

# 本書目錄

## 第 01 章 最夯的程式語言－Visual C++ 2008

<b>1.1 認識 Microsoft .NET</b> .....	<b>1-2</b>
1.1.1 Microsoft .NET 特點 .....	1-2
1.1.2 Microsoft .NET 支援的語言 .....	1-4
1.1.3 通用語言執行環境 .....	1-5
<b>1.2 安裝 Visual C++ 2008 Express</b> .....	<b>1-7</b>
<b>1.3 應用程式範本</b> .....	<b>1-10</b>
1.3.1 Windows Form 應用程式.....	1-10
1.3.2 Win32 主控台應用程式.....	1-11
1.3.3 CLR 主控台應用程式.....	1-13
<b>1.4 操作環境的設定</b> .....	<b>1-14</b>
1.4.1 改變方案總管視窗的位置.....	1-14
1.4.2 自動全部隱藏視窗 .....	1-15
1.4.3 重設視窗配置.....	1-15
1.4.4 程式加上行號.....	1-16
1.4.5 變更專案儲存目錄 .....	1-16
<b>1.5 第一個 Win32 主控台應用程式</b> .....	<b>1-17</b>
1.5.1 std::cin、std::cout、std::endl、system("pause").....	1-17
1.5.2 程式註解： .....	1-18
1.5.3 建立 Win32 主控台應用程式專案 .....	1-19
1.5.4 儲存及關閉專案(方案) .....	1-22
1.5.5 開啟專案.....	1-23
<b>1.6 Win32 主控台應用程式資料輸出與輸入</b> .....	<b>1-24</b>
<b>1.7 第一個 CLR 主控台應用程式</b> .....	<b>1-26</b>
1.7.1 Write、WriteLine、Read、ReadLine 方法.....	1-26
1.7.2 建立、執行 CLR 專案.....	1-27
<b>1.8 CLR 主控台應用程式資料輸出與輸入</b> .....	<b>1-29</b>

習作園地 .....	1-31
------------	------

## 第 02 章 資料型別與運算子

<b>2.1 變數</b> .....	<b>2-2</b>
2.1.1 宣告變數 .....	2-2
2.1.2 命名規則 .....	2-3
<b>2.2 資料型別</b> .....	<b>2-4</b>
2.2.1 數值型別 .....	2-4
2.2.2 字元型別 char、wchar_t .....	2-5
2.2.3 字串型別 (string) .....	2-6
2.2.4 日期時間型別 (time_t) .....	2-15
2.2.5 布林型別 (bool) .....	2-17
2.2.6 未設定初始值的變數 .....	2-17
<b>2.3 型別轉換</b> .....	<b>2-18</b>
2.3.1 型別自動轉換 .....	2-18
2.3.2 隱含轉換 (implicit conversion) .....	2-18
2.3.3 強制轉換 (explicit conversion) .....	2-18
2.3.4 字串、數值型別轉換 .....	2-19
2.3.5 列管程式碼 (Managed) 資料型別轉換 .....	2-19
<b>2.4 常數</b> .....	<b>2-21</b>
2.4.1 以 #define 自訂常數 .....	2-21
2.4.2 以 const 自訂常數 .....	2-21
2.4.3 列舉常數 .....	2-22
<b>2.5 運算子</b> .....	<b>2-25</b>
2.5.1 算術運算子 .....	2-25
2.5.2 比較運算子 .....	2-26
2.5.3 邏輯運算子 .....	2-27
2.5.4 複合指定運算子 .....	2-28
2.5.5 位元運算子 .....	2-28

2.5.6 sizeof 型別資訊運算子.....	2-29
2.5.7 運算子的優先順序.....	2-29
<b>習作園地.....</b>	<b>2-31</b>

