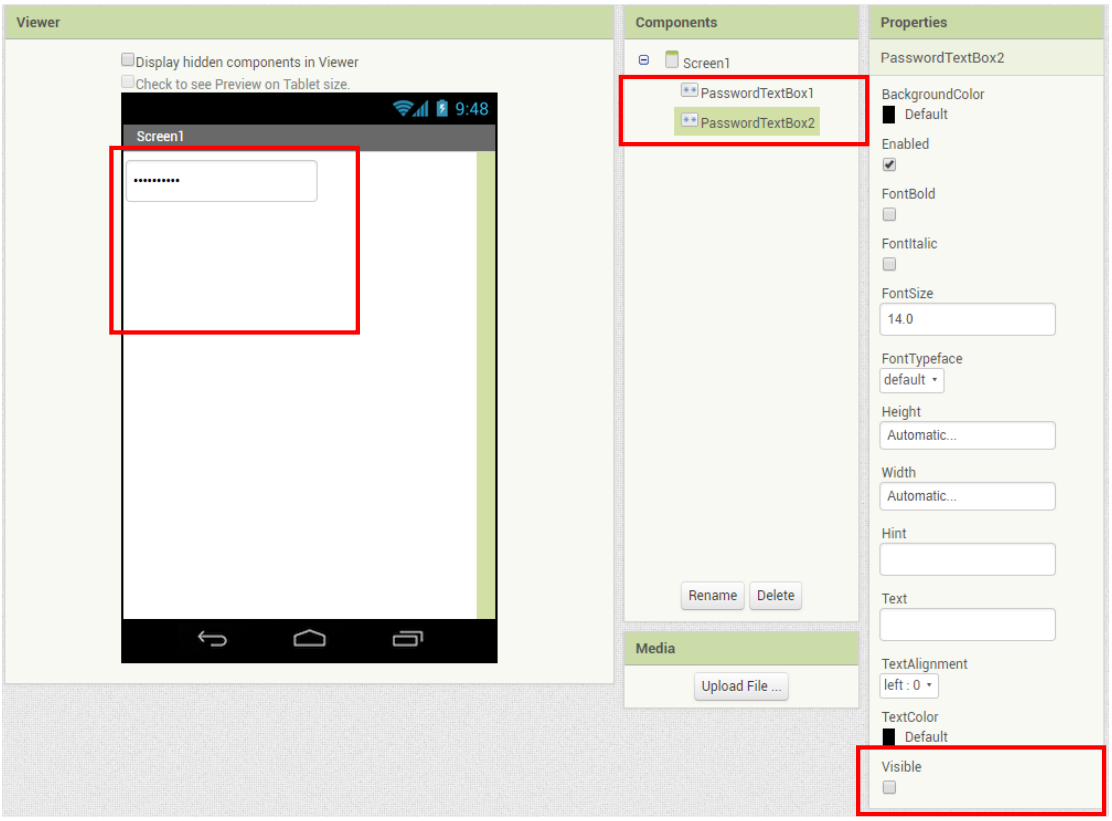
密碼文字方塊元件是用來輸入密碼的文字方塊，使用密碼文字方塊輸入文字，會以「＊」或「●」來取代，以達到資料保密的目的。



PasswordTextBox 元件的常見屬性說明：

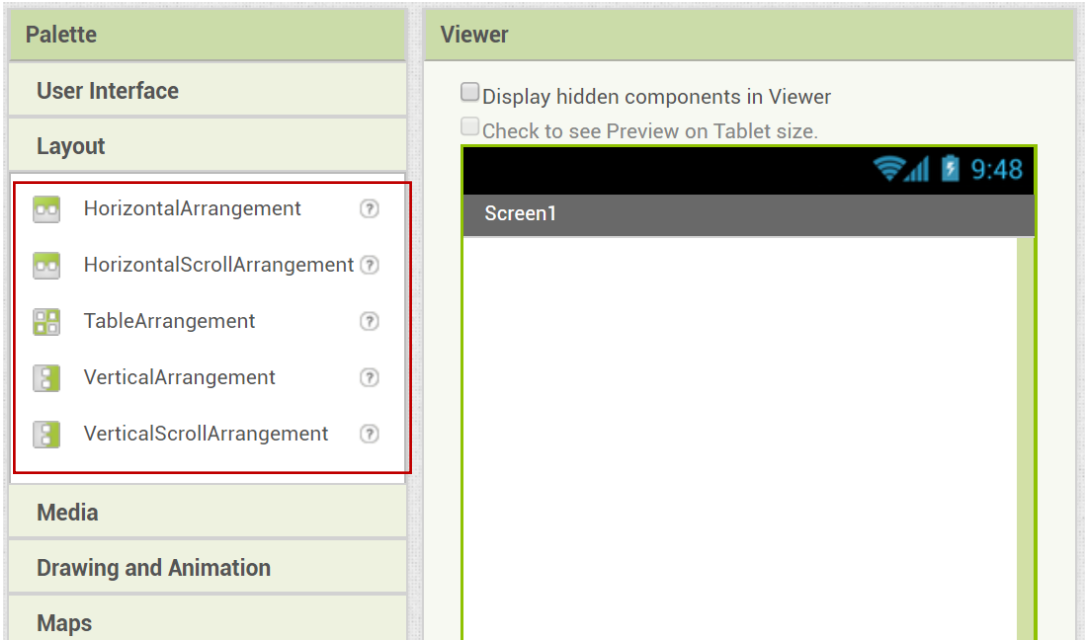
屬性名稱	屬性說明
BackgroundColor	密碼文字方塊背景顏色
Enabled	是否使用密碼文字方塊
FontBold	粗體文字
FontItalic	斜體文字
FontSize	文字大小
FontTypeface	文字字體
Hint	密碼文字方塊的提示文字
Text	密碼文字方塊的文字
TextAlignment	密碼文字方塊內的文字對齊位置（靠左、置中、靠右）
TextColor	文字顏色
Visible	是否顯示密碼文字方塊
Width	密碼文字方塊的寬度
Height	密碼文字方塊的高度

下圖將其中一個 PasswordTextBox 的 Visible 屬性取消，可以看到雖然 Components 裡有兩個 PasswordTextBox，但在 Viewer 只顯示一個。

