



EMBARCADERO
TECHNOLOGIES®

白皮書

Delphi 2009 全覽

作者：Marco Cantù

2008 年 11 月

公司總部
100 California Street, 12th Floor
San Francisco, California 94111

EMEA 總部
York House
18 York Road Maidenhead,
Berkshire SL6 1SF, United
Kingdom

亞太地區總部
L7.313 La Trobe Street
Melbourne VIC 3000
Australia

簡介：DELPHI IDE

自從 Delphi 8 for .NET 及 Delphi 2005 for Win32 以來，Delphi 已採用根據略微不同型制的全新 IDE，其中的內嵌式編輯器具有設計工具窗格，可停駐於定位，而非浮動的編輯器及設計工具。這個 Delphi IDE 的「第二版」通常以內部代號 Galileo 表示。

Delphi 2009 採用第 6 版 Galileo IDE。這不僅是第一版擁有所有轉換為 Unicode 的設計工具，更有一些相當有意思的新功能，特別是有關專案管理方面的功能。

安裝及執行

和 Delphi 2007 一樣，Delphi 2009 也是以 InstallAware 進行安裝。不過這一次，安裝過程有很大的改善，特別是安裝的速度。Delphi 2009 安裝可在 20 分鐘內完成，不需要數小時的時間。

這方面的重大變革是說明可個別安裝，因此可個別從主要產品進行更新(因此，若要取得更新的說明，便不需要重新安裝 Delphi；萬一要重新安裝 IDE，也不需要重新安裝說明)。安裝說明會比安裝實際產品花費更多的時間，說明安裝影像會大於 IDE 影像。

在 Windows Vista 進行安裝時，會將產品(預設)安裝在下列資料夾：

```
C:\Program Files\CodeGear\RAD Studio\6.0  
C:\Users\Public\Documents\RAD Studio\6.0\Demos\  
C:\Program Files\Common Files\CodeGear Shared
```

不需要 .NET SDK

從 Delphi 8 一直到 Delphi 2007，安裝 IDE 的必要條件是需要 Microsoft .NET SDK (版本 1.1 以前，版本 2.0 以後)。這對於 Delphi 2009 而言已非必要條件。您仍需要安裝極少部分的 Microsoft .NET 執行階段，這可能已經安裝在作業系統中，不過另一方面則不需要開發套件，這個套件相當大，需要數百 MB。

在這個 Delphi 版本中，CodeGear 使用 Microsoft 的「文件總管」。過去這只能在 SDK 中取得，但現在可以個別安裝部署。

Delphi 的說明相當龐大(這也就是為什麼安裝時要花上那麼多時間)，因為它同時包含 CodeGear 說明文件與 Microsoft 平台說明文件。不過，在這一版中，研發團隊修正了一些「排序」問題，讓特定 Delphi 主題能持續列在一般性平台問題之前。特定 Delphi 內容也經過大幅修訂。

Windows 安裝程式清除

有時，解除安裝 Delphi 來安裝更新版本時，安裝程式會出現問題而停止，讓我們無法如預期般完成工作。出現這種情況時，CodeGear 建議您清除所有應用程式資料夾 (根據作業系統，可能另外包括一些隱藏資料夾)。或者，您也可以使用 Microsoft 提供的「Windows 安裝程式清除工具」公用程式，這可在此找到：

```
http://support.microsoft.com/default.aspx?scid=kb;en-us;290301
```

要注意的是，使用這種低層級工具可能會對系統造成干擾，因此在執行工具時，要十分小心 (最好先閱讀使用說明，而且您必須自行承擔可能的風險)。

-IDECAPTION 旗標

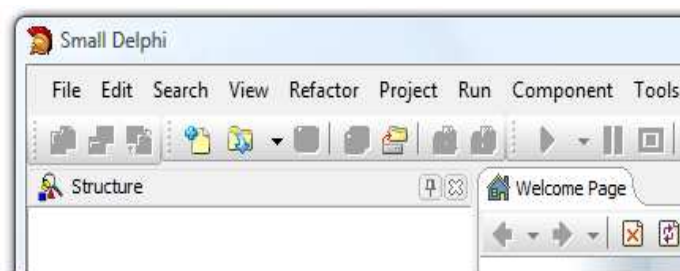
您可能已經知道 (儘管這是多年來秘而不宣的事實) 可以同時利用 -R 指令行旗標，以不同登錄設定執行多個 IDE 執行個體。

同時執行兩種不同的 IDE 版本，會造成難以彼此分辨的問題。另一個 IDE 適用的伴隨指令行參數是 -idecaption，這會將標題視為一個值。結合這兩個旗標，您就可以利用下列連結執行 IDE：

```
"C:\Program Files\CodeGear\RAD Studio\6.0\bin\bds.exe" -pDelphi -rSmall -idecaption="Small Delphi"
```

此一指令僅會執行具備 Delphi Win32 特性的 Delphi IDE、啟動「小型」登錄設定，並將 IDE 標題變更為「Small Delphi」，如下所示：

若非由指令行指定，則會從「特性」部分中的「登錄」擷取 IDE 標題，在其中，對於 IDE 的各個版本 (或使用特性)，都會有不同的字串。



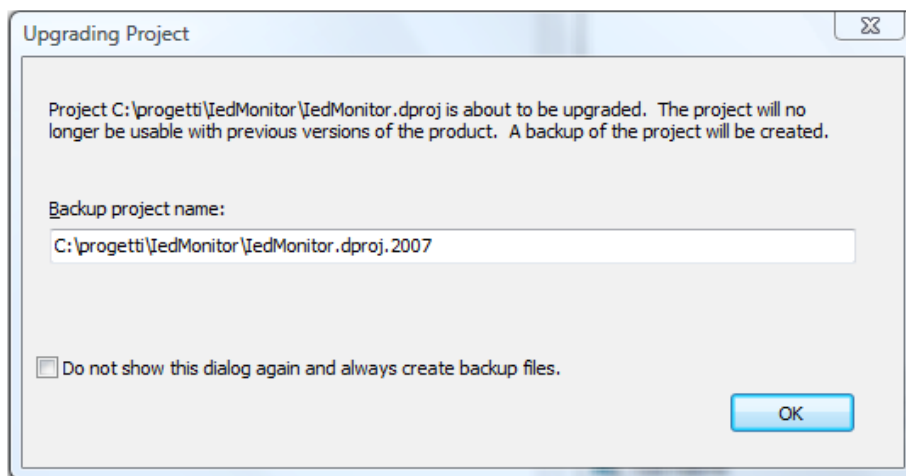
管理 DELPHI 專案

管理專案可說是相當常用的操作。如果 Delphi 2007 增加一些全新的概念，例如 MSBuild 支援、目標建置 (除錯與發行)、建置前與建置後事件，新的版本會讓這些功能更有彈性且更易於使用，一切都是從完全改寫的「Project Manager」本身開始進行操作。不過，在探討「Project Manager」前，必須先看看升級專案檔案與更新的「Project Option」對話方塊。

升級專案組態檔

從 Delphi 問世初期起，專案原始程式碼檔案 (附檔名 .DPR) 就已經包含 Object Pascal 程式碼，並使用一個或多個個別專案組態檔來儲存其他設定。在最近幾次改版中，專案組態檔的格式與附檔名歷經數次變更，從 INI 檔變更為 XML 檔，變更為 XML 檔後，即可供 MSBuild 使用 (.DPROJ 檔案格式)。

從 Delphi 2007 到 Delphi 2009，此一專案組態檔的整體格式並沒有任何變更。不過，內容確實仍有所不同。Delphi 2007 無法辨識新版 IDE 所增加的其他選項。開啓現有的 Delphi 2007 專案時，Delphi 2009 IDE 會要求您輸入備份檔案的名稱，以將專案組態檔現有版本複製到備份檔案：



