1-6 Timer 與 ScrollBar

1-6-1 前言

本節中首先要介紹是如何產生亂數，所謂亂數是指下一個出現的數是沒有規則性可言，無法以某一個公式準確預測。亂數在程式設計中應用領域非常廣，最常見的就是希望在程式中模擬某一個隨機變數者，例如想要模擬骰子出現的點數，就必須產生一個介於 1 到 6 的亂數。在 QB 中是以內建函數 Rnd()來產生介於 0 與 1 間的亂數，再經過適當的平移轉換產生符合個人所需的亂數範圍，而 C#中則是以 Random 類別為主，除了符合物件導向需求，使用上也較為簡單。

其次，要介紹兩種類別 Timer(計時器)與 ScrollBar(捲軸)。前者是利用電腦內部計時器所設計而成的一種進階型的計時器，經常應用於固定間隔去執行某項作業，例如圖 1-6-1 的螢幕的小時鐘，就是每隔 1 秒鐘去讀取系統時間再做適當的顯示。而 ScrollBar 一般翻譯成捲軸，分成橫式捲軸(Horizontial ScrollBar)與直式捲軸(Vertiacal ScrollBar)，常用於當資料的範圍大於物件的範圍時調整顯示區域用，如圖 1-6-2 記事本中的橫式與直式捲軸，另外也可當作一種類比式的資料輸入，如同本節中範例所呈現者。

圖 1-6-1 Timer 應用：小時鐘

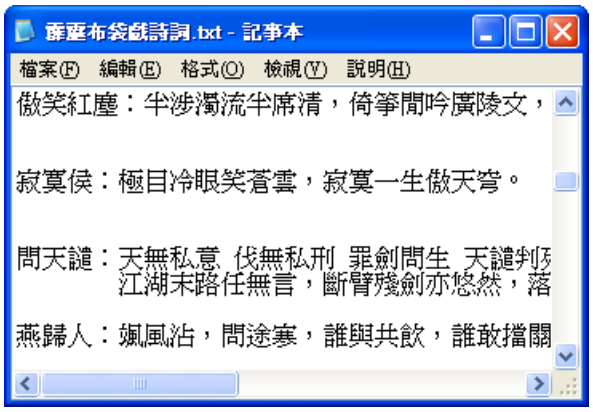

圖 1-6-2 記事本中的橫式與直式捲軸

1-6-2 學習目標

1.熟悉 Random 類別常用屬性與方法。
2.熟悉 Timer 類別常用屬性與方法。
3.熟悉 ScrollBar 類別常用的屬性、事件與方法。

1-6-3 亂數產生方式

在分析各種問題時，如果須要模擬某種不可預期且不規則的現象，例如抽籤的號碼等，

此時可以利用亂數(random number)方式產生近似這樣的數據。經由亂數產稱的數據每一次的值皆不相同（因為我們要求其具有不可預期且不規則的特性），它是由數學理論推導出的方程式來計算。我們可以將亂數依其統計分佈特性分為：均勻(uniform)亂數與常態(normal)亂數。均勻亂數是指其值平均的分佈於一區間，而常態亂數的值則是呈現高斯(Gaussian)分佈，形狀像一個中間高二頭低的山丘。下面介紹 Random 類別。

(一)、Random 類別

Random 類別的實作是以 Donald E. Knuth 的減法亂數產生器演算法為基準，所以是屬於虛擬亂數。所謂虛擬亂數是以相等的機率從有限的數字集中選取，選取的數字並非是完全隨機，因為是使用有限性數學演算法來選取它們，但是用於實際用途已足夠。如需詳細資訊，請參閱 D. E. Knuth 所著的《The Art of Computer Programming, volume 2: Seminumerical Algorithms》，Addison-Wesley, Reading, MA, second edition, 1981。

建構函數

Random 類別不在設計環境的工具箱中，所以必須自行利用 New 關鍵字呼叫建構函數方式來產生此物件。就 Random 類別而言，它的建構函數有兩種型式：

Random()	使用與時間相依的種子值來初始化 Random 類別的新執行個體，所以每次所取得的亂數都不相同。
Random(Seed As Int32)	使用指定的種子值，初始化 Random 類別的新執行個體。