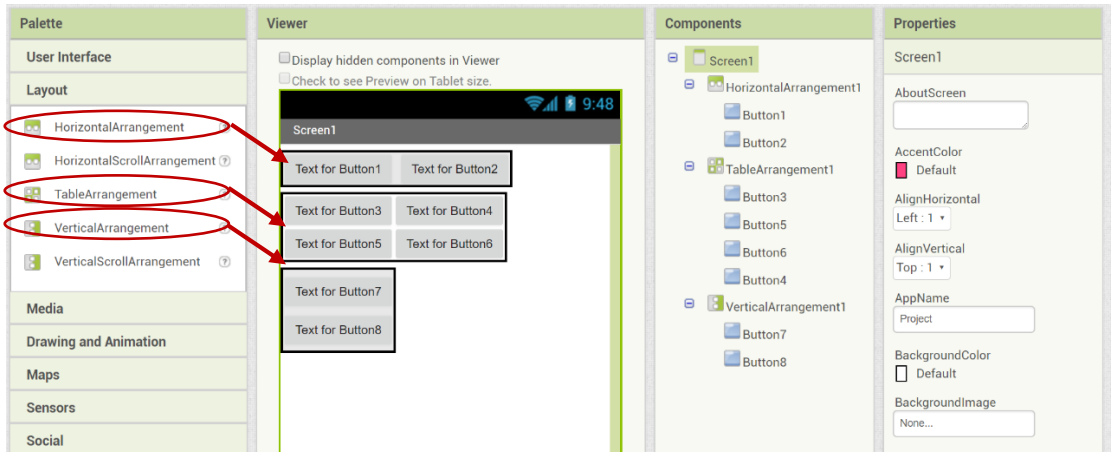


編排元件（Layout）

編排元件是用來對螢幕元件做排版，方便我們對螢幕上的元件進行對齊。Layout 具有容器的概念，首先在螢幕元件上布置 Layout 元件，然後再將其他元件放入 Layout 元件中即可。



常用的 Layout 元件對齊方式有以下三種。HorizontalArrangement（橫向排列）、TableArrangement（表格排列）、VerticalArrangement（垂直排列）

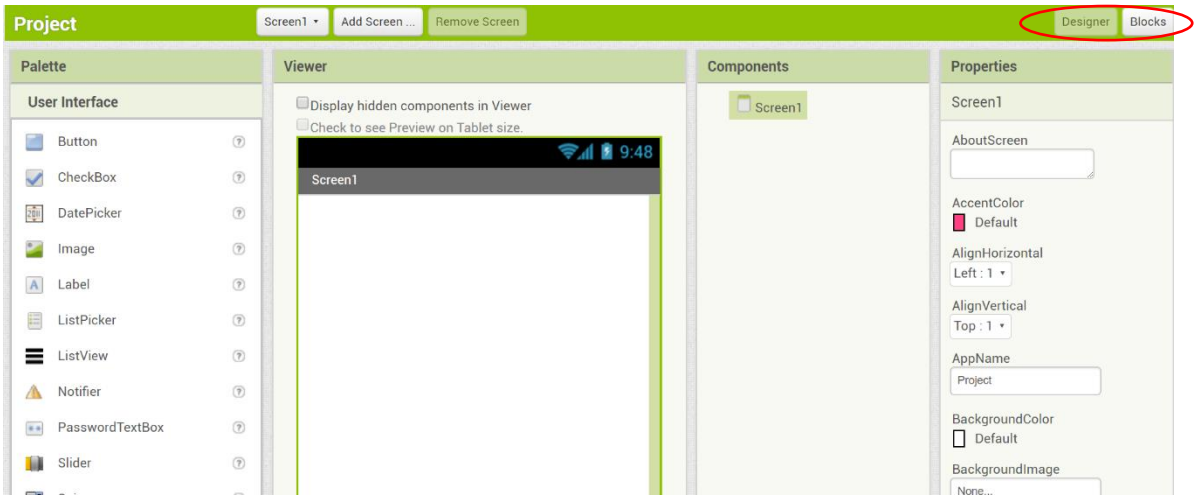


練習題

在 Screen 元件上布置 1 個 Layout 元件，並在 Layout 元件中放入 1 個 TextBox 元件、1 個 Button 元件和 1 個 Label 元件。

拼圖模式操作

MIT App Inventor 2 的程式設計方式是透過拼圖模式來完成。要切換到拼圖模式，按下「Blocks」按鈕，如要返回編輯元件，按下「Desinger」按鈕。



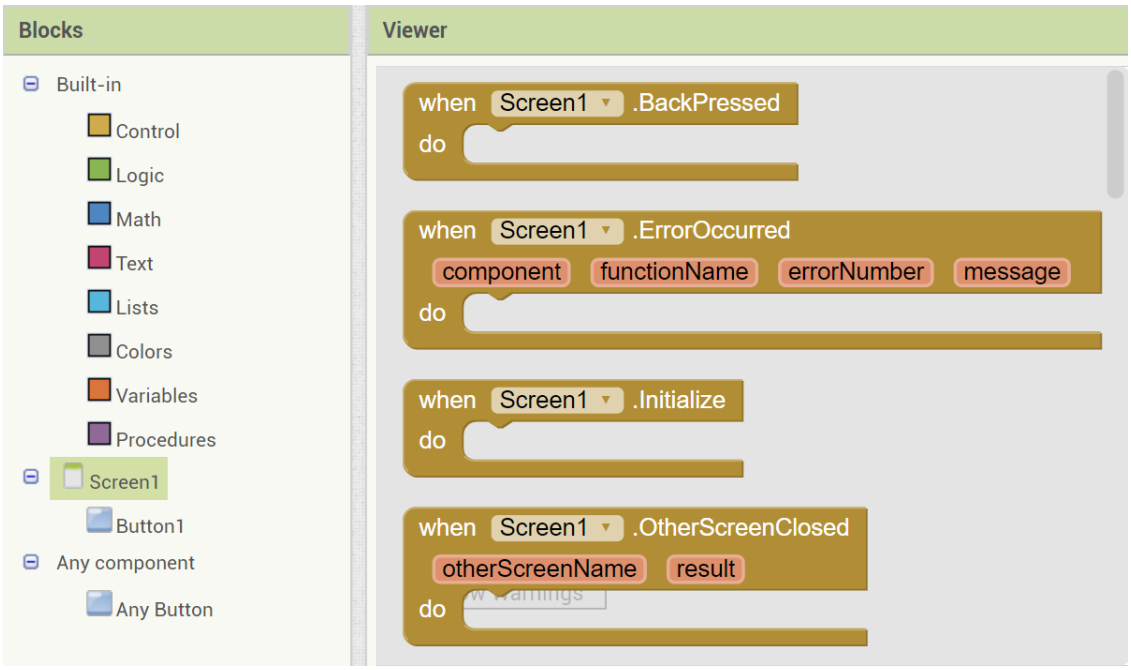
「Blocks」面板可分為三部分

Built - in

此為系統內建的拼圖塊，分為八類。控制 (Control)、邏輯 (Logic)、數學 (Math)、文字 (Text)、清單 (Lists)、顏色 (Colors)、變數 (Variables)、程序 (Procedures)、