專案組態備份檔的預設名稱爲專案名稱，但附檔名改爲 .dproj.2007。在此一特定情況中，我將專案檔案重新命名爲 IedMonitor2007.dproj。執行此一操作後，IDE 會將以下一行新增至訊息窗格：

```
Upgrading project. Backup  
C:\progetti\IedMonitor\IedMonitor2007.dproj created.
```

請注意，在您確實完成儲存動作之前，不會建立更新的 Delphi 2009 版本專案組態檔。

您可利用這個備份版本重新開啓 Delphi 2007 中的專案。不過，如果您需要向下相容，最好是以不同名稱儲存 Delphi 2009 版本專案。

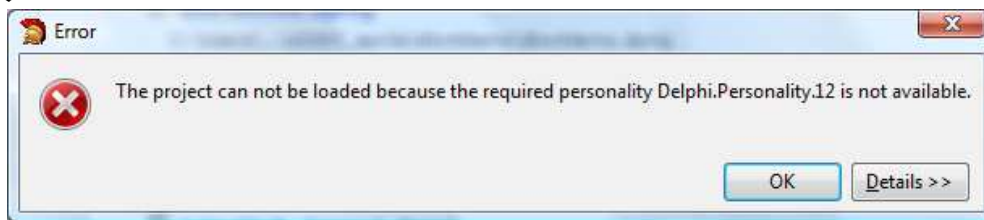
在新的 .DPROJ 檔案中，Delphi 2009 增加一個新的專案版本旗標：

```
<ProjectVersion>11.1</ProjectVersion>
```

升級包括建置組態中的變更 (稍後說明)，以及資源管理中的變更。下列是新增部分或經過大幅修改的部分：

```
<PropertyGroup Condition="'$(Config)'=='Release' or  
    '$(Cfg_Release)'!=''>  
    <Cfg_Release>true</Cfg_Release>  
    <CfgParent>Base</CfgParent>  
    <Base>true</Base>  
</PropertyGroup>  
<PropertyGroup Condition="'$(Config)'=='Debug' or  
    '$(Cfg_Debug)'!=''>  
    <Cfg_Debug>true</Cfg_Debug>  
    <CfgParent>Base</CfgParent>  
    <Base>true</Base>  
</PropertyGroup>  
<PropertyGroup Condition="'$(Base)'!=''>  
    <DCC_DependencyCheckOutputName>SimpleApp.exe  
    </DCC_DependencyCheckOutputName>  
</PropertyGroup>  
<ItemGroup>  
    <DelphiCompile Include="SimpleApp.dpr">  
        <MainSource>MainSource</MainSource>  
    </DelphiCompile>  
    <DCCReference Include="SimpleAppMainForm.pas">  
        <Form>Form30</Form>  
    </DCCReference>  
    <BuildConfiguration Include="Base">  
        <Key>Base</Key>  
    </BuildConfiguration>  
    <BuildConfiguration Include="Release">  
        <Key>Cfg_Release</Key>  
        <CfgParent>Base</CfgParent>  
    </BuildConfiguration>  
    <BuildConfiguration Include="Debug">  
        <Key>Cfg_Debug</Key>  
        <CfgParent>Base</CfgParent>  
    </BuildConfiguration>  
</ItemGroup>
```

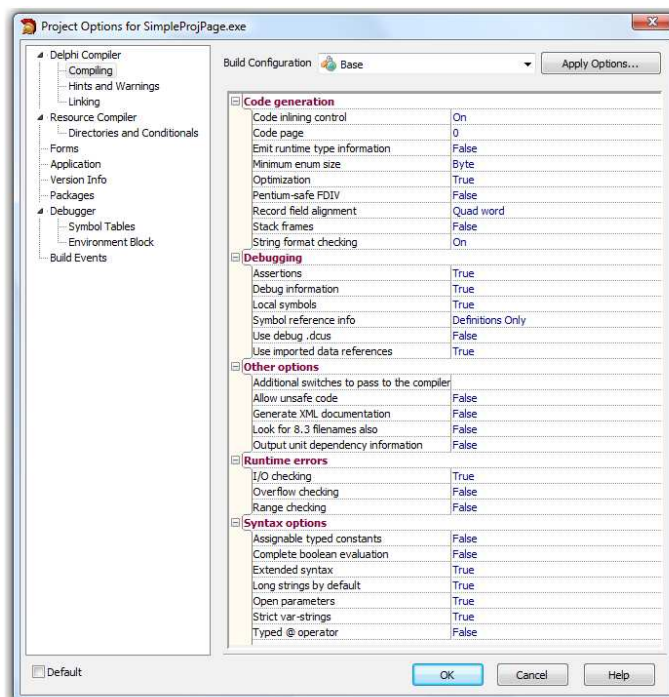
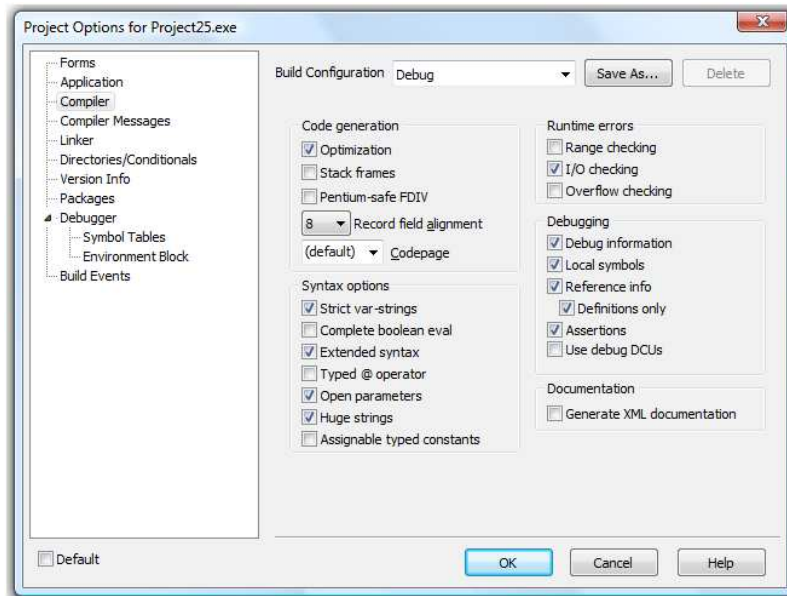
如果您嘗試在 Delphi 2007 中重新開啓專案檔案 (唯一能辨識出此種格式的舊版本)，您會看見下列錯誤：



重新設計的 Project Option 對話方塊

「Project Option」對話方塊是我較常使用的 Delphi 對話方塊，我相信有很多人也是如此。因此，Delphi 2009 的大幅重新設計讓我有時感到些許迷惑。重新設計的部分與建置組態中列出選項的頁面有關，且 (我們稍後會在「建置組態與組態設定」部分討論) 這些對話方塊的頁面也會在「Project Configuration Manager」中用到。

舉例來說，您可以比較 Delphi 2007 與 Delphi 2009 中「Delphi Compiler Option」頁面有哪些不同之處：



兩者確實有顯著的不同。核取方塊以「 True/False 」取代，選項按鈕則以內含多種選項的下拉式方塊取代。兩者頁面底部都有說明區域 (在上圖中，我將這塊區域最小化，這樣才能顯示對話方塊頁面中的所有選項)，在這塊區域中，會提供各種選項的精簡資訊。「Description」區域中提供的一個有趣項目是選項的預設值。