這兩種建構函數最大區別在於種子值，前者是隨時間變動，後者則是固定的。由於亂數的產生始於種子值，如果重複使用相同的種子就會產生相同的連續數字。例如下面範例，AutoRand 與 AutoRand2 的種子值都是 120，所以產生的亂數序列完全相。所以若要讓不同的 Random 物件產生不同序列，就必須以第一種建構子 Random()來建立物件。

```
Random AutoRand = new Random(120);
for(int i = 1; i <= 4; i++)
{
    MessageBox.Show(Convert.ToString(AutoRand.Next(1, 6)));
}
Random AutoRand2 = new Random(120);
for (int i = 1; i <= 4; i++)
{
    MessageBox.Show(Convert.ToString(AutoRand2.Next(1, 6)));
}
```

常用方法

Next	傳回一個亂數，多型，有 3 種型式： Next()：傳回小於 2^{31} - 1 的非負值的整數亂數。

	Next(maxValue As Int32)：傳回小於指定最大值的非負值整數亂數，所以傳回亂數的最大值是 maxValue-1 Next(minValue As Int32, maxValue As Int32)：傳回大於或等於 minValue，並且小於 maxValue 的整數亂數;也就是說,傳回值的範圍包含 minValue 但不包含 maxValue。例如若 AutoRand 為 Random 物件，則 AutoRand.Next(6,8)只有兩種可能亂數 6 或 7。
NextDouble	：傳回大於或等於 0.0 且小於 1.0 之間的亂數,資料型態為雙精度浮點數,例如 randObj.NextDouble()。

1-6-4Timer 與 ScrollBar

(一)、Timer

功能

提供一套機制，可於指定間隔執行指定作業。

建構函數

Timer()	建立一個 Timer 類別的新物件。例如 Timer Timer1 = new Timer();會產生一個名為 Timer1 的 Timer 物件。

常用屬性

Enabled	計時器的啟動或停止
Interval	計時的時間，單位為千分之一秒

常用事件

Tick	計時終了時觸發的事件。

(二)、ScrollBar

功能

具有滑動 Thumb 的捲軸，其位置會與值對應。

建構函數

HScrollBar()	建立一個新的橫式捲軸物件，例如 HScrollBar HS = new HScrollBar();會產生一個名為 HS 的橫式捲軸物件。
VScrollBar()	建立一個新的直式捲軸物件，例如 VScrollBar VS = new VScrollBar();會產生一個名為 VS 的直式捲軸物件。

重要屬性

Value	取得捲軸上的捲動方塊目前位置所代表的數值，或移動捲動方塊到指定的數值。
Minimum	橫式捲軸方塊移動到最左邊時 value 屬性的值，或直式捲軸方塊移動到最上方時 value 屬性的值。
Maxmum	橫式捲軸方塊移動到最右邊時 value 屬性的值，或直式捲軸方塊移動到最下方時 value 屬性的值。
SmallChange	按下橫式捲軸左(右)微調鍵時 value 屬性要減少(增加)的值，或按下直式捲軸上(下)微調鍵時 value 屬性要減少(增加)的值。
LargeChange	按下橫式捲軸的捲動方塊左(右)邊空白處時 value 屬性要減少(增加)的值，或按下直式捲軸的捲動方塊上(下)方空白處時 value 屬性要減少(增加)的值，