Screen

顯示出布置在 Screen 中的元件所支援的所有拼圖塊。例如在 Button 元件中當使用者點擊 Button 時可以觸發「when Button.Click do」，也可設定（set）或取得（get）Button 中的屬性。

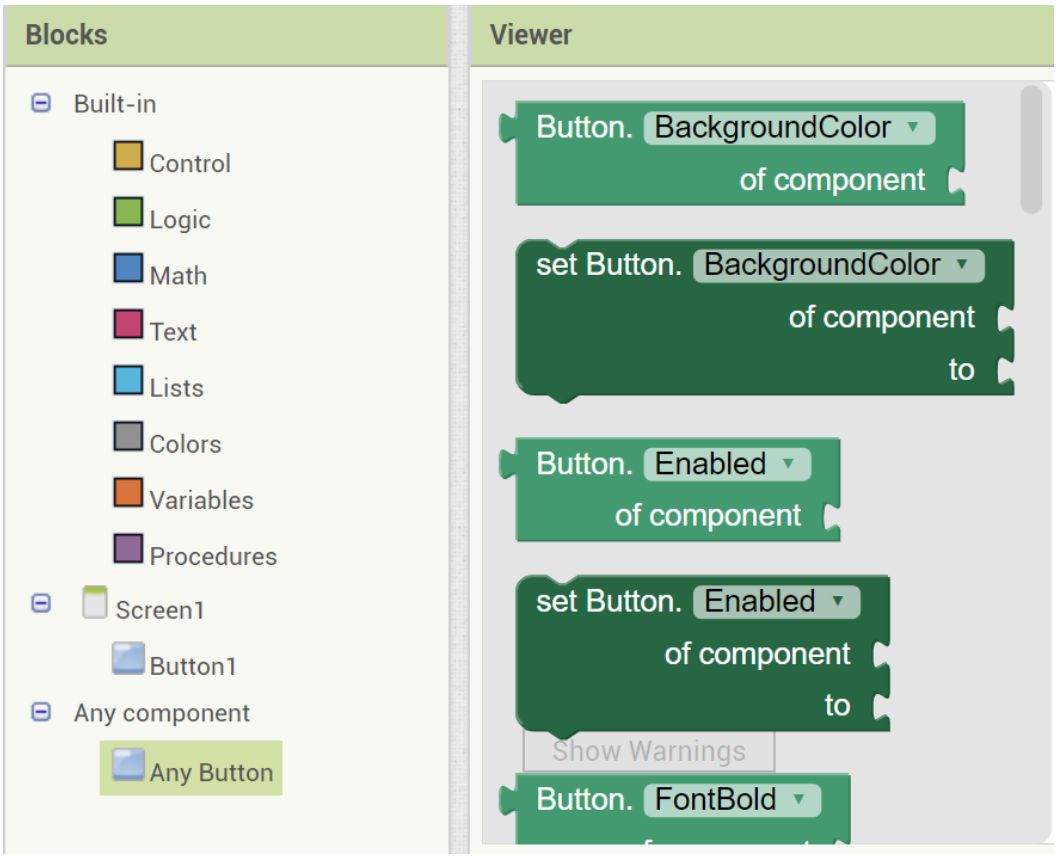


上圖為 Screen 支援的事件往下捲動有其他的方法及屬性，Initialize：應用程式一啟動時就同步呼叫本事件，本事件可用來初始化某些變數以及執行一些前置性的操作。

上圖為 Button 支援的事件往下捲動有其他的方法及屬性，Click：當使用者點擊並放開按鈕時，執行 do 區塊中的指令。

Any Component

將布置在 Screen 的元件做分類，將同類型的元件群組在一起。



上圖為 Any Component 的 Button 支援的屬性如果利用此方塊進行屬性設定則會改變在 Screen 中所有相同類型的元件屬性。

程式設計四大天王

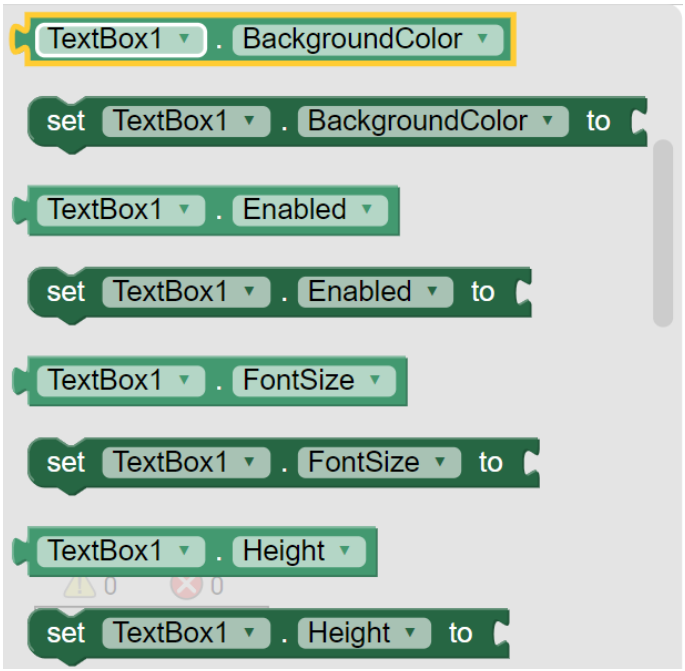
元件

在日常生活中，任何實體的物品都能視為元件。而在程式設計裡的元件是將內部資訊封裝 (encapsulate)，只提供「屬性」、「方法」給外界操作。因此我們不需了解元件內部的運作，只需要了解元件提供的界面和特性，就可以直接使用元件的功能。