圖形的重新設定得花上一點時間適應；除此之外，每個群組的項目現在都以字母順序排列，因此列出的順序跟先前版本也有所不同。目錄選項現在出現在主要 Delphi 編譯器節點下。不過，除了這些組織架構方面的變更，是否還有哪些項目已經移除或新增？

編譯器的全新專案選項

在「Delphi Compiler/Compiling」頁面(過去稱為「Compiler」)中，「Code generation」部分有下列新選項：

- z 「Code inlining control」與 **\$INLINE** 編譯程式指示詞相對應，並控制內嵌的運作方式。
- z 「Emit runtime type information」與指令行的 **-\$M** 旗標或 **\$M** 程式指示詞相對應，且會決定指定類別(或專案中所有類別)的執行階段時間資訊產生。
- z 「Minimum enum size」與 **-\$Z** 旗標(或 **\$Z** 程式指示詞)相對應，且會決定用於列舉類型值的最小大小(1 位元、1 個字組、1 個雙字組或 1 個四字組)。
- z 「String format checking」預設為啟用，您可以停用此功能，以避免一些自動字串格式檢查(如呼叫 **EnsureUnicodeString** 函式以及「Ensure String」系列其他函式)；此函式與 **\$STRINGCHECKS** 程式指示詞相對應。這是 Delphi 2009 的全新編譯器選項，過去我們都認為這是未公布的隱藏功能，因此，發現這個功能明顯出現在「Project Options」對話方塊時，確實讓人感到十分驚訝。
- z 「Code page」在過去的版本中已經存在，但現在則與 **AnsiString** 類型的運作方式更有相關性(這會在此白皮書系列裡討論 Delphi 2009 中的 Unicode 時加以說明)。

「Debugging」部分有一個新的選項「Use imported data references」(對應於 **\$G**)，能夠控制建立匯入資料參照(這能提升記憶體效率，但無法存取其他執行階段套件中定義的全域變數)。

「Runtime errors」與「Syntax options」部分則具有與過去 Delphi 版本相同的項目(以及相同的預設值)。「Other options」部分則出現新的選項，但其中的「Generate XML documentation」則是先前已提供的選項：

- z 「Additional switches to pass to the compiler」可用來直接插入 IDE 未特別支援的指令行編譯器選項(雖然出現這個功能，表示 Delphi 2009 在技術上已經可以支援每個編譯器選項)。
- z 「Allow unsafe code」可以讓您編譯在 .NET 之類的受管理環境中被視為不安全的程式碼，不過這甚少(或完全不)適用於 Win32 編譯器。
- z 「Look for 8.3 filenames also」會指示編譯器在相當老舊的 Windows 版本上工作；與 **-P** 編譯器選項相對應。
- z 「Output unit dependency information」會開啓 **--depends** 編譯器旗標，這一項目前顯然未保留。

其他全新的專案選項

「Hints」與「Warnings」頁面與舊的「Compiler Messages」頁面相對應。當然，其中有一些與 Unicode 字串和其他全新編譯器功能有關的提示。

「Linking」頁面(過去稱為「Linker」)儘管看起來有很大的不同(看起來更為精簡)，但裡面只有一個新選項「Set base address for relocatable images」。

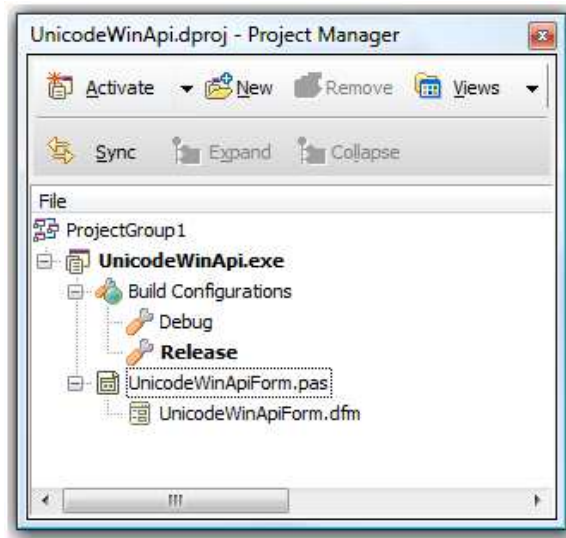
主要層級 Delphi 編譯器頁面的選項，與先前「Directories/Conditionals」的選項完全相同。但真正令人感到困惑的是，在「Resource Compiler」主要層級頁面下的資源編譯器組態中，也有一個「Directories and Conditionals」頁面。這些都是全新的頁面，主要目的是讓您能夠以過去從未採用的方式，從 Delphi IDE 控制資源編譯器。本白皮書稍後的「在 IDE 中管理資源」部分會討論這個主題。

預設專案位置

從 Delphi 2005 起，所有新專案的預設位置都已經是在使用者文件資料夾下。很少 Delphi 開發者知道，只需在「Environment Options」頁面「Tools」|「Options」對話方塊的「Default project edit」編輯方塊中設定一個值，就能夠變更這個位置。

PROJECT MANAGER

除了「Project Options」對話方塊經過重新設計之外，Delphi 2009 也針對 IDE 其中一個最常使用的「Project Manager」窗格進行大幅度更新。就算只是大略檢視這個視窗，也能夠看出幾個新功能：



您可以看到一個新的「Build Configurations」節點，此一節點下有數個子節點，可以用來啟動一個建置組態，方法比在 Delphi 2007 下進行要簡單許多。稍後會在「建置組態與組態設定」部分說明這個主題。

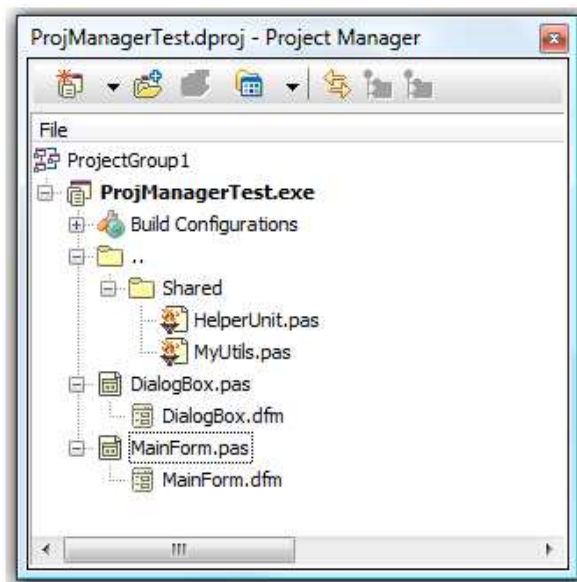
「Project Manager」工具列有幾個新的按鈕。新的「Sync」按鈕可以從「Project Manager」編輯器中選取現有檔案，當然，這個檔案必須是專案的一部分。與此相對的操作方式(就是在編輯器中啟動「Project Manager」的目前選取項目)是按兩下滑鼠。

「Expand」與「Collapse」按鈕則可以在現有節點下重複展開與收合所有節點。在一個專案群組下執行「Expand」，就可以看見群組中所有專案全部組態與檔案節點的樹狀結構。我得說，這真的相當方便。接著，我要談談第四個新按鈕「Views」。

