

Chap X 範例

作者：郭俊佑

記帳本範例

小郭是一個剛上大學一年級的學生，由於就讀的學校離家裡有點遠，所以必須離開家裡到外面租房子自己生活。對於小郭這個第一次自己在外生活的人而言，父母給的零用錢時常莫名其妙的減少，因此小郭需要一款 app 來記錄自己的收入和支出。這款 app 的主要需求是檢視身上剩餘的金額，可以看到總收入與總支出，並且隨時隨地都可以記錄收入或支出在 app 裡面；還可以在 app 上面檢視每日的收入與支出，以了解自己零用錢的收入與支出的情況；也能在想要清空所有記錄，重新開始做記錄的時候將以前儲存的歷史記錄刪除，總結以上我們可將 app 的需求整理成列表如下：

	需求名稱	需求說明	對應課程
需求 1	顯示當天日期	可供使用者確認時間	Clock
需求 2	記錄收支	可將收入和支出記錄下來	TextBox TinyDB
需求 3	顯示記錄	顯示所記錄下來的總收入、總支出、本月剩餘金額(總收入 - 總支出)	Label TinyDB
需求 4	資料清除	將收支紀錄全部清除	Button TinyDB
需求 5	畫面警示	在清除所有資料前警告使用始確認是否清除資料	Notifier
需求 6	選取日期	可讓使用者選取日期來檢視當日收支記錄	DatePicker
需求 7	檢視記錄	可供使用者檢視每日的收入和支出	List View TinyDB
需求 8	換頁	可切換頁面	Control
其餘需求	按鈕、變數	程式中所需要的資料可暫時存放於變數	Button Variables