屬性

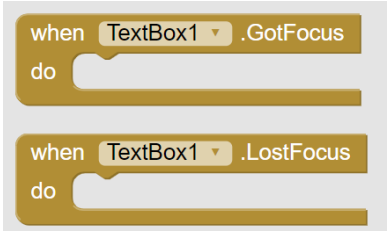
屬性是用來描述元件的特性。日常生活也經常用到屬性的概念，例如：物品的名稱、外觀...。在程式設計中可以透過設定屬性來改變元件的名稱、外觀...。

MIT App Inventor 2 中的屬性設定可以在 Designer 頁面中的 Properties 設定，或在 Blocker 頁面中的 Screen 拖曳拼圖塊「Set」設定元件屬性 (深綠色)、取得元件屬性 (淺綠色)。



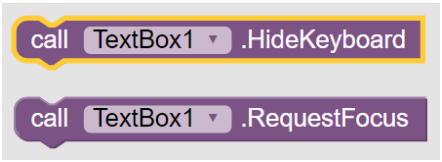
事件

事件是用來驅動程式運作的一個動作。例如：使用者按下按鈕、觸碰文字方塊...等。當這些元件的動作發生時，就會驅動程式運作該事件下的拼圖塊。其拼圖塊顏色為土黃色。



方法

方法是元件本身就具有的功能。例如：TextBox 中的「HideKeyboard」可以決定文字方塊輸入時，不要顯示鍵盤。其拼圖塊為紫色。

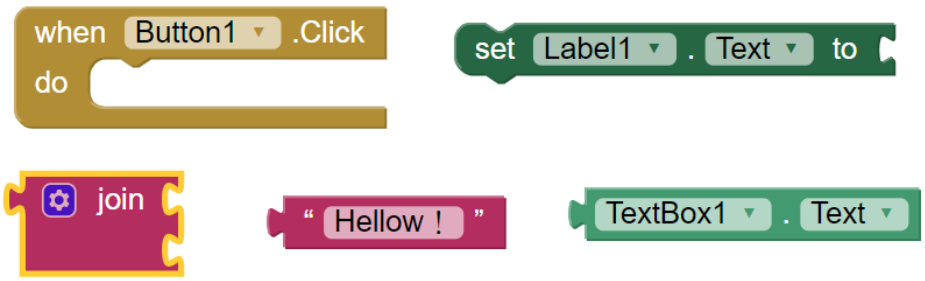


根據方塊的接口可以判斷事件是其他方塊的起點，而方法與設定屬性必須連接事件或是接續其他的方法或設定屬性方塊才能使用。

練習題

在 Screen 元件上布置 1 個 Layout 元件，並在 Layout 元件中放入 1 個 TextBox 元件、1 個 Button 元件和 1 個 Label 元件。當使用者在文字方塊中輸入自己的姓名後，按下按鈕將所輸入的姓名顯示在 Label 元件，例如：在 TextBox 輸入「Jason」，按下按鈕後 Label 顯示「Hello！Jason」。

提示：會用到以下拼圖。



常數與變數

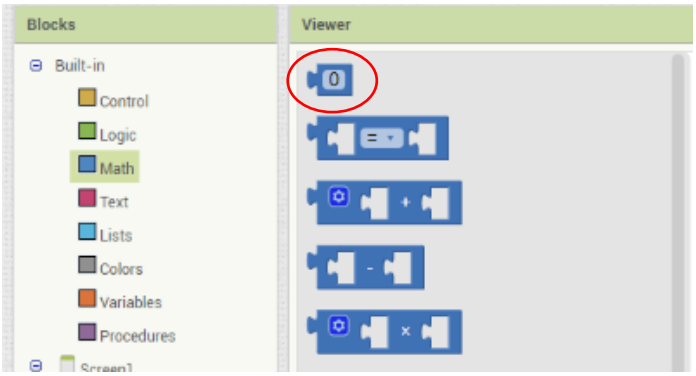
在使用變數之前，必須先告訴電腦「我要使用變數」，電腦會幫我們準備記憶空間儲存該變數。這個動作稱為「宣告」，而宣告又分為「常數」和「變數」，資料型態又分為「數值」、「字串」、「邏輯」。

常數(constant)

常數在建立時就設定其初始值，此常數不能在程式中加以改變。例如：圓周率。

數值常數

用來儲存數字的常數。



點選內建方塊(Blocks)項目點選數學(Math)，選擇數值方塊。

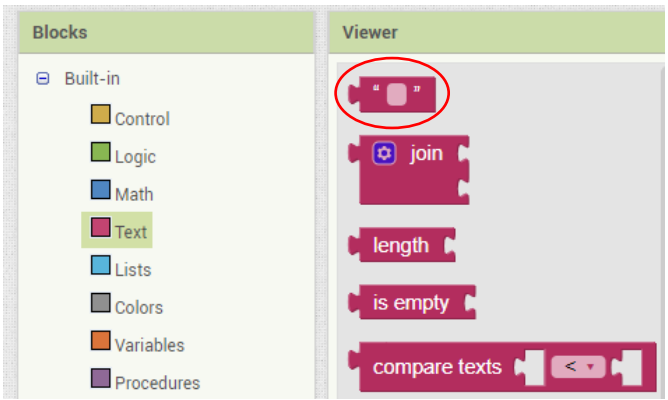
數值常數在設定前，預設值為 0。

要改變數值常數為 10 直接把預設值 0 改為 10 即可。

若在數值常數中輸入文字則會被視為無效輸入其值會還原為預設值 0。

字串常數

用來儲存文字的常數。



點選內建方塊(Blocks)項目點選文字(Text)，選擇字串方塊。

數值常數在設定前沒有任何字元，即為空字串。

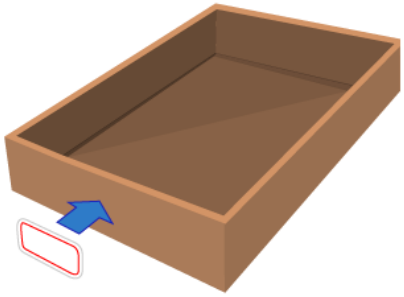
字串常數不只可以輸入英文字也可以輸入中文字，改變字元只需直接輸入即可如上圖。

邏輯常數

邏輯常數是用來儲存是或非的常數用來設定元件的布林 (boolean) 屬性值，或用來表示某種狀況是否成立。因此只有 true 與 false 兩種值。

變數 (Variable)