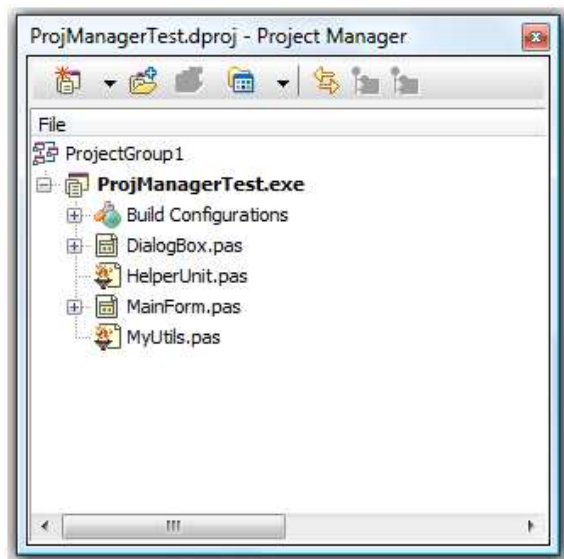
PROJECT MANAGER VIEWS

另一個全新功能就是「Project Manager Views」設定。在工具列右側，您可以看到新的「Views」按鈕，按一下這個按鈕，就能改變「Project Manager」顯示不同資料夾中檔案的方式。其中有三種選項。為了測試這三種選項，我建立一個範例程式(名稱爲「ProjManagerTest」)，在主資料夾中有兩種格式，在第二資料夾(名稱爲「Shared」)中則有兩個單元，這些格式與單元都放在檔案系統階層的另一層級：

- 「Directory (Nested)」是預設設定(也是 Delphi 8 至 Delphi 2007 中唯一可用的項目)，會顯示依目錄區分的檔案，且目錄會以個別可以展開的節點來模擬實際磁碟架構(因此，您可能必須展開多個節點，才能將幾個子節點下移)：



- 「Directory (Flat)」是一種新的檢視方式，檔案仍舊依目錄區分，但無論這個目錄在檔案系統的哪個位置，每個目錄都是清單的一部分。也就是說，您會看到一個資料夾清單，每個資料夾中都有檔案，這(可能)與其他巢狀資料夾不同：



- 「List」是一種新的檢視選項，與傳統 Delphi 7 專案管理員中的檔案清單相對應。其中完全沒有目錄，清單中的檔案是按照字母順序排列：

建置組態與組態設定

如我稍早所言（可從前面幾頁的圖片中看到），針對每個專案（亦即，假設您作業的專案群組中有多个進行中的專案），「Project Manager」都有一個新的「Build Configuration」節點。這個節點可以替代 Delphi 2007 中原先用來管理建置設定但使用不便的個別視窗。使用節點與其下的子節點，只需要按兩下滑鼠，就能變更目前的建置設定，並直接在指定節點上執行實際建置。

選取特定建置組態或主要節點，都可以新增新的組態。視您執行此動作時選取的項目而定，您可以建立主要組態或子組態。更精確的說，由於預先定義的組態會繼承基本組態的核心設定（即來自「Debug」與「Release」所繼承的核心組態），您選擇的節點會決定基本組態。

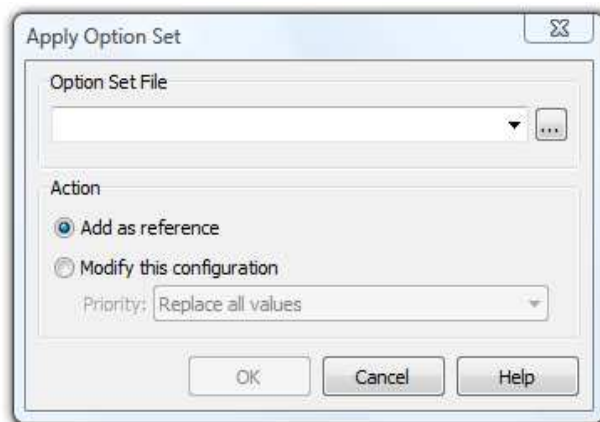
[-] Runtime errors	
[-] I/O checking	True
Value from "Base"	True
[-] Overflow checking	False
Value from "Base"	True
[-] Range checking	True
Value from "Base"	True

我所提到的從組態「繼承設定」是什麼意思？Delphi 2009 擁有一個全新的組態管理系統，在這個系統中，您可以將一個設定套用至特定組態（如「Debug」與「Release」），也可以設定從基本組態「繼承」兩種組態的選項。在特定組態中，您可以看到特定值以及從基本組態中繼承的值列在連續兩行中，由此可判斷它們是否彼此符合並改變自身或彼此（這也會影響特定組態）。選取左側的「Plus」符號，可展開每個組態設定行，就能加以封存。

在「Delphi Compiler/Compiling」頁面中展開三個執行階段錯誤行後，會得到下列的結果：
修改基本組態中的設定，也會影響從該設定繼承而來的其他任何組態。

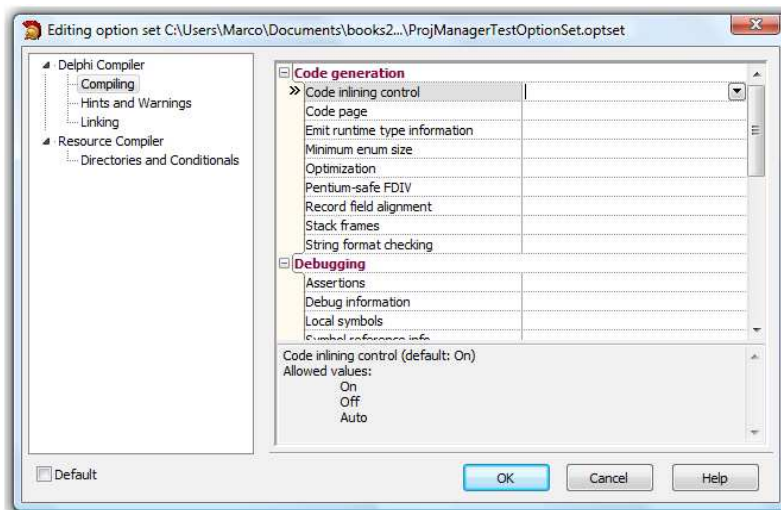
在「Project Manager」中，您可以選取一個建置組態，並將其中的設定匯出至「選項集」檔案。這就類似於將組態「*template*」或架構儲存至外部檔案，而且架構將連結至檔案。

由於您可以使用「Project Manager」(在建置組態上時，使用「Apply Option Set」本機功能表項目) 或「Project Options」對話方塊 (使用「Apply Options」按鈕)，這可讓您輕鬆將設定移至新的或其他現有的專案。在這兩種狀況中，Delphi 都會開啓「Apply Option Set」對話方塊，您可以在其中選擇檔案，並選擇是否要維持連結外部組態檔 (以便在檔案出現任何變更時，反映在使用這個外部組態檔的專案)，或只使用一些「*priority*」規則來合併現有設定：



在檔案中建立外部選項集後，您就可以使用「Project Manager」窗格上的「Edit」本機功能表，在任何指向此檔案的專案上編輯外部選項集。這會開啓包含「Project Options」對話方塊頁面子設定的編輯器，如下所示：