用 App Inventor 2 設計出的程式畫面如下



輸入欄位

當按下清除鍵時將跳出確認視窗如上圖

當輸入欄未填入資料時按下收入或支出鍵將跳出提示如下圖

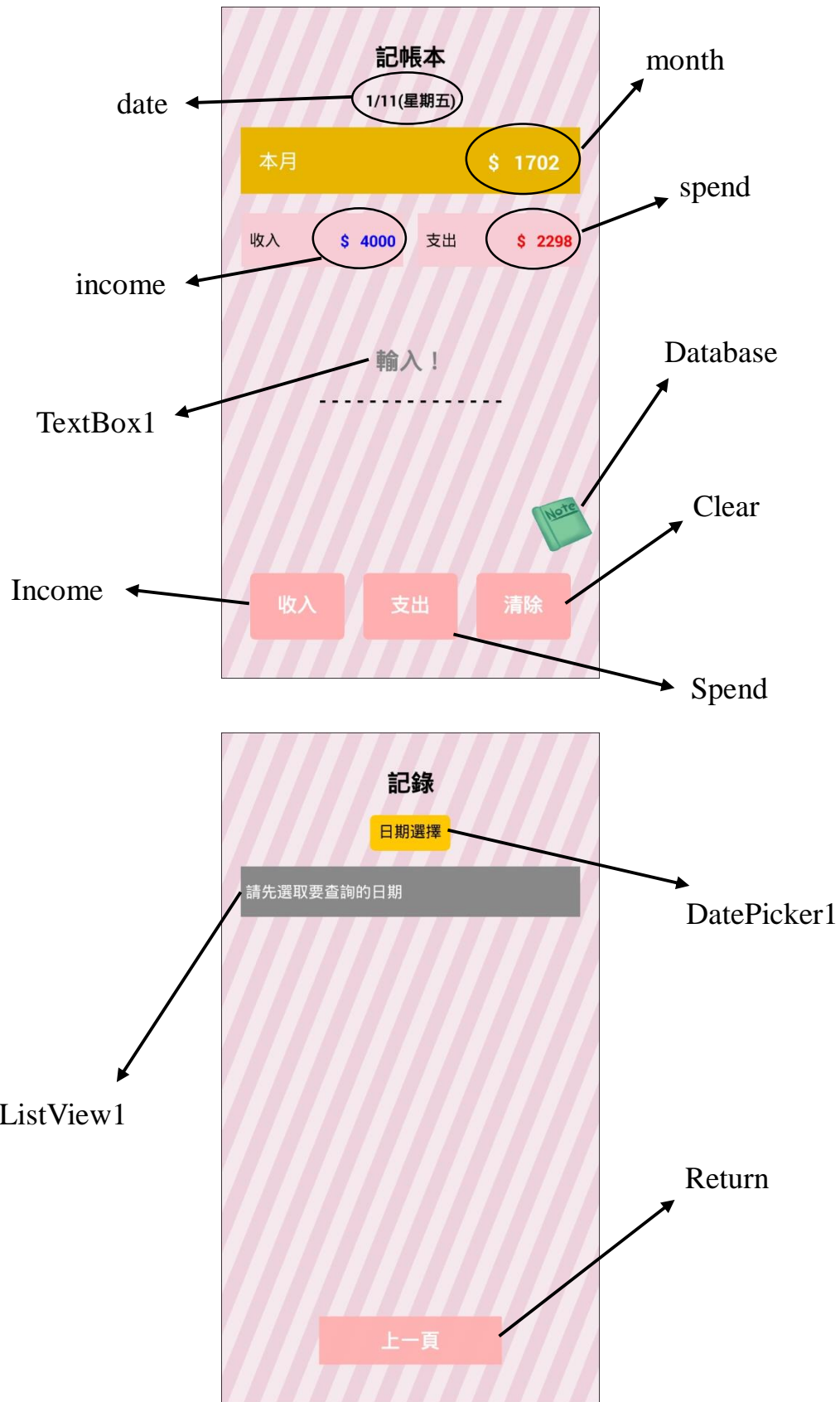


左上這張圖為程式主畫面，當輸入欄未填入資料時按下收入或支出鍵將會跳出提示。而按下清除鍵時將跳出確認視窗讓使用者確認，當使用者按下確定時 app 的內存資料將會被刪除。而在清除鍵上方的綠色書本，則是進入記錄頁面的按鈕，按下會切換到記錄的頁面。

下圖為記帳本的記錄頁面



下圖為記帳本的主畫面與記錄頁面的欄位名稱



下面為主畫面的 blocks

宣告變數是寫程式缺一不可的步驟，如同數學裡的變數可以用來解決許多題目一樣，程式裡的變數也有著異曲同工之妙。在程式設計中，我們時常需要記錄某些資料，這些資料有可能是文字，也有可能是數字，我們把這些資料記錄在記憶體某個位址中，並給他一個名稱，這就是變數。變數大致上分成兩種，一種是全域變數，一種是區域變數。全域變數就是宣告在所有函式之外，使所有的函式都可以共用這個變數；相反的，區域變數就是宣告在某些函式之中，使這個變數只能被這個函式所使用，並不能被其他函式所共同使用。