重要事件

Scroll	發生於捲動方塊已經由滑鼠或鍵盤動作移動時。
ValueChanged	發生於使用 Scroll 事件後、或程式變更 Value 屬性時。

1-6-5 程式範例

(一)、幸運 7

1.執行畫面

圖 1-6-3

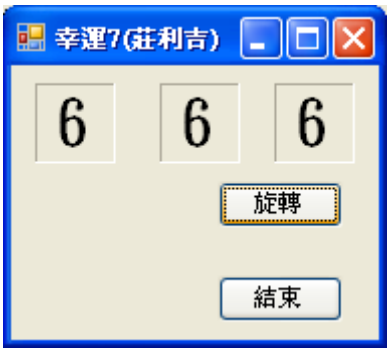

圖 1-6-4

程式功能：

1.按下旋轉按鈕時會產生 3 個亂數顯示於螢幕上，如果 3 個數都是 7 則顯示錢幣的圖片，否則不顯示圖片。

2.亂數值不是 6 就是 7。

2.物件名稱說明

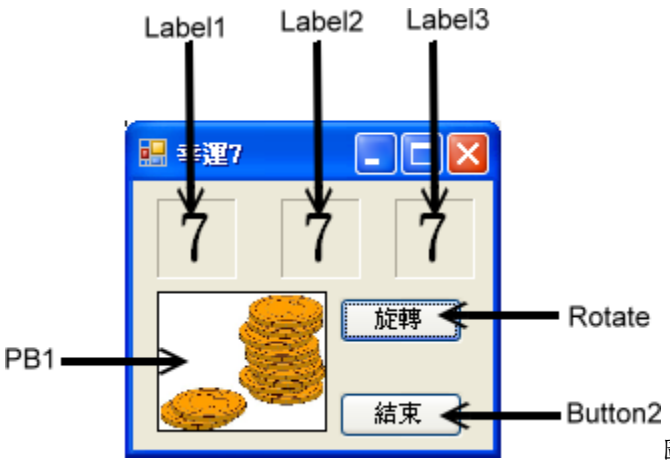

圖 1-6-5

3.演算法說明

(1) 將錢幣圖片加入為專案資源，並顯示於 PB1 上。

(2) 在 Rotate 的 Click 事件中完成下列事項

Step1：隱藏 PB1 使得錢幣圖片消失不見

Step2：產生 3 個不是 6 就是 7 的亂數整數分別顯示於 Label1、Label2、Label3 中。

Step3：檢查 3 個 Label 的 Text 是否同時為 7，若是則讓 PB1 重新顯現。

4.程式說明

```
namespace 幸運7
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            label1.AutoSize = false;
            label1.BorderStyle = BorderStyle.Fixed3D;
            label1.Size = new Size(100, 100);
            label1.Text = "7";
            label1.TextAlign = ContentAlignment.MiddleRight;
            label1.Font = new Font("微軟正黑體", 60, FontStyle.Regular);
            PB1.Image = Properties.Resources.硬幣;
            PB1.SizeMode = PictureBoxSizeMode.StretchImage;
        }

        private void Rotate_Click(object sender, EventArgs e)
        {
            PB1.Visible = false;
            Random rRand = new Random();
            label1.Text = Convert.ToString(rRand.Next(6, 8));
            label2.Text = Convert.ToString(rRand.Next(6, 8));
            label3.Text = Convert.ToString(rRand.Next(6, 8));
            if (((label1.Text == "7") && (label2.Text == "7")) && (label3.Text ==
"7"))PB1.Visible = true;
        }
```

```
        private void button2_Click(object sender, EventArgs e)
        {
            this.Close();
        }
}
}
```