.OPTSET 檔是 XML 檔案，格式與 .DPROJ 格式類似，都是根據 MSBUILD XML 格式，屬於 OptionSet 專案類型。



在這個特定範例中，ProjManagerTestOptionsSet.optset 檔有下列內容：

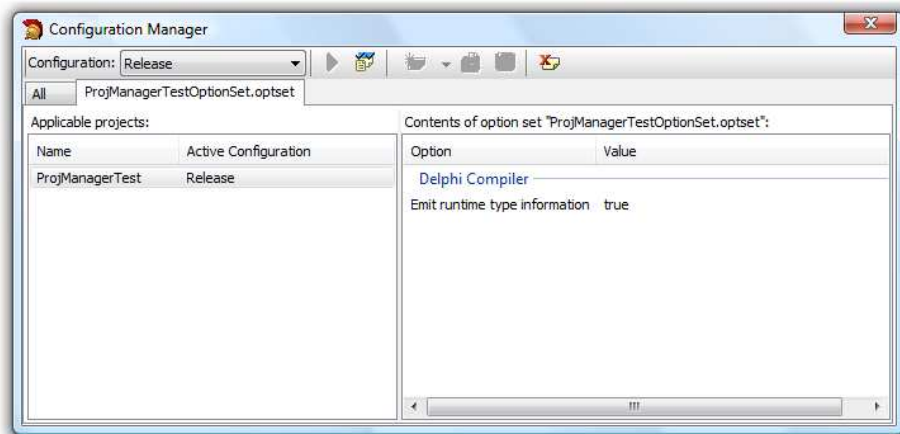
```
<Project xmlns="http://.../msbuild/2003">
  <PropertyGroup>
    <DCC_RunTimeTypeInfo>true</DCC_RunTimeTypeInfo>
  </PropertyGroup>
  <ProjectExtensions>
    <Borland.Personality>
      Delphi.Personality
    </Borland.Personality>
    <Borland.ProjectType>
      OptionSet
    </Borland.ProjectType>
    <BorlandProject>
      <Delphi.Personality/>
    </BorlandProject>
  </ProjectExtensions>
</Project>
```

PROJECT CONFIGURATION MANAGER

由於可以直接在「Project Manager」窗格中找到建置選項，因此您不需要使用「Configuration Manager」變更現有建置組態。不過，「Configuration Manager」對話方塊還是相當實用，因為您可以用它來同時變更專案群組中許多專案的建置組態。事實上，在 Delphi 2009 中仍然可以使用「Configuration Manager」，而且有長足的提升：它可讓您一次管理群組中所有專案的不同建置組態與選項集。

若要叫用「Configuration Manager」，您不能像在 Delphi 2007 那樣，使用「Project Manager」上的本機功能表；您必須在 Delphi 主要功能表下的「Project」功能表中，選取相對應的項目。

如此一來，您就能取得這個重新設計的使用者介面：



左側顯示專案的清單，以及各個作用中的組態。在右側，您可以看見選取組態的一些細節，例如非預設設定的清單（圖片中顯示的是在前一部分所列選項集檔案的摘要）。

使用標籤，您就可以根據指定組態或作用中的選項集來過濾左側的專案。

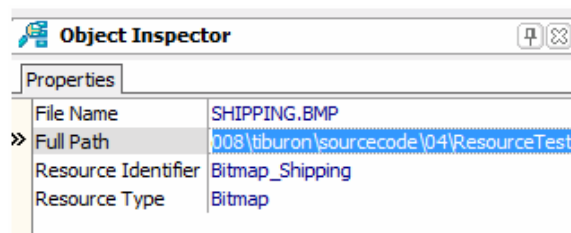
在 Delphi 2009 中，您可以使用「Configuration Manager」編輯每個建置組態的專案選項、增加新設定、建立或編輯選項集、修改作用中設定，並在單一位置執行大部分相關作業，使用起來也不會感到繁瑣。

當您在一個專案群組內進行多個專案的作業時，「Configuration Manager」佔有的顯著優勢是可在「Project Manager」中瀏覽，以便進行建置組態的作業。因此，針對單一專案，「Project Manager」擁有您所需要的一切功能。

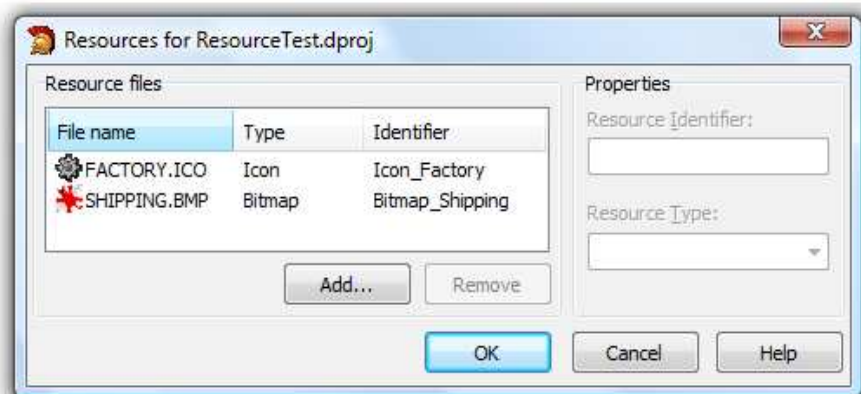
在 IDE 中管理資源

在最新的 Delphi 版本中，您可以在「Project Manager」中增加資源指令碼 (.RC 檔) 或經過編譯的資源檔案 (.RES 檔)，以便與專案一同編譯，然後連結至執行檔。在 Delphi 2009 中，由於包含了幾個新工具，所以管理資源的程序有所簡化。

首先，您可將個別資源檔拖曳至「Project Manager」，以便在專案中成為資源。您可以拖曳圖示、點陣圖與其他項目。Delphi 會針對這些額外的專案資源產生資源指令碼檔案，並直接連同程式進行編譯，將這些資源內嵌至執行檔中。您可以在「Object Inspector」中變更這些資源檔的任何屬性 (包括它們的內部名稱)：



然後，在 IDE 主要功能表的專案下拉式選單中，有一個新的「Resources」功能表項目。選取這個項目就會出現「Resources」對話方塊，您可以利用這個對話方塊來修改程式的所有資源、增加新的資源檔案、重新命名檔案、變更格式等：



在專案內增加數個資源後，Delphi 會在編譯時產生適合您的資源檔。針對名為「ResourceTest」的程式 (具備上面說明的資源)，Delphi 2009 會產生一個資源指令碼檔案，其中列出名稱為「ResourceTestResource.rc」的專案資源：

```
Icon_Factory Icon "FACTORY.ICO"  
Bitmap_Shipping Bitmap "SHIPPING.BMP"
```

這個資源指令碼檔案不可新增至專案 (如果您這麼做，會出現資源重複的警告)，不過會連同專案一併編譯。事實上，如果不小心出現錯誤 (例如將點陣圖宣告為圖示)，編譯器會出現錯誤訊息，並停止作業：

```
[BRCC32 Error] ResourceTestResource.rc(2): resource file  
SHIPPING.BMP is not in 3.00 format
```

並在違反規則的那一行開啓資源指令碼檔。在編譯階段，Delphi 2009 會產生 (或更新) 資源指令碼，並進行編譯，然後併入執行檔中。中繼檔是附檔名為 DRES 的檔案，這包含指示詞自動新增至專案原始程式碼檔案 (具有包含專案圖示與字串資源的標準 RES 檔案) 的專案中：

```
program ResourceTest;  
  {$R *.dres}  
uses Forms,  
    ResourceTest_MainForm in  
    'ResourceTest_MainForm.pas' {FormResourceTest};  
  {$R *.res}  
begin  
  Application.Initialize;  
  ...  
end
```

從 Delphi 2007 起，在資源編譯步驟中就已經出現 MSBuild 支援。如果您保留資源編譯器選項的 .Verbose 旗標，會看到下列相關輸出：

```
c:\program files\codegear\rad studio\6.0\bin\cgrc.exe -c65001 -v  
ResourceTestResource.rc -foResourceTest.dres  
  
CodeGear Resource Compiler/Binder  
Version 1.00 Copyright (c) 2008 Embarcadero Technologies Inc.  
  
Microsoft (R) Windows (R) Resource Compiler Version 6.0.5724.0  
Copyright (C) Microsoft Corporation. All rights reserved.  
  
Creating ResourceTest.dres  
Using codepage 65001 as default  
ResourceTestResource.rc.  
Writing ICON:1, lang:0x409, size 744  
Writing GROUP_ICON:ICON_FACTORY, lang:0x409, size 20.
```

```
| Writing BITMAP:BITMAP_SHIPPING, lang:0x409, size 44264
```