變數是指一個包含數值或資訊 (即一個值) 的儲存位址，使用容易辨識的符號名稱代表該儲存位址。也可以把它視為一個具有名稱的盒子，裡面可以放置數值、文字等資料。



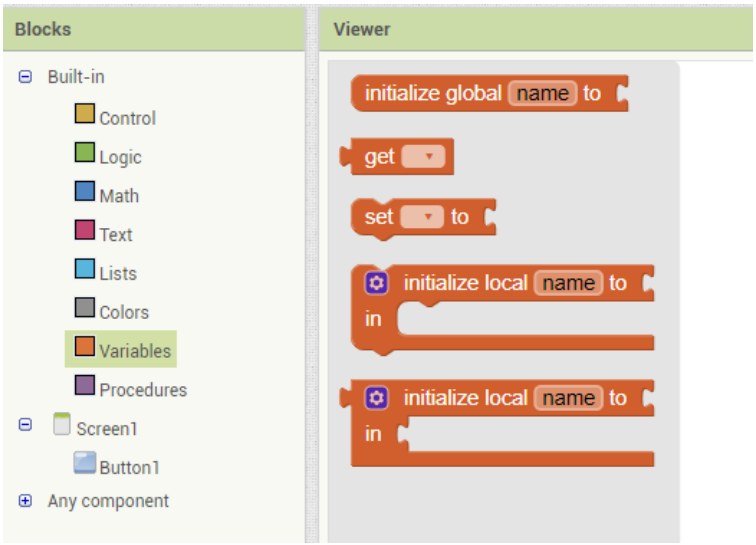
(圖片來源)

電腦內有大量的記憶體空間可存放資料，就像是有很多的盒子在電腦裏面，變數名稱如同貼一個標籤在某個盒子，而變數的值則為將值存放其中，之所以稱為變數表示在程式的執行過程中是可以被改變的。

在許多程式語言中，變數的有效存取範圍可分為全域變數(Global Variable)與區域變數(Local Variable)。App Inventor 2 亦將變數分為此兩類，其全域變數是指變數的作用範圍在「整個頁面」，此頁面的任何一個事件都可直接取得、修改，當此頁面關閉時，全域變數就會消失。

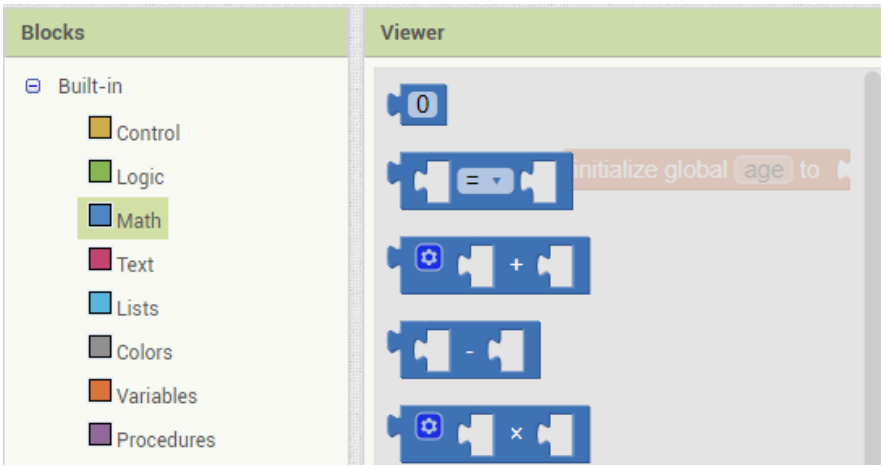
全域變數宣告與初始值設定

全域變數的宣告方式是在內建方塊(Blocks)項目點選變數(Variables)，選擇 **initialize global name to** 拼塊。



接下來可更改方塊中的 name，例如改為 age，表示使用一個名稱為 age 的全域變數。

此時變數 age 還沒有存放任何的資料，我們可以設定其中的內容(可以為數值、文字或邏輯值)，以此例來說，我們想要存放數值 20 到全域變數 age 中。

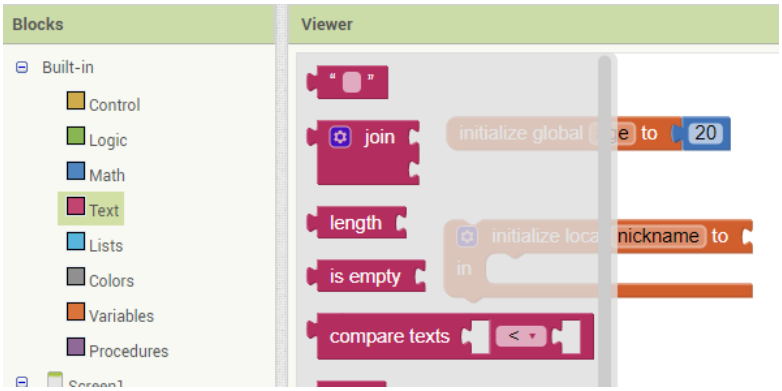


點選 Math 後選擇數值方塊拼貼至全域變數方塊後，修改數值 0 為 20。

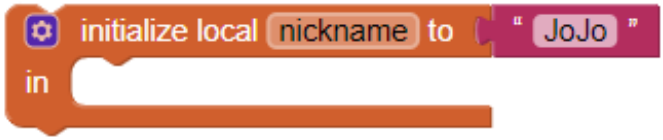
區域變數宣告與初始值設定

區域變數有存取範圍限制，宣告方式如同全域變數是在內建方塊(Blocks)項目點選變數(Variables)，選擇 **initialize local name to** 方塊，方塊有兩個，功能相同僅接口不同。

假設我們想要一個區域變數 nickname 存放暱稱字串，則將方塊原為 name 的名稱改為 nickname，並於內建方塊(Blocks)項目點選文字(Text)項目，選擇最上方的空字串方塊並拼接至區域變數方塊上。