下面是整個主畫面的程式都可以使用的全域變數。在後面的 blocks 中可看到這些變數的使用方式。



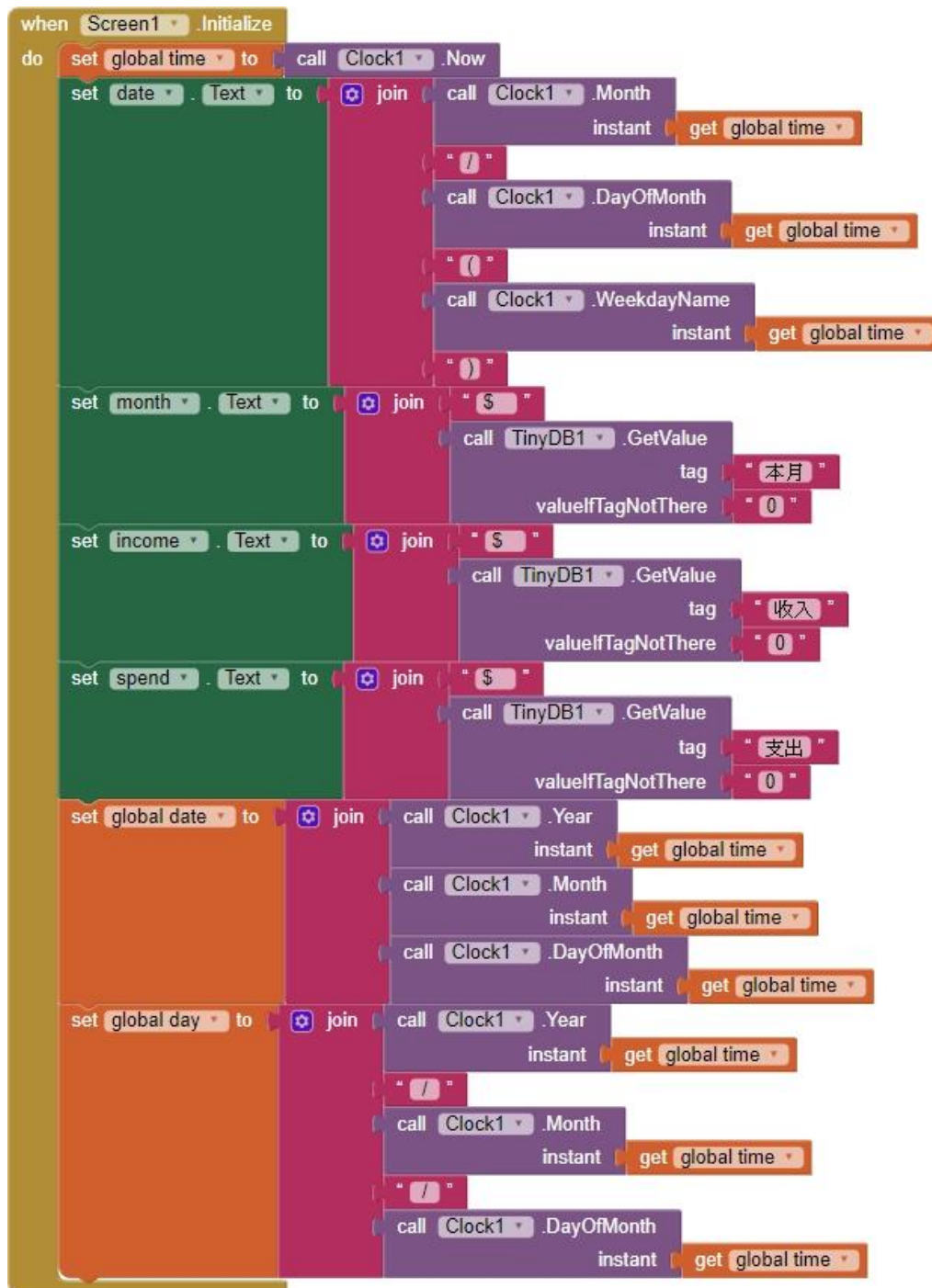
下面為主畫面綠色書本的點擊事件

下面的 Database 這個元件是在主畫面的那本綠色的書，為開啟記錄畫面的按鈕。而下面是 Database 這個元件的點擊事件：當 Database 這個元件被點擊時，將會開啟一個名叫 database 的畫面。這邊有一點要注意的是當這個名叫 database 的畫面被開啟時，主畫面並不會因此而關閉，而是留在 database 這個畫面下方，當 database 這個畫面被關閉時，主畫面將會顯示於畫面上。



下面為主畫面初始化的 blocks

初始化就是每次畫面開啟只會執行一次，所以一開始先用 `time` 這個變數來儲存手機當下的時間，然後使 `date` 顯示 "月/日(星期)" 這樣的格式。接著讓 `month` 顯示 TinyDB 中標籤為本月的資料，如果沒資料的話就回傳 0 這個值。另外兩個 `income` 和 `spend` 也做一樣的事情，只有標籤不一樣而已。最後用 `date` 和 `day` 這兩個變數記錄時間格式以便後面使用(抓的時間一樣，格式不一樣)全部執行完後，除了畫面關掉重開以外，將不會再執行第二遍。