如果您從未直接使用 Windows 資源，「ResourceTest」程式會出現幾行程式碼，以載入圖示作為應用程式與主表單圖示，並會將點陣圖載入於「Image」元件中：

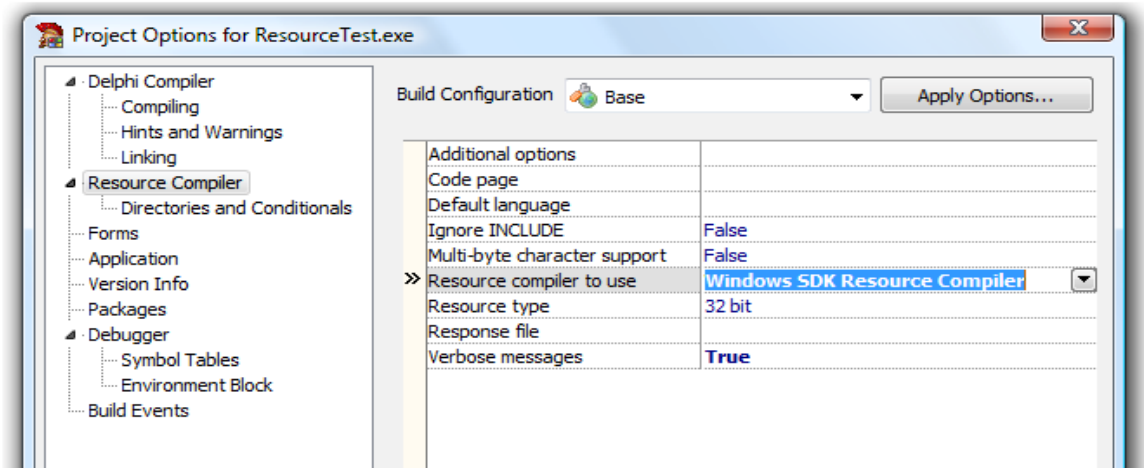
```
procedure
  TFormResourceTest.btnGifClick( Sender:
    TObject);
begin
  Image1.Picture.Bitmap.LoadFromResourceName(
    hInstance, 'Bitmap_Shipping');
end;

procedure
  TFormResourceTest.btnIconClick( Sender:
    TObject);
begin
  Icon.LoadFromResourceName(hInstance, 'Icon_Factory');
  Application.Icon.LoadFromResourceName(
    hInstance, 'Icon_Factory');
end;
```

「全新」資源編譯器

從舊版 Delphi 一直到 Delphi 2007，都使用「Borland Resource Compiler」(BRCC32.EXE)。

Delphi 2009 採用全新的資源編譯器，更準確的說法是不同的資源編譯器：來自 Microsoft Windows SDK 的編譯器。在支援 Windows 處理的所有新資源格式方面，這確實相當有幫助，但也衍生一些問題，因為一直以來，Borland 資源編譯器都會延伸 Microsoft 編譯器，提供現在已經不存在的額外功能。在「Project Option」對話方塊的資源編譯器部分中，您可設定相對應選項 (可在此編輯資源編譯器的數個其他參數)，以設定要使用的資源編譯器：



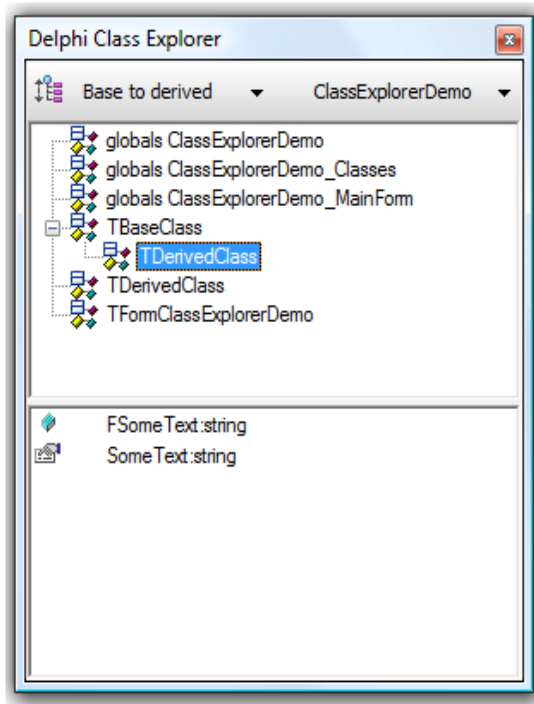
「Windows SDK Resource Compiler」需經由新的「CodeGear Resource Compiler/Binder」(這只是 SDK 編譯器的前端) 叫用。資源編譯器中的變更包括新增處理內嵌影像 (二進位) 資料的能力、支援字串清單中字串後的尾端逗號、處理字串的不同方式 (現在視為 C 語言字串，這會使得必須在含有雙反斜線的檔案名稱中避免使用任何 \ 符號)、管理資料夾中內含項目的方式等。

再強調一次，如果您從未直接使用資源檔，也許可以忽略這些變更。所有由 Delphi 環境直接管理的項目，從內嵌 DFM 檔作為資源，到使用 **resourcestring** 宣告，都完全向下相容。如果您直接使用資源，則務必謹慎修改資源檔案。

DELPHI CLASS EXPLORER

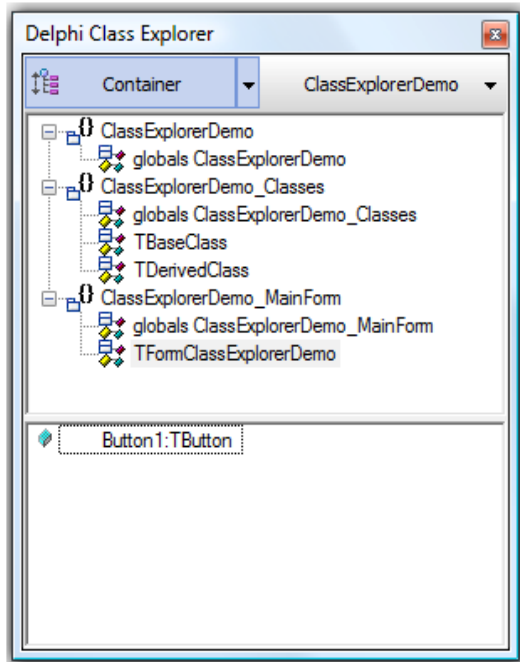
Delphi 2009 的「Delphi Class Explorer」窗格是全新窗格 (可從「View」功能表下的「Delphi Class Explorer」項目中取得)。「Delphi Class Explorer」會提供專案整體的代表符號，這與透過圖形方式 (比較類似的方式) 顯示單一單元項目的「Structure View」不同。