## 第 03 章 流程控制

<b>3.1 流程控制的認識.....</b>	<b>3-2</b>
<b>3.2 跳躍指令 goto.....</b>	<b>3-3</b>
<b>3.3 單向選擇 if.....</b>	<b>3-4</b>
<b>3.4 雙向選擇 if...else.....</b>	<b>3-9</b>
<b>3.5 多向選擇 .....</b>	<b>3-12</b>
3.5.1 if.....else if .....	3-12
3.5.2 switch...case .....	3-15
<b>3.6 重複結構 .....</b>	<b>3-20</b>
3.6.1 for...迴圈 .....	3-20
3.6.2 巢狀 for...迴圈.....	3-24
3.6.3 for...迴圈易混淆的觀念.....	3-25
3.6.4 for each...迴圈 .....	3-27
3.6.5 for(;;) 無限迴圈 .....	3-31
3.6.6 while 前測試迴圈.....	3-33
3.6.7 do... while...後測試迴圈.....	3-34
3.6.8 continue、break 的使用 .....	3-36
<b>3.7 try...catch...finally 錯誤處理.....</b>	<b>3-38</b>
3.7.1 C++ 結構化錯誤處理 .....	3-38
3.7.2 C++/CLR 結構化錯誤處理.....	3-42
3.7.3 自訂錯誤處理.....	3-44
<b>習作園地.....</b>	<b>3-46</b>

## 第 04 章 陣列

<b>4.1 一維陣列 .....</b>	<b>4-2</b>
-----------------------	------------

4.1.1 陣列的意義 .....	4-2
4.1.2 一維陣列宣告 .....	4-2
4.1.3 一維陣列初值設定 .....	4-3
4.1.4 使用迴圈顯示陣列 .....	4-6
4.1.5 一維陣列空間大小 .....	4-6
4.1.6 陣列的應用：泡沫排序 .....	4-7
4.1.7 陣列的應用：搜尋 .....	4-10
<b>4.2 一維字元陣列 .....</b>	<b>4-14</b>
<b>4.3 CLR 一維陣列 (Managed) .....</b>	<b>4-17</b>
4.3.1 CLR 整數一維陣列 .....	4-17
4.3.2 CLR 字元一維陣列 .....	4-17
4.3.3 CLR 字串一維陣列 .....	4-18
4.3.4 CLR 陣列初值設定 .....	4-19
4.3.5 CLR 一維陣列空間大小 .....	4-20
4.3.6 CLR 泡沫排序 .....	4-20
4.3.7 CLR 循序搜尋 .....	4-20
4.3.8 CLR 二分搜尋 .....	4-21
4.3.9 CLR 字元陣列的存取 .....	4-21
<b>4.4 多維陣列 .....</b>	<b>4-24</b>
4.4.1 二維陣列宣告 .....	4-24
4.4.2 二維陣列初值設定 .....	4-26
4.4.3 二維字元陣列 .....	4-29
<b>4.5 CLR 多維陣列 .....</b>	<b>4-30</b>
4.5.1 CLR 整數二維陣列 .....	4-30
4.5.2 CLR 字元二維陣列 .....	4-31
4.5.3 CLR 字串二維陣列 .....	4-32
<b>4.6 Array 類別 .....</b>	<b>4-35</b>
<b>4.7 ArrayList 類別 .....</b>	<b>4-39</b>
<b>習作園地 .....</b>	<b>4-43</b>

## 第 05 章 指標

---

<b>5.1 何謂指標</b> .....	<b>5-2</b>
5.1.1 「&」取址運算子.....	5-2
5.1.2 指標變數宣告.....	5-3
5.1.3 指標的使用 .....	5-4
5.1.4 new、delete 運算子 .....	5-5
5.1.5 指標的指標 .....	5-8
5.1.6 結構 (struct).....	5-10
<b>5.2 一維陣列與指標</b> .....	<b>5-12</b>
5.2.1 一維陣列與指標的存取 .....	5-12
5.2.2 一維字元陣列與指標.....	5-15
<b>5.3 二維陣列與指標</b> .....	<b>5-20</b>
5.3.1 二維陣列與指標的存取 .....	5-20
5.3.2 二維字元陣列與指標.....	5-24
<b>5.4 CLR 動態陣列</b> .....	<b>5-29</b>
習作園地.....	<b>5-32</b>

## 第 06 章 函式

---

<b>6.1 函式</b> .....	<b>6-2</b>
<b>6.2 建立函式</b> .....	<b>6-3</b>
6.2.1 如何建立函式.....	6-3
6.2.2 如何呼叫函式.....	6-4
6.2.3 函式原型.....	6-7
6.2.4 _tmain() 函式中的參數 .....	6-8
<b>6.3 return 返回呼叫函式</b> .....	<b>6-13</b>
<b>6.4 參數</b> .....	<b>6-15</b>
6.4.1 傳值呼叫 ( call by value ) .....	6-15
6.4.2 傳參考呼叫 ( call by reference ) .....	6-17
6.4.3 傳址呼叫 ( call by address ) .....	6-20