(二)、旋轉幸運 7

1.執行畫面

圖 1-6-6

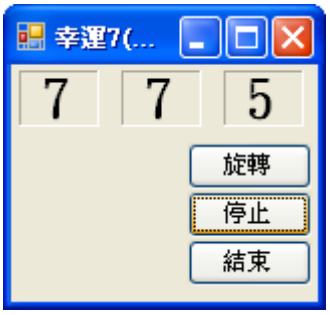

圖 1-6-7

程式功能：大致與上一個範例相同，但螢幕上的數字會不斷變換，直到按下停止按鈕時才會停止。

2.物件名稱說明

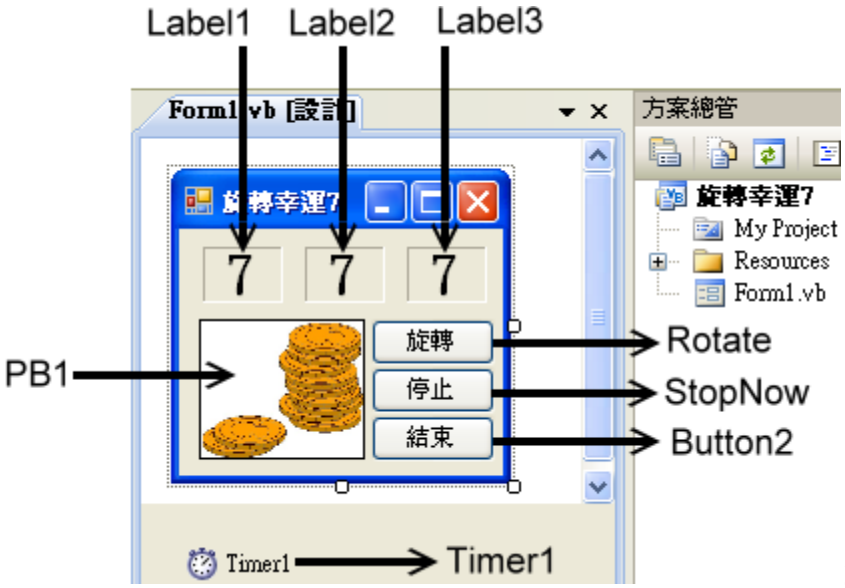

圖 1-6-8

3.演算法

(1) 將錢幣圖片加入為專案資源，並顯示於 PB1 上。

(2) 程式開始時先將 Timer1.Enabled 設為 False，關閉計時器，直到使用者按下 Roate 按鍵才啟動計時。

(3) 所以在 Rotate 的 Click 事件中應隱藏 PB1 然後啟動計時器。

(4) 在 Timer1 的 Tick 事件中產生 3 個亂數顯示於 Label1 ~ Label3。由於計時器每次計時完畢都會產生 Tick 事件，所以程式每隔一段時間就會顯示 3 個新亂數，讓使用者覺

得數字好像在跳動般。

(4) 當 StopNow 被按下時先關閉計時器，然後檢查 3 個 Label 上的數字是否都為 7，如果是則讓 PB1 從新顯現。

4.程式說明

```
namespace 幸運旋轉7
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            label1.AutoSize = false;
            label1.BorderStyle = BorderStyle.Fixed3D;
            label1.Size = new Size(100, 100);
            label1.Text = "7";
            label1.TextAlign = ContentAlignment.MiddleRight;
            label1.Font = new Font("微軟正黑體", 60, FontStyle.Regular);
            PB1.Image = Properties.Resources.硬幣;
            PB1.SizeMode = PictureBoxSizeMode.StretchImage;
            timer1.Enabled = false;
            timer1.Interval = 1000;
        }

        private void Rotate_Click(object sender, EventArgs e)
        {
            timer1.Enabled = true;
            PB1.Visible = false;
        }

        private void StopNow_Click(object sender, EventArgs e)
        {
            timer1.Enabled = false;
            if (((label1.Text == "7") && (label2.Text == "7")) && (label3.Text == "7"))
PB1.Visible = true;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            Random rRand = new Random();
            label1.Text = Convert.ToString(rRand.Next(5, 8));
            label2.Text = Convert.ToString(rRand.Next(5, 8));
            label3.Text = Convert.ToString(rRand.Next(5, 8));
        }
    }
```

```
}
```