在「Delphi Class Explorer」的第一個層級，您會看到管理每個單元全域定義的節點清單 (以及專案檔)，其他節點則顯示專案中定義的所有類別：

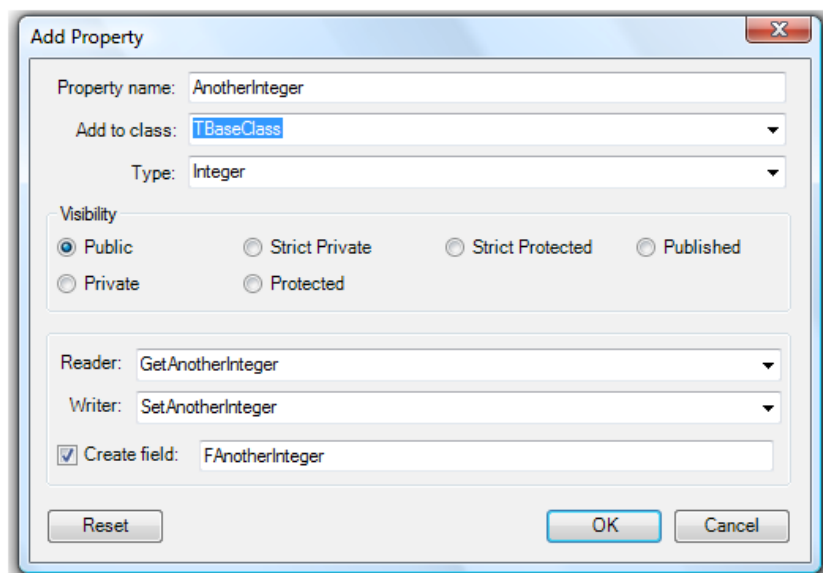


在各個類別中，您可以看到特定成員 (非繼承成員)，以及與其他類別的關係。這會根據選取的第一個工具列按鈕來進行說明：「Base to derived」(如上所示)、「Derived to base」或「Container」。在最後一種情況中，會依照單元來區分類別 (與全域)，不會顯示繼承關係：

您可使用本機功能表將新欄位加入類別、新作業（或方法，包括建構函式與解構函式）或屬性，如下圖所示：



以正確的 Delphi 方式可以新增屬性（比以 UML 為基礎的塑模更有用）。雖然您可以新增屬性並要求建立對應欄位，以進行直接欄位對應，但是此工具會對應到 `setter` 和 `getter` 方法。



「Delphi Class Explorer」會將以下一行新增至類別，如先前螢幕擷取畫面所示：

```
type
  TBaseClass = class
    strict private
      function GetAnotherInteger : Integer;
      procedure SetAnotherInteger(val : Integer);
    public
      property AnotherInteger : Integer
        read GetAnotherInteger write SetAnotherInteger;
    strict private
  var
    FAnotherInteger:Integer;
  end;
```

我發現使用嚴格的私有 var 模塊相當怪異，但這是正確的，而且新增額外 var 關鍵字會更便於產生程式碼，而且降低風險。如果要將程式碼重新調整為我喜歡的格式，我就不會只宣告屬性，而會改為使用「Class Completion」，這會產生更清楚且更標準的 Delphi 程式碼。對我來說，如果要瀏覽專案的原始程式碼，「Delphi Class Explorer」是個相當實用的工具，除非我需要產生 UML 圖表，否則我不會使用「Model View」。

其他新功能

如果您不考慮整個 IDE 以 Unicode 啟動的事實，更新的「Project Options」對話方塊、「Project Manager」的新功能與延伸建置組態、改善的資源支援以及「Class Explorer」應該是 Delphi 2009 中 IDE 最顯著的最新功能。

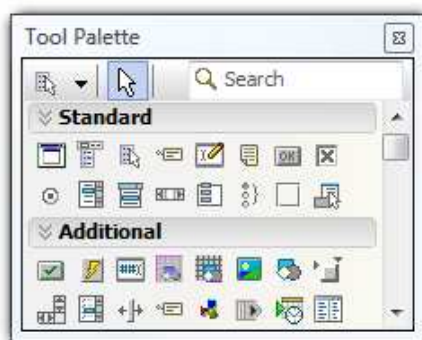
這一節列出一些能協助您每天在 Delphi 開發環境下工作的許多其他次要功能。其他顯著的 IDE 改善之處與下列有關：

- 「Integrated Translation Manager (ITM)」，其中已針對 Unicode 支援進行修訂，並且在許多方面大幅改善。
- IDE 中的變更與 COM 支援以及「Type Library」編輯器中的大幅變更有關。

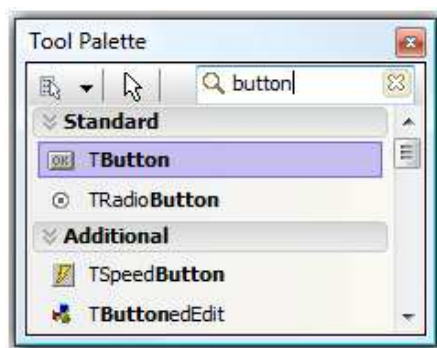
「TOOL PALETTE」搜尋方塊

在 Delphi 2006 中，選取「Tool Palette」時，可以輸入字詞，篩選出以這些字詞起首的元件 (不包括以 T 起首的項目)。在 Delphi 2007 中，您也可以進行相同動作，不過，您也可以選取元件名稱中的文字，因此，您可以輸入更為明顯的 HTTP 來選擇 idHTTP。

在 Delphi 2009 中，「Tool Palette」的運作方式也與在 Delphi 2007 中相同，只是使用者介面不同，對所有使用者而言，看起來更為清楚，可便於搜尋元件清單：



選取工具板後 (快捷鍵為 **Ctrl+Alt+P**)，就可以在搜尋方塊中輸入 (而非在窗格標題中輸入)，「Tool Palette」便會篩選顯示的元件：

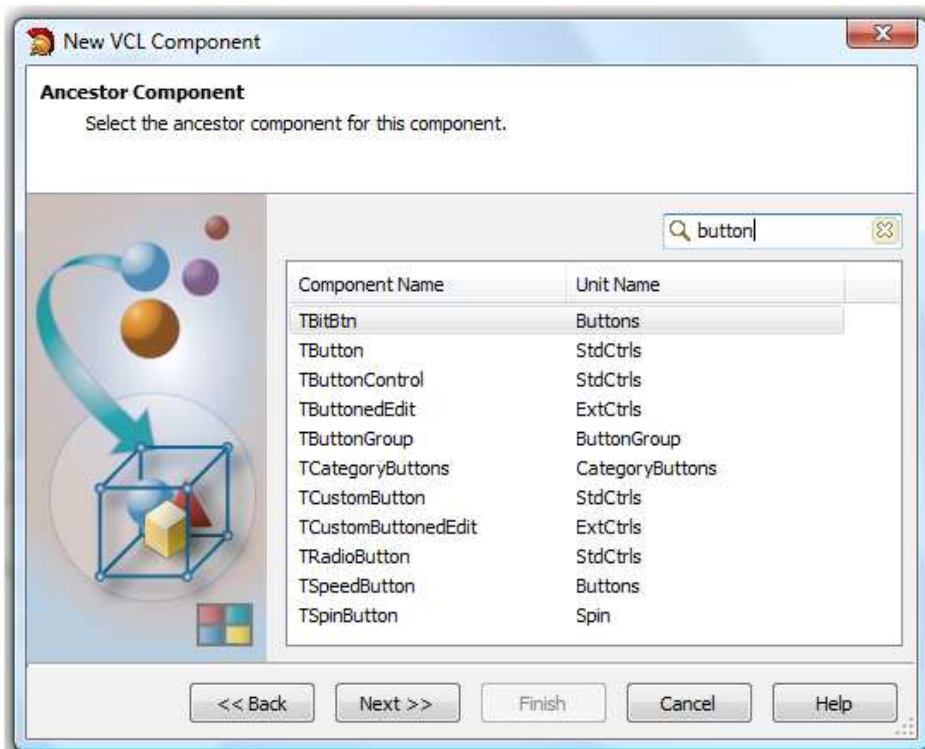


在「Tool Palette」中還有另外一項改變。先前如果要看到清單底部的類別，需要不斷捲動窗格，許多人對此感到不便，現在窗格則預設為自動收合。您可以用另一種方法來調整窗格，就是在選取元件後，選擇是否保留目前的搜尋方塊。