6.4.4 以陣列為參數傳遞 .....	6-22
<b>6.5 遞迴 (Recursive) .....</b>	<b>6-25</b>
<b>6.6 多載 (overloading) .....</b>	<b>6-27</b>
<b>6.7 變數存取範圍 .....</b>	<b>6-29</b>
6.7.1 區塊變數 (Block Variable) .....	6-29
6.7.2 區域變數 (Local Variable) .....	6-29
6.7.3 全域變數 (Global Variable) .....	6-29
6.7.4 綜合演練：區塊、區域及全域變數存取範圍 .....	6-30
6.7.5 靜態變數 (Static Variable) .....	6-31
<b>6.8 內建函式 .....</b>	<b>6-33</b>
6.8.1 亂數函式 rand() .....	6-33
6.8.2 亂數類別 (Random Class) - CLR 模式 .....	6-36
6.8.3 Math 數學函式 .....	6-38
6.8.4 Math 數學函式 - CLR 模式 .....	6-40
6.8.5 char 字元陣列函式 .....	6-43
6.8.6 日期時間函式 .....	6-46
6.8.7 DateTime 日期時間類別 .....	6-50
<b>6.9 資料型別轉換 .....</b>	<b>6-57</b>
6.9.1 型別自動轉換 .....	6-57
6.9.2 隱含轉換 (implicit conversion) .....	6-57
6.9.3 強制轉換 (explicit conversion) .....	6-57
6.9.4 字串、數值型別轉換 .....	6-58
<b>6.10 列管程式碼 (Managed) 資料型別轉換 .....</b>	<b>6-59</b>
6.10.1 Convert 類別 .....	6-59
6.10.2 ToString() .....	6-59
6.10.3 Parse() .....	6-59
6.10.4 Boxing、UnBoxing .....	6-59
6.10.5 綜合演練：CLR 資料型別轉換 .....	6-60
<b>6.11 資料輸出格式化 .....</b>	<b>6-62</b>

6.11.1 數字格式化 .....	6-62
6.11.2 日期格式化 .....	6-65
<b>習作園地.....</b>	<b>6-68</b>

## 第 07 章 物件與類別

<b>7.1 物件與類別.....</b>	<b>7-2</b>
7.1.1 物件導向程式設計 .....	7-2
7.1.2 物件與類別的觀念 .....	7-2
7.1.3 C++ 定義類別 .....	7-4
7.1.4 CLR 定義類別.....	7-7
7.1.5 public 欄位的缺點.....	7-13
<b>7.2 屬性 (property).....</b>	<b>7-15</b>
7.2.1 唯讀 (ReadOnly) 屬性.....	7-18
7.2.2 唯寫 (WriteOnly) 屬性.....	7-18
7.2.3 屬性存取 .....	7-18
<b>7.3 方法.....</b>	<b>7-20</b>
<b>7.4 多載.....</b>	<b>7-22</b>
7.4.1 關於多載.....	7-22
7.4.2 建構式 (Constructor) .....	7-25
7.4.3 解構式 (Destructor) .....	7-28
<b>7.5 事件.....</b>	<b>7-31</b>
7.5.1 認識事件.....	7-31
7.5.2 事件委派、宣告、觸發和連結.....	7-31
7.5.3 具名方法的委派 .....	7-32
7.5.4 事件程序處理.....	7-34
<b>7.6 靜態成員 (static).....</b>	<b>7-41</b>
<b>習作園地.....</b>	<b>7-44</b>