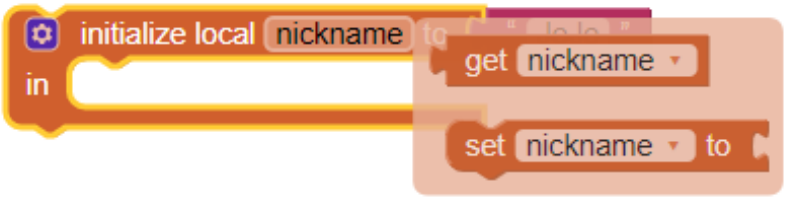
修改字串方塊內的文字為 JoJo，如下例所示



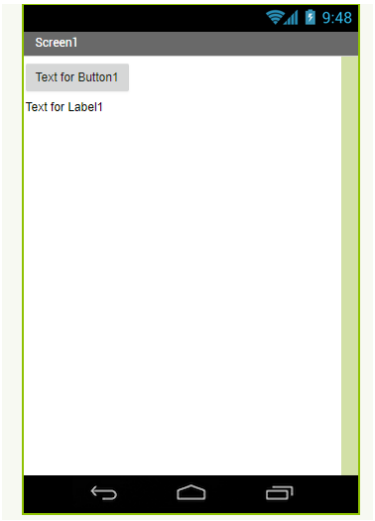
區域變數的有效存取範圍只有在 **in** 的缺口內，離開了缺口範圍，其他地方是看不到 nickname 這個變數的。因為僅為部分區域能存取，因此這種變數稱為區域變數。

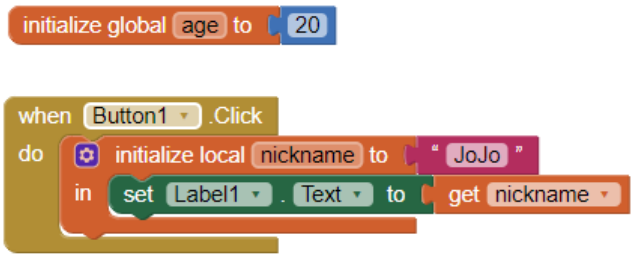
存取變數值

取得變數值的方塊可經由點選變數方塊後，將滑鼠移到宣告的變數名稱上，過一會將會出現兩個方塊可供選擇，一個是 get (取得)，另一個則為 set .. to (設定)。

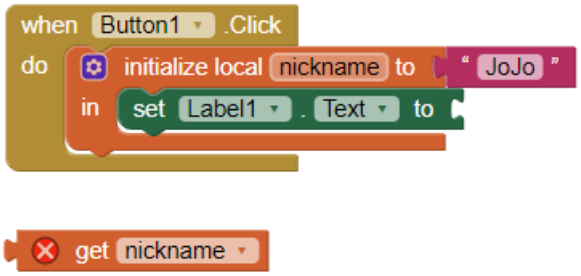


以下例試做看看，我們先在 Screen 1 配置一個按鈕及一個 Label，當按下按鈕時，將 Label 的文字改為 nickname 的值。





區域變數的使用是有範圍限制的，例如上例如果將 get nickname 方塊移出有效範圍，我們可以發現方塊前面出現紅色叉叉，表示放在外面是無效的。



活　動

請修改上面的例子，按下按鈕時先將全域變數 age 的值修改為 25 後，利用 Label1 輸出 age 的內容。

練習題

在 Screen 中，放入兩個 TextBox、一個 Label 和一個 Button，並在程式中宣告兩個變數用來儲存兩個 TextBox 的資料。請使用者在兩個 TextBox 中分別輸入兩個數後，當按下 Button 時，將兩個變數列印在 Label 中。例如：使用者分別輸入 aaaa 和 bbbb，當按下 Button，Label 會顯示出" 變數 1=aaaa 變數 2=bbbb" 。

提示：列印結果可以利用 TextBox 中的 TextBox.Text 屬性取得使用者輸入的值。

補充內容

一些傳統的程式語言，如 C、C++、JAVA，在使用變數前需要進行宣告，以利系統保留記憶體空間避免浪費，例如需存放整數就不需要像存放帶小數點的數字空間這麼大。另一個使用宣告的原因則為幫助程式設計者避免存放錯誤的資料內容，例如一個宣告為存放數值的變數，當不小心在後面的程式中放入字串，編譯器在編譯程式時就會提出警告或錯誤訊息。

C 語言變數宣告並設定數值範例

```
int year;    //宣告一個名稱為 year 的整數變數
year =21; //設定 year  內存放的數值為 21
```

部分程式語言為了便利學習或是方便使用，在使用變數前可以不需要宣告就直接使用，將由系統直接透過他的值去判斷類型，例如：Python

```
year=21 //設定一個整數變數，其中存放值為 21
name=”Greg” //設定一個字串變數，存放字串 Greg
year=name //將 year  的內容改由 name 的內容取代
```

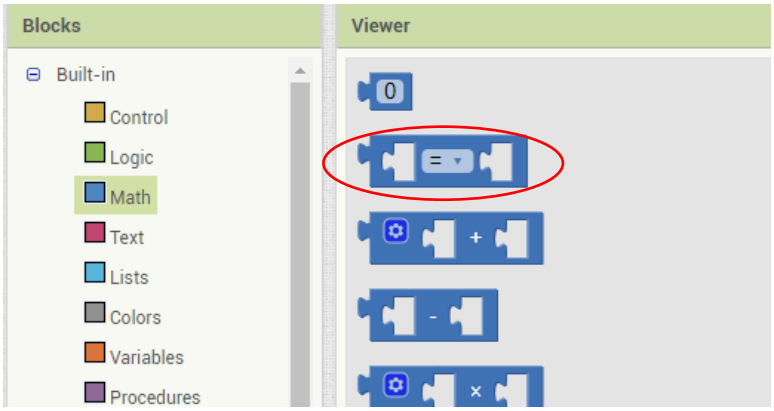
比較、邏輯與數學運算

比較運算

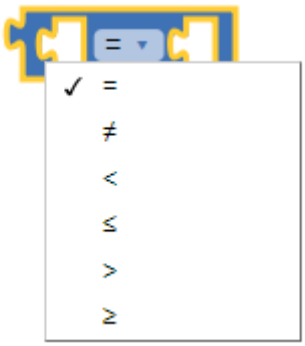
用於比較兩數字的大小，比較完後將結果以「布林值」(True 或 False) 回傳。