下面為主畫面收入按鈕的點擊事件

```

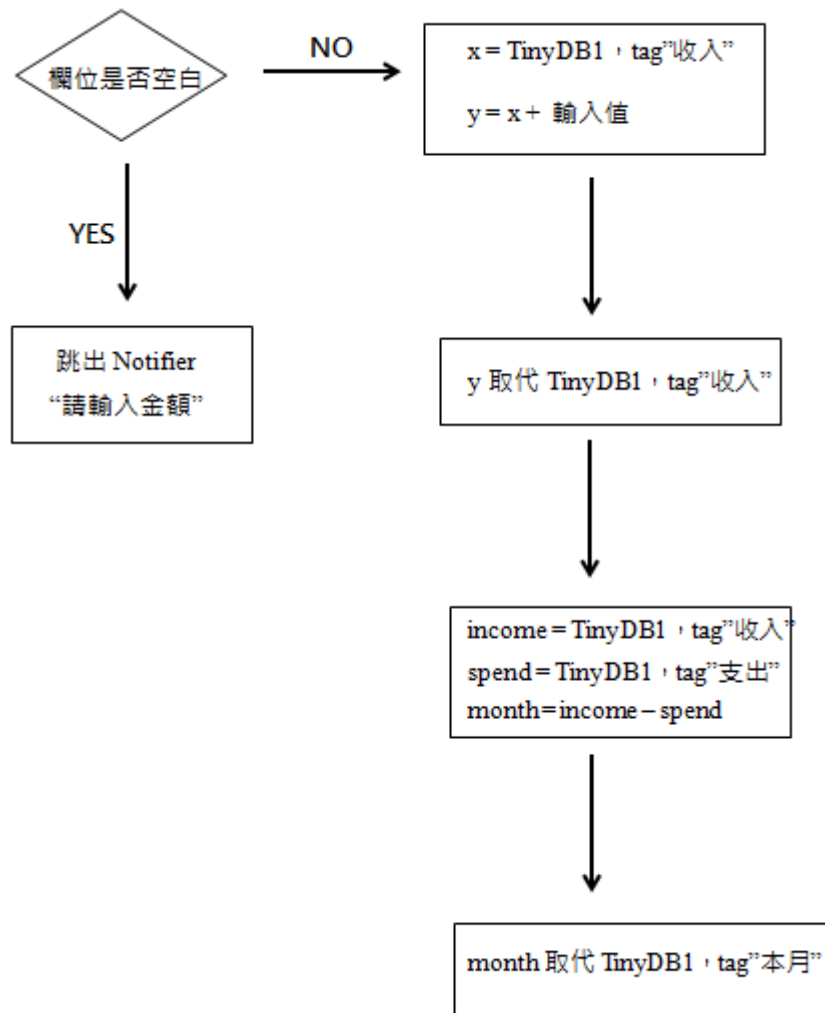
when IncomeButton.Click
do
  if TextBox1.Text = ""
  then
    call Notifier1.ShowDialog
      message "請輸入金額"
      title ""
      buttonText "確定"
  else
    set global x to call TinyDB1.GetValue
      tag "收入"
      valueIfTagNotThere "0"
    set global y to get global x + TextBox1.Text
    call TinyDB1.StoreValue
      tag "收入"
      valueToStore get global y
    set global income to call TinyDB1.GetValue
      tag "收入"
      valueIfTagNotThere "0"
    set global spend to call TinyDB1.GetValue
      tag "支出"
      valueIfTagNotThere "0"
    set global month to get global income - get global spend
    call TinyDB1.StoreValue
      tag "本月"
      valueToStore get global month
    if call TinyDB1.GetValue
      tag join get global date 1
      valueIfTagNotThere ""
    then
      set global list to 1
      call TinyDB1.StoreValue
        tag join get global date
          get global list
          get global day
          "-"
          TextBox1.Text
        valueToStore join
          get global date
          get global list
          get global day
          "-"
          TextBox1.Text
      set global list to get global list + 1
    else
      call TinyDB1.StoreValue
        tag join get global date
          get global list
          get global day
          "-"
          TextBox1.Text
        valueToStore join
          get global date
          get global list
          get global day
          "-"
          TextBox1.Text
      set global list to get global list + 1
    set month.Text to join "$"
      get global month
    set income.Text to join "$"
      get global income
    set TextBox1.Text to ""
  