## 第 08 章 物件繼承與委派

<b>8.1 繼承</b> .....	<b>8-2</b>
8.1.1 建立子類別 .....	8-2
8.1.2 子類別的應用 .....	8-6
8.1.3 抽象類別 (abstract) .....	8-8
8.1.4 密封類別與密封方法 (sealed).....	8-13
8.1.5 朋友類別和朋友方法 (Friend).....	8-14
<b>8.2 多載</b> .....	<b>8-19</b>
8.2.1 不同參數的多載.....	8-19
8.2.2 保留父類別的多載方法 .....	8-21
8.2.3 建構式的多載與繼承.....	8-23
<b>8.3 覆寫</b> .....	<b>8-25</b>
8.3.1 以 virtual、override 進行覆寫.....	8-25
8.3.2 相同參數的多載 (多型) .....	8-27
<b>8.4 介面與實作</b> .....	<b>8-29</b>
8.4.1 介面宣告.....	8-29
8.4.2 介面與抽象類別的異同 .....	8-32
8.4.3 介面應用.....	8-33
<b>8.5 委派</b> .....	<b>8-36</b>
<b>8.6 綜合練習</b> .....	<b>8-40</b>
習作園地 .....	<b>8-46</b>

## 第 09 章 視窗程式、表單介面設計、常用控制項

<b>9.1 Windows Form 應用程式</b> .....	<b>9-2</b>
9.1.1 Windows Form 簡介.....	9-2
9.1.2 建立 Windows Form 應用程式專案 .....	9-3
<b>9.2 VC++ 2008 Express 整合開發環境操作技巧</b> .....	<b>9-4</b>
9.2.1 控制項 .....	9-4
9.2.2 事件程式碼 .....	9-8



9.2.3 工具箱 .....	9-10
9.2.4 視窗 .....	9-11
9.2.5 程式加上行號.....	9-14
<b>9.3 表單.....</b>	<b>9-15</b>
9.3.1 表單屬性.....	9-15
9.3.2 表單事件.....	9-18
9.3.3 第一個視窗程式.....	9-20
<b>9.4 標籤控制項.....</b>	<b>9-26</b>
9.4.1 Label 控制項.....	9-26
9.4.2 LinkLabel 控制項.....	9-31
<b>9.5 文字編輯控制項.....</b>	<b>9-35</b>
9.5.1 TextBox 控制項.....	9-35
9.5.2 RichTextBox 控制項.....	9-40
9.5.3 MaskedTextBox 控制項.....	9-41
<b>9.6 按鈕控制項.....</b>	<b>9-45</b>
<b>9.7 訊息對話方塊.....</b>	<b>9-48</b>
9.7.1 MessageBox.Show 方法.....	9-48
9.7.2 MessageBox::Show()範例.....	9-51
<b>習作園地.....</b>	<b>9-53</b>

## 第 10 章 進階控制項

<b>10.1 清單控制項.....</b>	<b>10-2</b>
10.1.1 RadioButton 控制項.....	10-2
10.1.2 CheckBox 控制項.....	10-6
10.1.3 ListBox 控制項.....	10-8
10.1.4 CheckedListBox 控制項 .....	10-15
10.1.5 ComboBox 控制項.....	10-15
<b>10.2 旋轉控制項.....</b>	<b>10-19</b>
10.2.1 NumericUpDown 控制項.....	10-19

10.2.2 HScrollBar、VScrollBar 捲軸控制項 .....	10-22
<b>10.3 容器控制項 .....</b>	<b>10-24</b>
10.3.1 GroupBox 控制項 .....	10-24
10.3.2 Panel 控制項 .....	10-27
<b>10.4 日期時間控制項 .....</b>	<b>10-29</b>
10.4.1 MonthCalendar 控制項 .....	10-29
10.4.2 DateTimePicker 控制項 .....	10-34
10.4.3 Timer 控制項 .....	10-36
<b>10.5 圖形控制項 .....</b>	<b>10-39</b>
10.5.1 PictureBox 控制項 .....	10-39
10.5.2 ImageList 控制項 .....	10-43
<b>習作園地 .....</b>	<b>10-48</b>

## 第 11 章 功能表控制項與含有多表單和類別的方案

<b>11.1 功能表控制項與含有多表單和類別的方案 .....</b>	<b>11-2</b>
11.1.1 MenuStrip 功能表控制項 .....	11-2
11.1.2 ContextMenuStrip 快顯功能表控制項 .....	11-10
<b>11.2 工具列控制項 .....</b>	<b>11-14</b>
11.2.1 ToolStrip 工具列控制項 .....	11-14
11.2.2 StatusStrip 狀態列控制項 .....	11-14
<b>11.3 含有多表單和類別的方案 .....</b>	<b>11-20</b>
11.3.1 建立多個表單和類別檔的專案 .....	11-20
11.3.2 認識 MDI 表單 .....	11-22
<b>習作園地 .....</b>	<b>11-31</b>