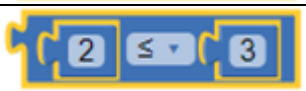
使用比較運算

比較運算是用內建方塊(Blocks)項目點選數學(Math)，選擇比較方塊。



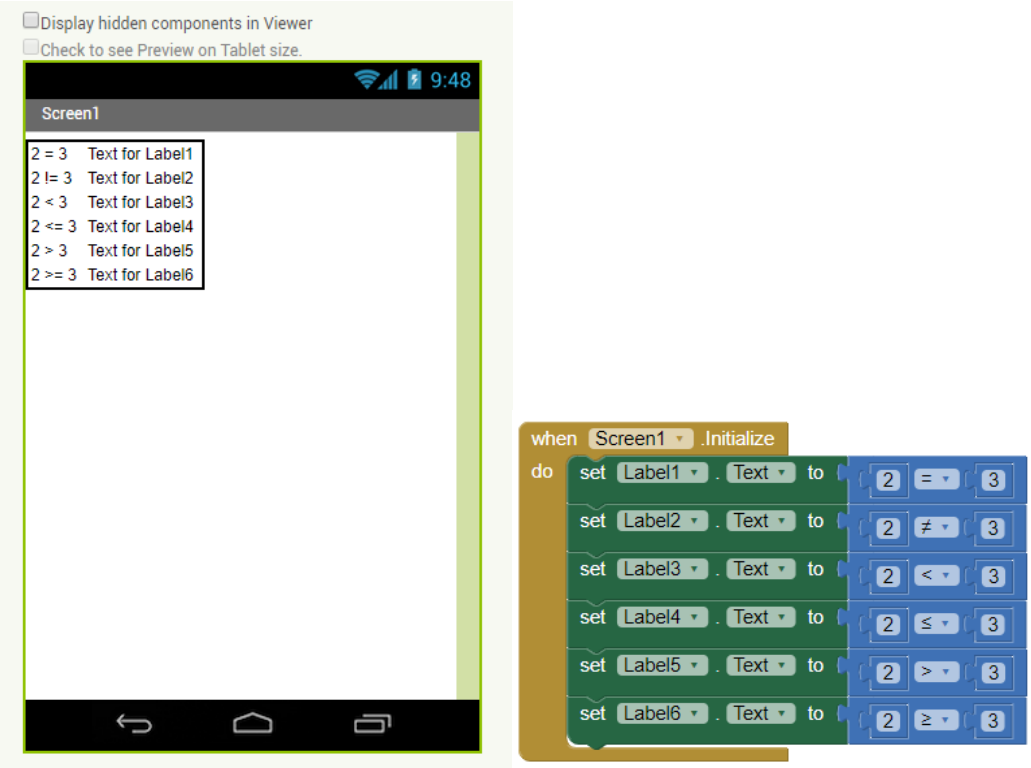
點選「=」旁的下拉是方塊可選擇其他運算子。



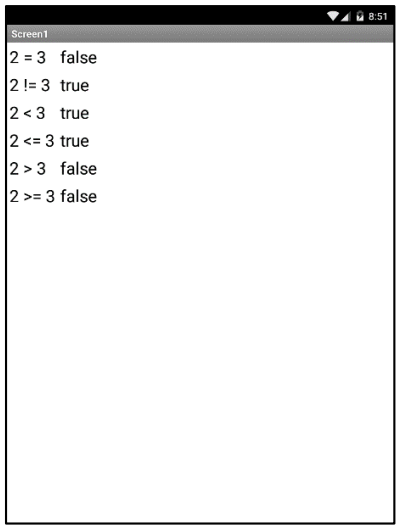
運算子	說明	範例	執行結果
=	等於		False
≠	不等於		True
<	小於		True
≦	小於等於		True
>	大於		False
≧	大於等於		False

以下例試做看看，我們先在 Screen 1 將上方表格的所有範例列出並分別給予一個 Lable，當開啟 Screen 時，將 Label 的文字改為比較運算的結果。

開啟 Screen 即執行程式的方塊為內建方塊(Blocks)項目點選 Screen1，選擇 when Screen1.lnitialize



以下是程式執行結果。



練習題

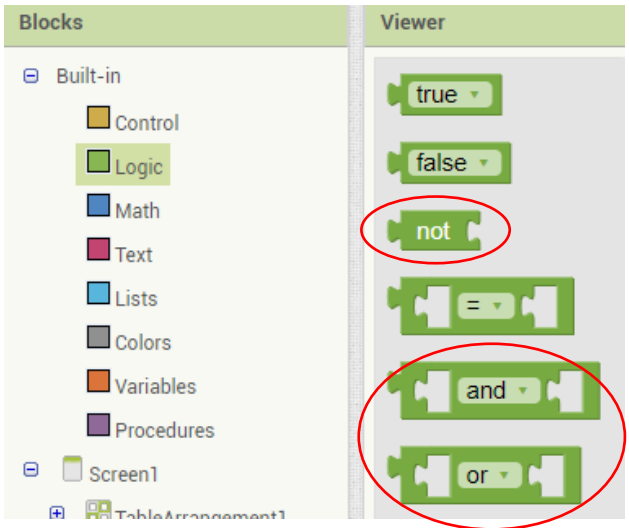
在 Screen 中，放入兩個 TextBox、一個 Label 和一個 Button。請使用者在兩個 TextBox 中分別輸入兩個數後，當按下 Button 時利用'<'比較兩數之大小，將結果顯示在 Label 中。例如：使用者分別輸入 1 和 2，當按下 Button，Label 會顯示出"1<2=false"。

邏輯運算

用於判斷兩個布林值根據 And、Or、Not 等三種運算後是否為成立的運算。結果一樣以布林值回傳。

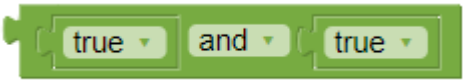
使用邏輯運算

邏輯運算是用內建方塊(Blocks)項目點選邏輯(Logic)，選擇邏輯方塊(and、or 與 not)。



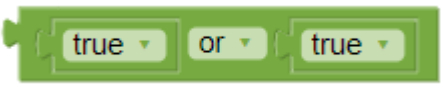
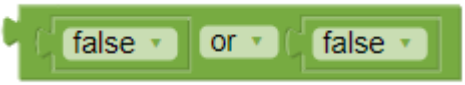
And

當兩邊的運算結果都成立（True）時，其結果也成立（True）。例如：「True」And「True」結果為 True，「True」And「False」結果為 False。

範例	執行結果
	True
	False
	False
	False

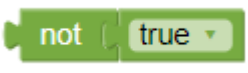
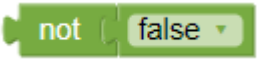
Or