(三)、可調轉速的幸運 7

1.執行畫面

圖 1-6-9

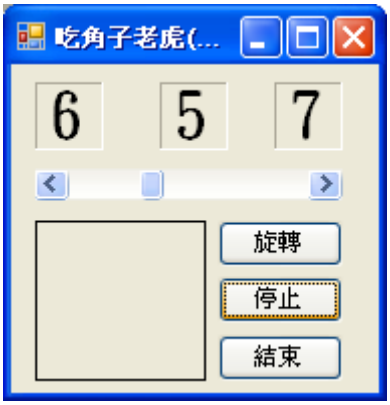

圖 1-6-10

程式功能：

1.大致與上一個範例相同，但提供捲軸讓使用者可調整螢幕上數字變換的速度。

2.方塊在捲軸最左邊時，數字變換速度最快。

2.物件名稱說明

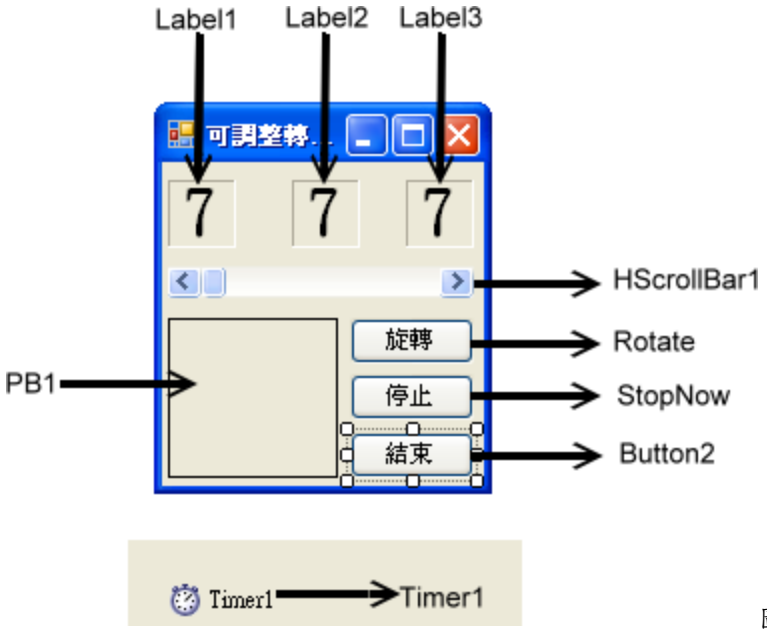

圖 1-6-11

3.演算法

(1) 變換圖片及停止後判斷 3 個數字是否相同的方法與上一個範例"旋轉幸運 7"的演算法完全一樣，不再復述。

(2) 要讓使用者利用捲軸調整圖片變換速度，最簡單的作法就是把捲軸的 Value 屬性作為 Timer1 的計時時間，只要計時時間改變，圖片變換速度自然跟著改變。

4.程式說明

```
namespace 可調轉速的幸運7
```

```
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            //設定捲軸各項參數值，由於捲軸value屬性值將作為計時的時間，所以等同決定數字變換的速度
            hScrollBar1.Minimum = 100;  //捲軸方塊在最左端時，代表值為100
            hScrollBar1.Maximum = 2000;  //捲軸方塊在最右端時，代表值為2000
            hScrollBar1.SmallChange = 10;  //按下左右兩端漸增或漸減圖示時，會將代表值增加或減少10
            hScrollBar1.LargeChange = 100;  //按下捲軸空白處時代表值會增加或減少100
            hScrollBar1.Value = 500;  //捲軸方塊預設值為500
            timer1.Interval =hScrollBar1.Value;  //數字變換間隔預設為0.5秒
            timer1.Enabled = false;
        }
        private void Rotate_Click(object sender, EventArgs e)
        {
            timer1.Enabled = true;
            PB1.Visible = false;
        }

        private void StopNow_Click(object sender, EventArgs e)
        {
            timer1.Enabled = false;
            if (((label1.Text == "7") && (label2.Text == "7")) && (label3.Text == "7")) PB1.Visible = true;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            Random rRand = new Random();
            label1.Text = Convert.ToString(rRand.Next(5, 8));
            label2.Text = Convert.ToString(rRand.Next(5, 8));
            label3.Text = Convert.ToString(rRand.Next(5, 8));
        }

        private void hScrollBar1_Scroll(object sender, ScrollEventArgs e)
        {
            if (timer1.Enabled == true)
            {
                timer1.Enabled = false;
                timer1.Interval = hScrollBar1.Value;
                timer1.Enabled = true;
            }
```

```
            else timer1.Interval = hScrollBar1.Value;
        }
    }
    }
```

1-6-6 習題

(一)、水果盤

1.執行畫面

圖 1-6-12 起始時

圖 1-6-13 按下旋轉

圖 1-6-14 按下停止

2.說明

1.這是模擬遊戲機上常見的變動水果盤。

2.設計提示：

(1).將顯示數字的 Label 換為 PictureBox。

(2).自行決定將每一個圖片對應到一個亂數，當產生新亂數時根據亂數值來顯示圖片。

(3).要顯示不同圖片，可參考 1-5 節將所需的圖片加入專案資源，然後利用[PictureBox 物件名稱.Image = My.Resources..圖片名稱]來開設定 PictureBox 上的圖片。

(4).其餘與範例相同。

(二)、賓果遊戲

設計一個賓果遊戲，遊戲規則與使用者介面請自行定義。