## 第 12 章 對話方塊

<b>12.1 字型對話方塊 .....</b>	<b>12-2</b>
<b>12.2 色彩對話方塊 .....</b>	<b>12-6</b>
<b>12.3 OpenFileDialog 對話方塊 .....</b>	<b>12-9</b>

<b>12.4 SaveFileDialog 對話方塊</b> .....	<b>12-11</b>
<b>12.5 FolderBrowserDialog 對話方塊</b> .....	<b>12-14</b>
<b>12.6 PrintDocument 列印文件控制項</b> .....	<b>12-15</b>
12.6.1 PageSetupDialog 列印格式對話方塊控制項.....	12-16
12.6.2 PrintPreviewDialog 預覽列印對話方塊控制項.....	12-17
12.6.3 PrintDialog 列印對話方塊控制項.....	12-18
12.6.4 綜合練習：列印文件控制項的使用.....	12-19
<b>習作園地</b> .....	<b>12-22</b>

## 第 13 章 滑鼠鍵盤與共享事件

<b>13.1 鍵盤事件介紹</b> .....	<b>13-2</b>
13.1.1 KeyPress 事件.....	13-2
13.1.2 KeyDown、KeyUp 事件.....	13-5
<b>13.2 滑鼠事件介紹</b> .....	<b>13-10</b>
13.2.1 Click 和 DoubleClick 事件.....	13-10
13.2.2 MouseDown、MouseUp 和 MouseMove 事件.....	13-10
<b>13.3 控制項共享事件方法</b> .....	<b>13-14</b>
13.3.1 建立共享事件方法.....	13-14
13.3.2 動態新增和刪除共享事件方法.....	13-18
<b>習作園地</b> .....	<b>13-22</b>

## 第 14 章 繪圖與多媒體

<b>14.1 繪圖工具</b> .....	<b>14-2</b>
14.1.1 繪圖座標.....	14-2
14.1.2 建立畫布.....	14-2
14.1.3 建立畫筆和筆刷.....	14-5
<b>14.2 繪製文字和圖形</b> .....	<b>14-10</b>
14.2.1 繪製文字和直線.....	14-10
14.2.2 繪製矩形和多邊形.....	14-13

14.2.3 繪製曲線.....	14-14
14.2.4 繪製填滿圖形 .....	14-18
14.2.5 圖形平移、縮放和旋轉.....	14-21
<b>14.3 圖檔的讀取和儲存.....</b>	<b>14-24</b>
14.3.1 Bitmap 類別.....	14-24
14.3.2 使用 Bitmap 物件建立 Graphics 物件.....	14-27
14.3.3 在畫布上顯示 Bitmap 物件 .....	14-28
<b>14.4 綜合練習：簡易小畫家.....</b>	<b>14-33</b>
<b>14.5 多媒體.....</b>	<b>14-39</b>
14.5.1 多媒體播放程式.....	14-39
14.5.2 System::Media 播放 .wav 音效檔和系統音效.....	14-43
<b>習作園地 .....</b>	<b>14-48</b>

## 第 15 章 資料庫程式設計工具的使用

<b>15.1 DataSet 物件模型.....</b>	<b>15-2</b>
<b>15.2 資料庫設計工具的使用.....</b>	<b>15-4</b>
15.2.1 連線 Access 資料庫 .....	15-4
15.2.2 連線 SQL Server 2008 Express 資料庫 .....	15-11
<b>習作園地 .....</b>	<b>15-17</b>

## 第 16 章 檔案與資料夾處理

<b>16.1 檔案存取 .....</b>	<b>16-2</b>
<b>16.2 資料夾處理.....</b>	<b>16-3</b>
16.2.1 DirectoryInfo 類別 .....	16-3
16.2.2 Path 類別 .....	16-9
<b>16.3 檔案處理—FileInfo 類別.....</b>	<b>16-12</b>
<b>16.4 磁碟機處理—DriveInfo 類別 .....</b>	<b>16-23</b>
<b>16.5 檔案內容存取.....</b>	<b>16-26</b>
16.5.1 FileStream—存取檔案內容 .....	16-26