```

The code block is titled "when IncomeButton.Click". It contains a "do" block with an "if" statement. The "if" statement checks if "TextBox1.Text" is equal to an empty string. If true, it calls "Notifier1.ShowDialog" with a message "請輸入金額", an empty title, and a "確定" button. If false, it performs several database operations: it gets the current "收入" (income) value, adds the input from "TextBox1", stores the new value, gets the current "支出" (expense) value, calculates the difference to get the "本月" (this month) total, and stores it. It then checks for a "global date" tag. If it exists, it updates the "global list" and stores the new date, list, and day information along with the input. If it doesn't exist, it stores the date, list, and day information along with the input. Finally, it updates the "month" and "income" labels with the current values and clears the input field.

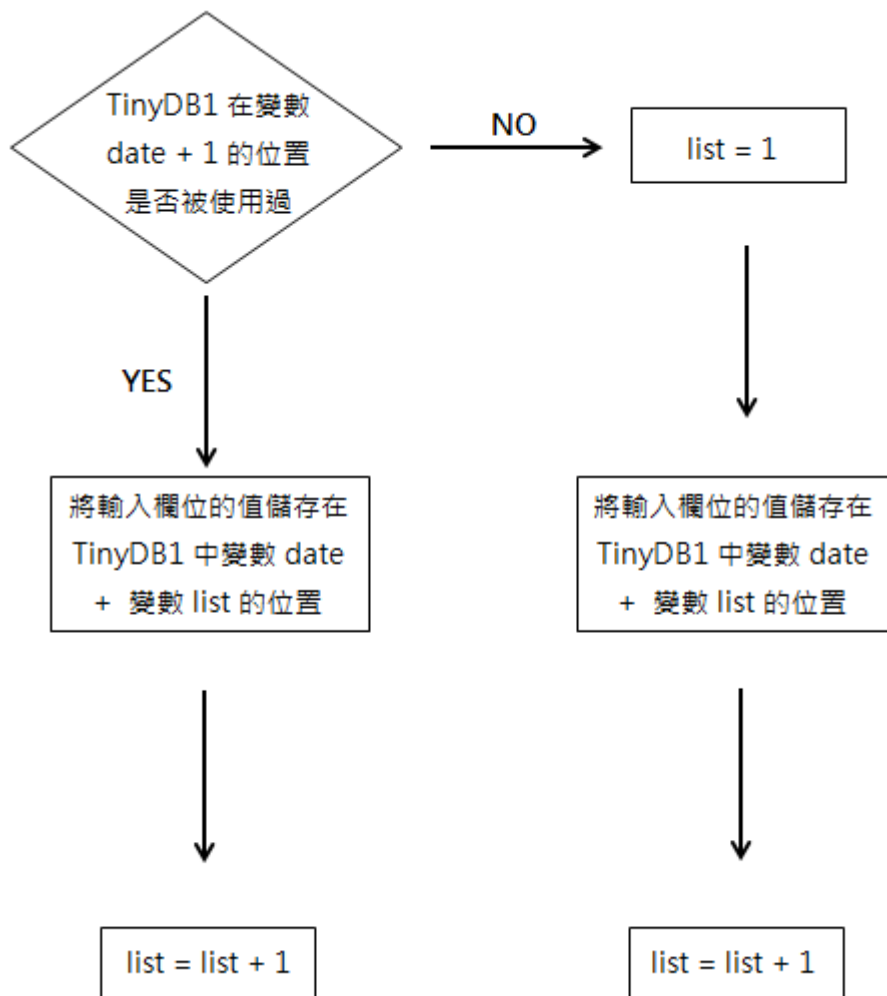
Three blue arrows point to specific parts of the code:

- [1] points to the condition `TextBox1.Text = ""`.
- [2] points to the `if` block that checks for the `global date` tag.
- [3] points to the `set month.Text` and `set income.Text` blocks.