只要兩邊的運算結果有一邊成立時，其結果就成立。例如：「True」And「True」結果為 True，「True」And「False」結果為 True。

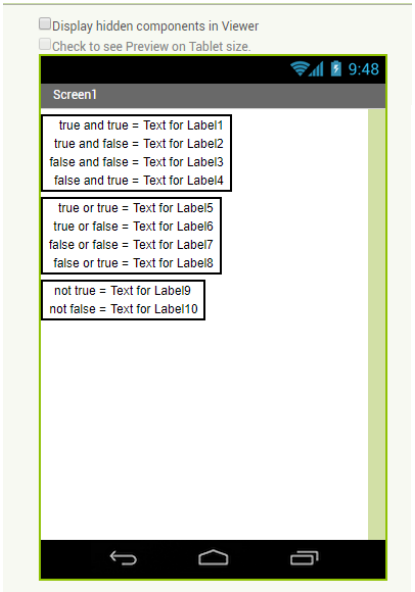
範例	執行結果
	True
	True
	False
	True

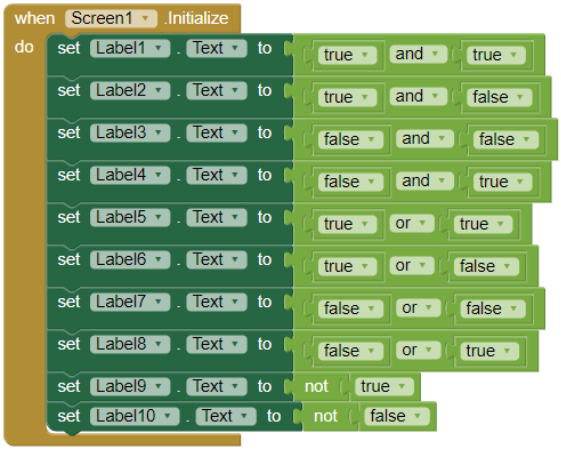
Not

此運算方式與上述兩種不同式直接將原本的運算結果顛倒，不需要兩個運算結果也可以執行。

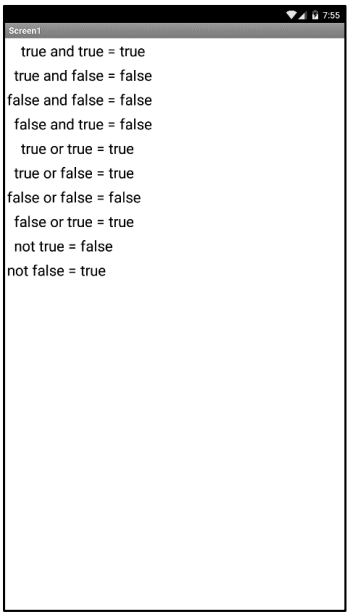
範例	執行結果
	False
	True

以下例試做看看，我們先在 Screen 1 將所有的邏輯運算列出並分別給予一個 Lable，當開啟 Screen 時，將 Label 的文字改為邏輯運算的結果。





下圖是程式執行結果



數學運算

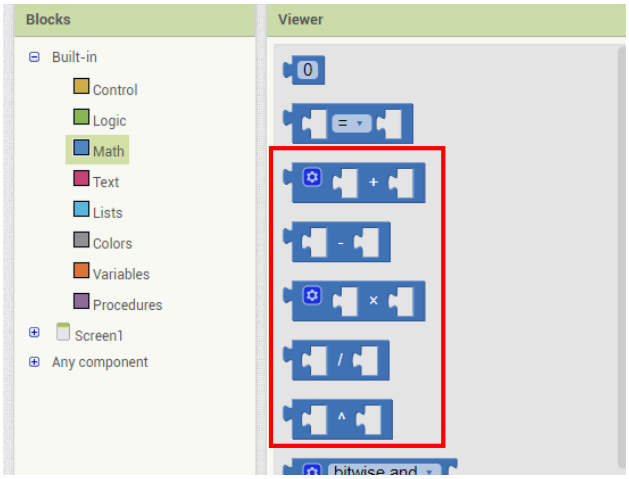
一般程式都只會提供基本的運算子，而其他運算公式則要由設計者自行輸入。基本的運算子符號有加「＋」、減「－」、乘「＊」、除「／」、指數與平方根「^」(X 的平方為 X^2，X 的平方根為 X^(1/2))。另外還有比較最大值「max」、最小值「min」、三角函數...等運算。

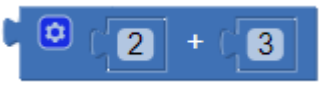
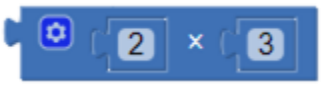
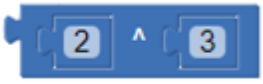
電腦的數學運算規則基本上與四則運算相同，先乘除後加減有括號要先算、由左而右計算，只是在同時做乘、除、指數運算時會依照指數 > 乘 = 除的順序計算。同時進行比較、邏輯、數學運算時會以數學 > 比較 > 邏輯的順序運算。

例如：2^3*4（＝8*4＝32）與 4*2^3（＝4*8＝32）結果是相同的，而不會變成 4*2^3（＝8^3＝512）。

使用數學運算

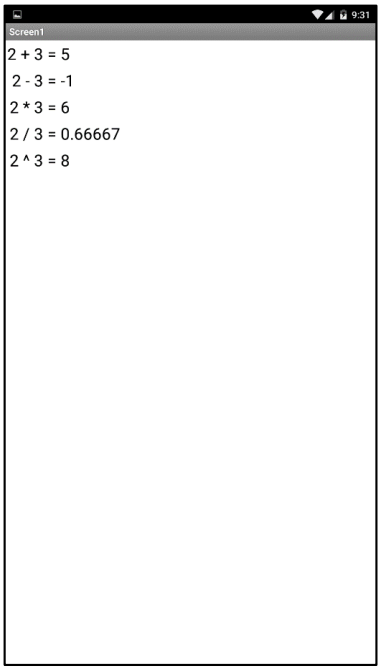
數學運算是用內建方塊(Blocks)項目點選數學(Math)，選計算方塊（+、-、*、/、^等）。



運算子	說明	範例	執行結果
+	加		5
-	減		-1
×	乘		6
/	除		0.66667
^	指數		8

以下例試做看看，先在 Screen 1 將上方表格的基本運算子列出並分別給予一個 Lable，當開啟 Screen 時，將 Label 的文字改為邏輯運算的結果。

以下是程式執行結果



練習題

在 Screen 中，放入兩個 TextBox、一個 Label 和四個分別為 + - * / 的 Button，請使用者在兩個 TextBox 中分別輸入兩個數後，當分別按下 + - * / 的 Button 時，將兩數做運算並把結果顯示在 Label 中。