16.5.2 StreamReader—讀取文字檔案內容 .....	16-30
16.5.3 StreamWriter—寫入文字檔案內容.....	16-32
<b>習作園地.....</b>	<b>16-43</b>

## **第 17 章 專題製作**

---

<b>專題一：簡易猜數字遊戲.....</b>	<b>17-2</b>
<b>專題二：拼圖遊戲.....</b>	<b>17-7</b>
<b>專題三：配對記憶遊戲.....</b>	<b>17-15</b>
<b>專題四：踩地雷遊戲.....</b>	<b>17-23</b>
<b>專題五：兩人文字聊天室.....</b>	<b>17-34</b>

## 物件與類別

物件 (Object) 可說是一件東西，物件具有屬性 (Property)、方法 (Method) 以及觸發的事件 (Event)。例如：David (物件) 是一個人 (類別)，David 的身高是 183 公分 (屬性)，David 會跑步 (方法)。

類別 (Class) 是一個物件的設計藍圖，而物件則是由該類別所定義出來的實體。例如：建築師設計了一張房子的設計圖 (Class)，工程人員可以根據這張設計圖，蓋出了很多棟的房子 (Object) 來。

### 學習重點

---

- 物件導向程式設計
- 物件與類別
- C++ 定義類別
- CLR 定義類別
- 屬性
- 方法
- 多載
- 建構式
- 解構式
- 事件
- 靜態成員

## 7.1 物件與類別

### 7.1.1 物件導向程式設計

目前的程式設計主流是物件導向 (OOP)，C++ 已提供完整的物件導向功能。在物件導向設計裡，物件可以被封裝 (保護) 和繼承 (重複使用)。因為物件的封裝會使物件受到保護，要存取物件必須透過屬性 (Property)、方法 (Method) 和事件 (Event)。物件的封裝如同日常生活中的 ATM 提款機，其提款必須透過正確的方式才能提款 (輸入帳號和密碼)，而不可用不當破壞方式提款。

物件的繼承可使程式碼重覆再利用，當產生一個表單或拖曳一個按鈕到表單中時，就已經繼承了別人撰寫的物件。因為物件的繼承使得前人的經驗和智慧可以被不斷的累積，如此，才能創造出像 Windows 這麼龐大的系統。

### 7.1.2 物件與類別的觀念

物件 (Object) 可說是一件東西，物件具有屬性 (Property)、方法 (Method) 以及觸發的事件 (Event)。例如：David (物件) 是一個人 (類別)，David 的身高是 183 公分 (屬性)，David 會跑步 (方法)。如果用 C++ 程式表示如下：

```
Person David;           //建立 David 物件屬於 Person 人類類別
David.Height=183;      //David 的身高是 183 公分
David.Walk(5);         //David 每小時可以跑 5 公里
```

在 C++ 工具箱中的控制項，通常都有事先已經定義好的屬性、方法和事件，由這些控制項可建立物件。例如：由 Button 控制項建立的按鈕 MyButton1 就是一個物件，它具有 Text、BackColor 等屬性，Show()、Hide() 等方法和 Click() 等事件。

類別 (Class) 是一個物件的設計藍圖，而物件則是由該類別所定義出來的實體。例如：建築師設計了一張房子的設計圖 (Class)，工程人員可以根據這張設計圖，蓋出了很多棟的房子 (Object) 來。

本書會先以原生 C++ 的語法介紹物件導向的觀念，隨後再以 CLR 的語法詳細說明物件導向的實作、屬性、方法、建構式、解構式、多載、繼承…。

C++ 建立實體物件的語法有三種，第一種語法為：

```
類別名稱 物件變數名稱; //根據類別建立實體物件
```

例如以 `Person` 類別建立 `David` 物件的語法如下：

```
Person David;
```

第二種語法為建立指標物件，再將指標指向物件變數配置記憶體位址：

```
類別名稱 物件變數名稱;
類別名稱 *指標物件變數; //先宣告指標物件變數屬於某個類別
指標物件變數 = &物件變數名稱; //再將指標指向物件變數配置記憶體位址
```