[1] 上面的主畫面收入按鈕的點擊事件在事件一開始先加入一個 if-then-else 條件判斷式來判斷輸入欄位是否已被輸入資料。當 if 後面的條件成立時，程式將會執行 then 後面的步驟；而當 if 後面的條件不成立時，程式將會執行 else 後面的步驟。這邊在事件一開始先判斷輸入欄位是否是空白的，如果是的話跳出 Notifier 的訊息對話視窗，請使用者先輸入金額再點擊按鈕；如果不是的話，程式將會先用變數 x 儲存 TinyDB 裡標籤為 " 收入 " 的資料，再用變數 y 儲存 x 與輸入欄位相加的結果，這個結果就是新增後總收入的金額，將這個結果取代 TinyDB 裡標籤為 " 收入 " 的資料，接著使用 income 和 spend 分別取得 TinyDB 裡標籤為 " 收入 " 和 " 支出 " 的資料，並將 $income - spend$ 的值存入變數 month，這個變數 month 就是使用者剩餘的金額(收入-支出)，再將 month 取代 TinyDB 裡標籤為 " 本月 " 的資料。



[2] 接著再加入一個 if-then-else 來決定這筆資料的儲存位置。將要儲存在 TinyDB 裡的資料標籤設定為變數 date + 變數 list，變數 list 將會在儲存資料過後比原先數值增加 1，例如：date = 2019111，list = 1 那麼資料標籤將會是 20191111，儲存過後 list = 2。會這樣做的原因是 TinyDB 是以標籤來儲存資料，所以每一個標籤只能儲存一筆資料，所以設定變數 list 在儲存過後數值增加 1 可避免同一個標籤被一筆以上的資料所使用。所以在資料儲存前需要先判斷(if) TinyDB 在變數 date + 1 的位置是否有被使用過，如果沒被使用過那麼這位置會是空白的，是空白時將執行 then 後面的事件，先把變數 list 設定為 1，然後將輸入欄位的值儲存在 TinyDB 中標籤為變數 date + 變數 list 的位置，儲存過後變數 list 的值增加 1；不是空白時執行 else 後面的事件，直接將輸入欄位的值儲存在 TinyDB 中標籤為變數 date + 變數 list 的位置，儲存過後變數 list 的值增加 1。這樣的儲存方式可以保證每日資料都將於 list = 1 時開始儲存，每次 list 增加 1，這樣可便於查詢資料時所需要做的動作。



[3] 在上述的步驟都執行完後，執行畫面更新的動作，將會變動的值在畫面上做更新的動作，而這邊是增加收入的按鈕，所以在畫面上會變動的值只有收入和本月，於是更新收入與本月的顯示，最後將輸入欄位上的數字刪除，可便於使用者馬上做下一筆的新增。

下面的主畫面支出按鈕的點擊事件與上述的收入按鈕點擊事件大致相同，只是從收入改為支出，所以有一些小地方不同，例如：變數 x 取得的位置、更新 TinyDB 的位置、儲存進 TinyDB 資料的數字多了負號(-)、更新畫面時的 income 改為更新 spend。

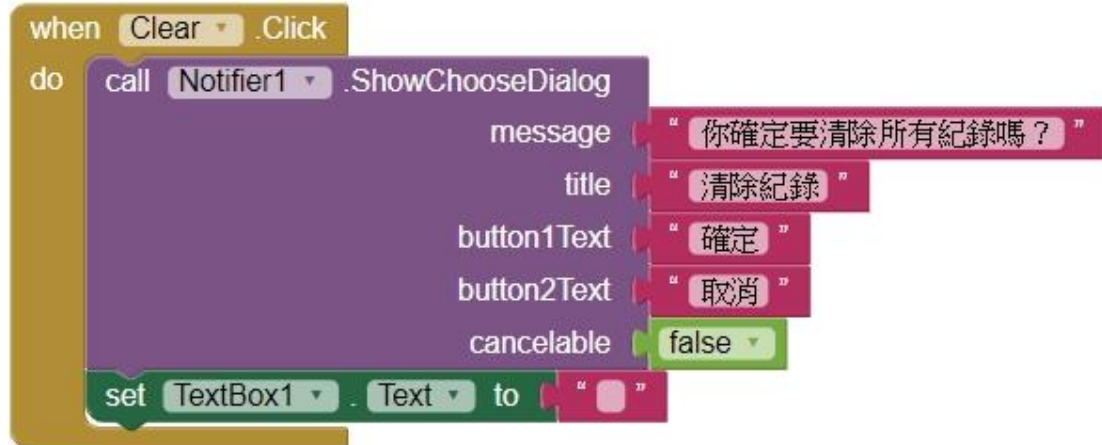
下面為主畫面支出按鈕的點擊事件

The screenshot displays the visual programming logic for the 'SpendButton' click event. The logic is structured as follows:

- when SpendButton .Click**
 - do**
 - if** `TextBox1 .Text = ""`
 - then**
 - call** `Notifier1 .ShowMessageDialog`
 - message: `請輸入金額`
 - title: `''`
 - buttonText: `確定`
 - else**
 - set** `global x` to `call TinyDB1 .GetValue`
 - tag: `支出`
 - valueIfTagNotThere: `0`
 - set** `global y` to `get global x + TextBox1 .Text`
 - call** `TinyDB1 .StoreValue`
 - tag: `支出`
 - valueToStore: `get global y`
 - set** `global income` to `call TinyDB1 .GetValue`
 - tag: `收入`
 - valueIfTagNotThere: `0`
 - set** `global spend` to `call TinyDB1 .GetValue`
 - tag: `支出`
 - valueIfTagNotThere: `0`
 - set** `global month` to `get global income - get global spend`
 - call** `TinyDB1 .StoreValue`
 - tag: `本月`
 - valueToStore: `get global month`
 - if** `call TinyDB1 .GetValue`
 - tag: `join get global date 1`
 - valueIfTagNotThere: `''`
 - then**
 - set** `global list` to `1`
 - call** `TinyDB1 .StoreValue`
 - tag: `join get global date get global list`
 - valueToStore: `join get global day '-' '-' TextBox1 .Text`
 - set** `global list` to `get global list + 1`
 - else**
 - call** `TinyDB1 .StoreValue`
 - tag: `join get global date get global list`
 - valueToStore: `join get global day '-' '-' TextBox1 .Text`
 - set** `global list` to `get global list + 1`
 - set** `month .Text` to `join '$ ' get global month`
 - set** `spend .Text` to `join '$ ' get global spend`
 - set** `TextBox1 .Text` to `''`