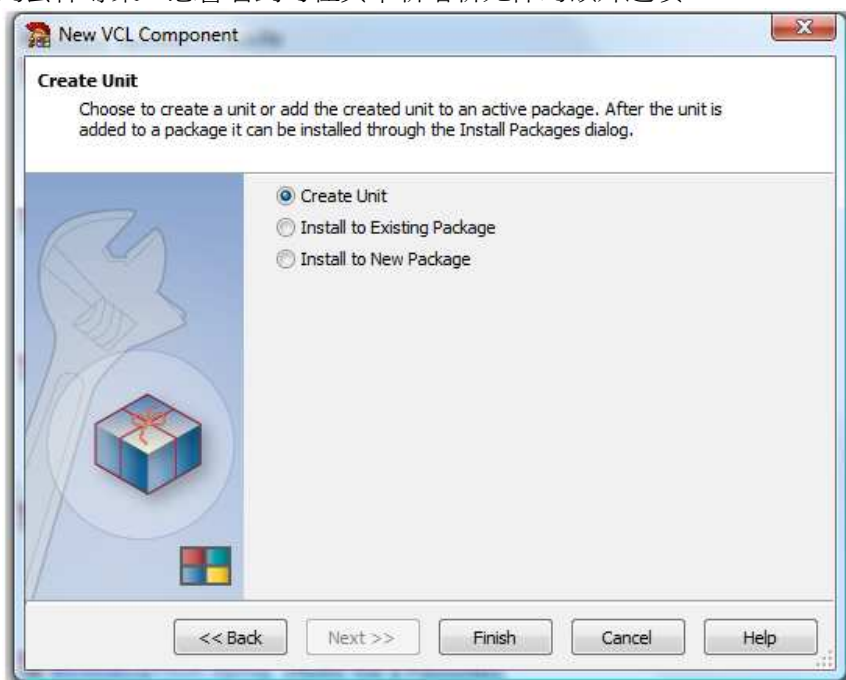
更新的「COMPONENT」精靈

用來建立新的 VCL 元件或匯入元件 (ActiveX 控制項或 .NET Assembly，以類似 COM 控制項的方式使用) 的對話方塊也有所改善，成為多重步驟精靈。建立空元件架構或包裝外部控制項的實際能力並未大幅修改。唯一新增的部分，是能夠將元件安裝到現有套件或必須命名的新套件 (兩個精靈均適用)。

相關變更出現在精靈的使用者介面中，您可以從 IDE 的「Components」功能表啟動。舉例來說，「New VCLComponent」精靈的起始頁面有一個用來篩選基本類別元件的搜尋方塊，這是繼承自：



進行時，輸入類別名稱與其他標準細節，就會出現最後的頁面，
您可在頁面中建立新套件，或將新元件新增至現有套件：
如果有作用中的套件專案，您會看到可在其中新增新元件的額外選項。

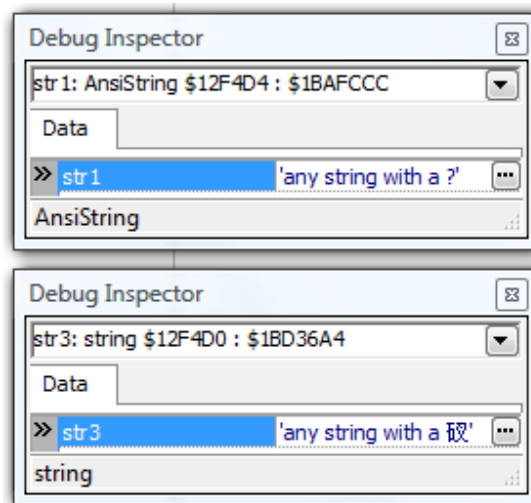


在「Import Component」精靈中，也增加了類似的功能。

除錯器

和 IDE 的其他部分一樣，除錯器也完全支援 Unicode。在過去的版本中，除錯器僅部分支援 Unicode，但在 Delphi 2009 中已經加以延伸。

舉例來說，如果您利用「Run」|「Inspect」(或編輯器本機功能表的「Debug」|「Inspect」)檢查字串變數，您不僅會得到正確的 Unicode 值，底部的指示也會告訴您變數的實際字串類型。以下是 AnsiString 與 UnicodeString (簡稱為「String」) 在「Inspect」窗格中的比較：



兩個視窗都顯示相同的字串，但因為出現中文字元，所以第一個視窗無法經過正確轉換。

除錯器的其他功能則與支援 Unicode 無關，例如 CPU 檢視可支援 SSE3 與 SSE4 指令 (對和我一樣很少使用組合語言的人來說，這不是主要的功能)。

一個雖然相當低階但很有意思的功能是，除錯器支援 Vista (以及 Windows Server 2008) 的「等候鍵周遊」功能。如需詳細資訊，請參閱說明作業系統層級「等候鍵周遊」的 MSDN 技術文件，以及由 CodeGear 的 Chris Hesik 撰寫並張貼於部落格的文件，此文件說明 Delphi 除錯器的此項功能。這兩份文件的 URL 為：

<http://msdn.microsoft.com/en-us/library/ms681622.aspx>
<http://blogs.codegear.com/chris hesik/2008/07/21/34833>

「Threads Status」有一個額外欄位，會顯示造成死結的多個執行緒，可協助您瞭解您的多重執行緒應用程式。

除錯與全新語言功能

雖然除錯器看起來與先前版本相當近似，此版本的工作人員仍花了很大的功夫，讓使用者能針對使用泛型與匿名方法的應用程式進行除錯。由於精細的程式碼產生是在背景完成，因此您正在進行除錯的程式碼會與您原先所撰寫的大不相同。儘管可能會出現一些錯誤限制，但是對於全新語言除錯的功能已經相當完備，這是一項不容易達到的成就。

專案管理交由 DELPHI 進行

在過去的數個版本中，Delphi IDE 已經有許多顯著的改善，但 Delphi 2009 仍持續進行改善。最顯著的改善，我想應該是 IDE 不斷提升的穩定性。其次則是大幅度延伸的專案管理功能，其中包括支援階層式建置組態、可從一個專案移到另一個專案的選項設定、整合式資源管理，以及可按照使用者的喜好來設定此窗格的多重「Project Manager」檢視。大幅改善的組態管理能讓使用者輕鬆執行大型專案，即使與經典的 Delphi 7 相較也毫不遜色。我確信，這些與 Delphi 開發者每日工作有關的新功能會產生相當顯著的影響。因此，我認為相當值得升級至 Delphi 2009 IDE。

關於作者

本白皮書是由暢銷系列 *Mastering Delphi* 的作者 Marco Cantù 為 Embarcadero Technologies 進行撰寫，其中內容摘錄自該作者最新的 *Delphi 2009 Handbook* 一書，詳見 <http://www.marcocantu.com/dh2009>。您可以在 Marco 的部落格 (<http://blog.marcocantu.com>) 閱讀關於他的相關訊息，並且寄送電子郵件到 marco.cantu@gmail.com 與他交流。



Embarcadero Technologies Inc. 能夠讓應用程式開發者與資料庫專業人員在所選擇的環境中，藉由工具設計、建立與執行軟體應用程式。全球超過三百萬使用者的社群以及「財富」雜誌一百大公司中的九十家公司都是仰賴 Embarcadero 的 DatabaseGear™ 與 CodeGear™ 系列產品增加生產力、公開協力合作與自由創新。Embarcadero 成立於 1993 年，總部位於加州舊金山，全球各地都設有辦公室。Embarcadero 的網址為 www.embarcadero.com。公司的旗艦版 DatabaseGear 工具包括：ER/Studio®、DBArtisan®、Rapid SQL® 與 Embarcadero® Change Manager™。