例如：

```
Person David;
Person *Amy;
Amy=&David;
```

第三種語法為建立動態配置的指標物件：

```
類別名稱 *指標物件變數; //先宣告指標物件變數屬於某個類別
指標物件變數 = new 類別名稱([初始值]); //再使用 new 動態配置記憶體空間
```

例如：

```
Person *David; //先宣告指標物件變數 David
David = new Person(); //配置記憶體空間
```

上面兩行也可合併成一行：

```
類別名稱 *指標物件變數 = new 類別名稱([初始值]);
```

例如：

```
Person *David = new Person(); //宣告並建立 Person 類別的 David 物件
```

要建立物件必須先有類別，有了類別才能產生物件，類別就像建築的藍圖，而物件則是由藍圖所建出的一棟棟的房子。



### 7.1.3 C++ 定義類別

要自行定義類別 C++ 必須使用「class 類別名稱{;}」語法，並依需要在類別中定義類別成員（包括欄位、方法和事件等），其中欄位相當於變數或常數，可以直接存取。

#### C++ 定義類別的語法

```
class 類別名稱
{
    private:
        [static] 資料型別 成員;
    public:
        [static] 資料型別 成員;
    protected:
        [static] 資料型別 成員;
};
```

例如：

```
class Person          //定義 Person 類別
{
    private:
        int Weight;    //宣告 Weight 體重欄位為私有型態
    public:
        int Tall;      //宣告 Tall 身高欄位為共用型態
        void Walk(int var) //定義 Walk 方法為共用型態
        {
            cout << "每小時跑公里數=" << var;
        }
};
```

存取修飾詞中，資料成員若宣告為 **public** 代表該資料為共用型態，其存取沒有限制，即允許目前應用程式中的所有類別均可存取。若宣告為 **private** 代表私有型態，只允許在類別內部使用。而 **protected** 則可以在類別內部以及其子類別中使用，若省略關鍵字，系統的預設值為 **private**。

「欄位」相當於變數或常數，用以儲存物件的資料，可用一般的資料型別或其他的類別來宣告。「屬性」可以保護物件欄位，屬性的建立較為複雜，將在 7.3 節中詳細說明。「方法」用以表示該類別所擁有的行為。

類別的最後面記得要加上「;」號，否則會產生編譯錯誤。

有了類別後，就可以依據類別來建立物件，並以「.」運算子存取其成員。

```
Person David; //宣告並建立 Person 類別的 David 物件
David.Tall=183; //存取成員
```

也可以指標物件宣告建立動態物件指標，並以「->」運算子存取其成員。

```
Person *Tony = new Person(); //建立物件指標宣告
Tony->Name = 183; //存取成員
```

## 範例：C++ 自行定義類別

定義 Person 類別，並建立 Name 姓名欄位、Tall 身高欄位為共用欄位，Walk() 方法顯示跑步的速度。最後在 \_tmain() 函式中使用 Person 類別建立兩個物件。(FirstClass1.sln)

執行結果



```

c:\VC++2008\ch07\FirstClass1\Debug\FirstClass1.exe
David 身高=183
David每小時可跑 5 公里
-----
Tony 身高=183
Tony每小時可跑 8 公里
請按任意鍵繼續 . . .

```

(可以開啟 <C:\VC++2008\ch07\完成參考\FirstClass1\FirstClass1.sln> 專案觀看結果)

動手實作 ( 本例請建立 **Win32 主控台應用程式** 專案)

1. 新增專案：新增 **Win32 主控台應用程式** 專案，專案名稱為「FirstClass1」。
2. 在 FirstClass1.cpp 內輸入下列程式碼來定義 Person 類別。

```

1 // FirstClass1.cpp : 定義主控台應用程式的進入點。
2 //
3
4 #include "stdafx.h"
5 #include "iostream"
6 #include "string"
7 using namespace std;
8