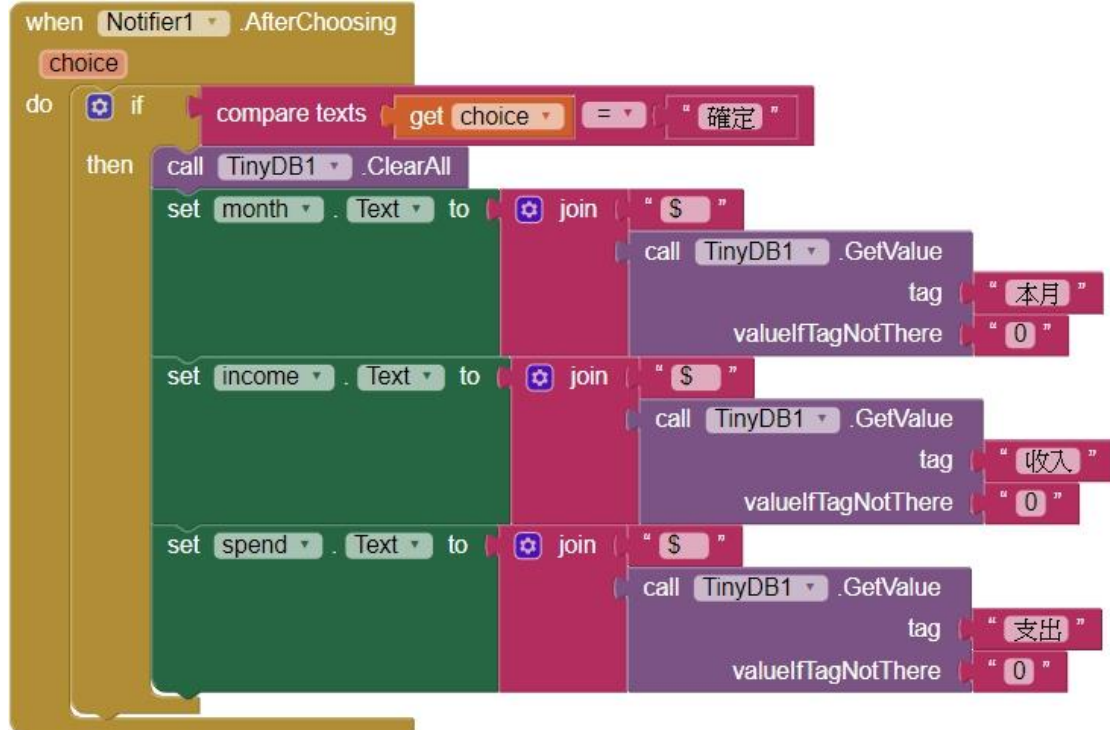
下面為主畫面清除按鈕的點擊事件

當 Clear 按鈕被點擊時，顯示 Notifier 的選擇視窗，詢問使用者是否確定清除所有紀錄，選擇視窗標題為 " 清除紀錄 "，按鈕有確認和取消。



下面為當 Notifier1 這個元件視窗跳出並選取完畢後執行的事件

首先判斷選取的按鈕文字是否為 " 確定 "，判斷按鈕文字是 " 確定 " 後就清空所有在 TinyDB 裡的資料，接著更新畫面上 month, income, spend 的顯示；判斷按鈕文字不是 " 確定 " 後將關閉選擇視窗不做任何動作。



下面為記錄頁面的 blocks

下面是記錄頁面的全域變數宣告，這邊的宣告可被所有紀錄頁面的 blocks 共同使用，並事先宣告 listview 為字串：請先選取要查詢的日期，在後面的 blocks 中可看到這些變數的使用方式。



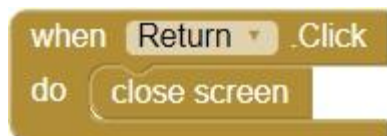
下面為記錄頁面初始化的 blocks

初始化就是每次畫面開啟只會執行一次，首先設定 DatePicker 這個日期選擇元件預設的時間為當下時間，並先將 ListView1 這個元件顯示為變數 listview。



下面為 Return 這個元件的點擊事件

當 Return 被點擊時，將這個視窗關閉。



下面為 DatePicker1 這個日期選擇元件選擇完日期的觸發事件

這邊需要先了解 TinyDB 在同一個 app 裡面所有畫面都將共用同一個 TinyDB 的儲存位置，例如我在主畫面儲存一個值在標籤 "收入" 裡，我仍然能在記錄頁面取得那個在主畫面存入的值，但是如果在記錄頁面儲存一個值進入標籤 "收入" 裡面，那麼原先在主畫面存入的值將會消失。

在 DatePicker 視窗按下完成鍵後將會執行，首先先讓 DatePicker 顯示所選擇的日期格式為(年/月/日)，接著先設定變數 date 為年+月+日，與主畫面設定的格式相同。下面加入 if-then-else 判斷 TinyDB 裡標籤為變數 date + 1 的位置是否為空白，為空白時等同沒有資料，將跳出 Notifier 裡的訊息對話視窗，告訴使用者此日期沒有資料，最後將 listview 設定為空白的名單，以刪除一開始設定的文字；當 TinyDB 裡標籤為變數 date + 1 的位置有資料時，先將 listview 設定為空白的名單以便查詢後顯示，再加入 while test-do，這邊就是當滿足上方條件時做無限迴圈的動作，當不滿足上方條件時就離開迴圈，此處先判斷 TinyDB 裡標籤為變數 date + 變數 list 的位置不是空白時，將此位置的值加入 listview 這個名單裡的最後一筆，每加入一筆資料後變數 list 增加 1，所以此迴圈將會執行到 TinyDB 裡標籤為變數 date + 變數 list 的位置沒有資料的時候，這個迴圈結束時將變數 list 設定為 1，下次執行時 list 又會從 1 開始執行。最後將 ListView 這個元件顯示出變數 listview 的這個清單。