```

```

9  class Person
10 {
11     public:
12     int Tall;    //宣告 Tall 身高欄位為公用型態
13     string Name; //宣告 Name 姓名欄位為公用型態
14     void Walk(int var)    //定義跑步方法
15     {
16         cout << Name << "每小時可跑" << var << " 公里" << endl;
17     }
18 };
19
20 int _tmain(int argc, _TCHAR* argv[])
21 {
22     Person David; //宣告並建立 Person 類別的 David 物件
23     David.Name = "David";    //設定 Name 欄位
24     David.Tall = 183;        //設定 Tall 欄位
25     cout << "David 身高=" << David.Tall << endl; //取得 Tall 欄位
26     David.Walk(5); //呼叫 Walk() 方法，並傳遞參數值 5
27     cout << "-----\n";
28
29     // 使用動態指標宣告
30     Person *Tony;    //先宣告 Person 類別的 Tony 指標物件
31     Tony = new Person(); //配置記憶體空間
32     Tony->Name = "Tony";
33     Tony->Tall = David.Tall;
34     cout << "Tony 身高=" << Tony->Tall << endl; //取得 Tall 欄位
35     Tony->Walk(8);
36     system("pause");
37     return 0;
38 }

```

### 3. 儲存專案及執行。

#### Person 類別程式說明

- 9-18 定義 Person 類別。建立的類別並不會主動執行，必須用類別建立物件後，由物件以呼叫屬性、方法等方式來執行。
- 12-13 建立二個欄位。
- 14-17 建立 Walk 方法，此方法可以傳送「速度 (var)」參數，同時顯示速度，此方法的存取是 public 如此才可以被不同類別 (\_tmain() 函式) 呼叫。

\_tmain() 程式說明

22      Person David, 建立 Person 類別的 David 物件。  
 23-24    設定二個欄位的初始值, 物件成員存取語法是「物件.成員」。  
 25      顯示欄位值。  
 26      呼叫 Walk 方法, 並傳遞參數, 於是第 16 列程式會顯示 David 的速度。  
 30-31    Person \*Tony 以指標物件建立另一個物件 Tony。  
 32-35    指標物件成員存取語法是「物件->成員」。

## 7.1.4 CLR 定義類別

了解 C++ 如何定義類別後, 接著要探討 CLR 定義類別的語法, CLR 定義類別的語法和 C++ 相似, 但 CLR 類別名稱類別前應加上關鍵字 [public][value|ref]。

### CLR 定義類別的語法

```
[public][value|ref] class 類別名稱
{
  private:
    [static] 資料型別 成員;
  public:
    [static] 資料型別 成員;
  protected:
    [static] 資料型別 成員;
};
```

例如: 宣告 Person 類別是全域性類別。

```
public class Person{};
```

省略 public 宣告, 預設是 public, 所以上面宣告可以省略為:

```
class Person{};
```

上面不使用 ref 或 value 關鍵字的語法其實是為了和舊的 C++ 原生語法相容, 亦即在 CLR 模式也可以使用原生 C++ 相容的語法「class 類別名稱{};」來建立類別, 但使用這種語法建立的類別, 在 CLR 模式會有一些限制, 例如: 無法建立堆積變數、屬性。

例如: 在 CLR 模式使用原生 C++ 建立類別語法建立類別, 並無法建立堆積變數、屬性。

```
class Car // 註: class 前未加 ref 或 value
```

```
{
private: int mSpeed;
public :
    int n;
    // 在非 ref class 不可建立屬性，以下建立屬性將產生編輯錯誤
    /*
    property int Speed      //定義 Speed 屬性
    {
        int get(){ return mSpeed; }
        void set(int value){ mSpeed = value; }
    }*/
};

int main(array<System::String ^> ^args)
{
    Car a1; // 建立物件方法一
    a1.n=1;
    Console::WriteLine(L"a1.n= {0}",a1.n);

    Car b1; // 建立物件方法二
    Car *a2;
    a2=&b1;
    a2->n=2;
    Console::WriteLine(L"a2->n= {0}",a2->n);

    Car *a3=new Car(); // 建立物件方法三
    a3->n=3;
    Console::WriteLine(L"a3->n= {0}",a3->n);

    // 在非 ref class 類別建立堆積變數，以下建立^a4 將產生編輯錯誤
    /* Car ^a4=gcnew Car(); */

    Console::Read();
    return 0;
}
```

最好是完全採用標準的 CLR 模式語法來建立類別，亦即必須再加上關鍵字 `ref` 或 `value`，`ref` 代表建立的類別是參考類別，`value` 代表建立的類別是數值類別。