

# VCL For Web和JSON應用程式- Part II C++Builder版

在多方的努力下，最近興德終於開始建立了 CodeGear 技術網站的內容，開始提供有關 CodeGear 產品中文的技術文章以及研討會相關的資料，雖然在起步的階段 CodeGear 技術網站的內容仍然需要加強，但我相信只要持續的增加技術內容，這個 CodeGear 技術網站將可以有效的幫助台灣的 CodeGear 開發人員。

由於目前 CodeGear 技術網站的資源不是很充足，因此我知道現在興德是由 Web Master 以人工的方式來維護 CodeGear 技術網站，例如我前幾天到 CodeGear 技術網站看到了如下的內容 ([http://www.sinter.com.tw/codegear/codegear\\_technique.html](http://www.sinter.com.tw/codegear/codegear_technique.html)):



圖 1 興德提供的 CodeGear 技術網

興德開始在 CodeGear 技術網站提供我的技術文章和以前舊的書籍提供有需要的網友下載。由於 CodeGear 技術網站的內容仍然在建構之中，因此還不太複雜，這樣的特性剛好可以讓我們使用來做為說明如何使用 JSON 伺服器 and VCL For Web 來自動建立 CodeGear 技術網站的內容。我們將使用 C++Builder 2009 的 DataSnap 和 VCL For Web 這兩項技術來說明如何建立類似如上圖的 CodeGear 技術網站。

## 使用的開發環境

---

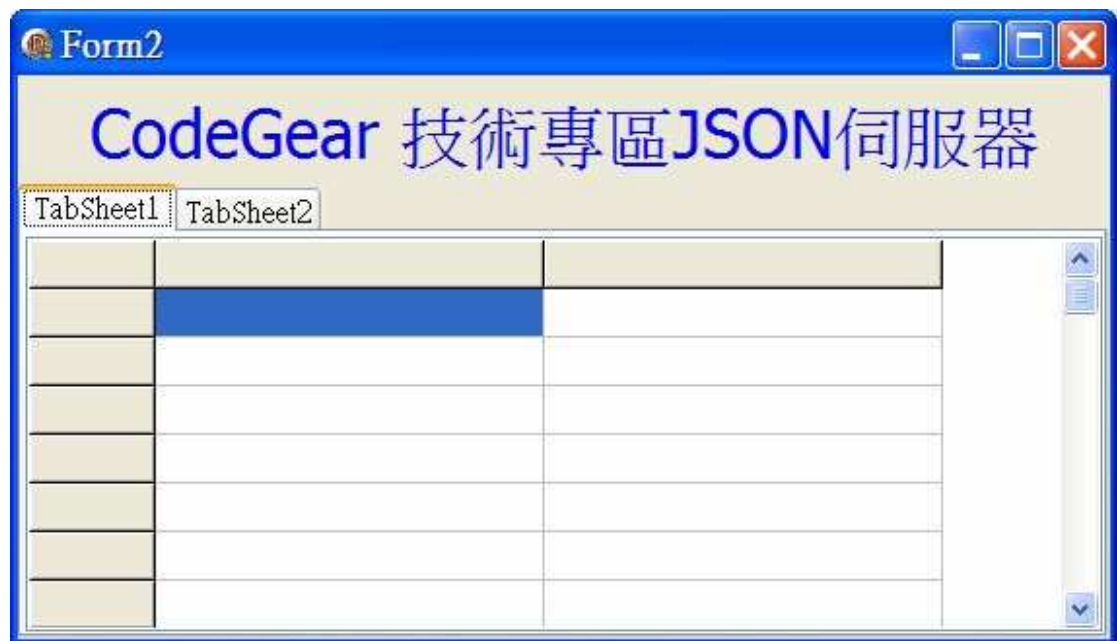
- C++Builder 2009 , Delphi 2009
  - DataSnap 2009
  - VCL For Web
  - dbExpress 4
- MS SQL Server 2005
- ER/Studio
- DBArtisan

## 範例 JSON 伺服器

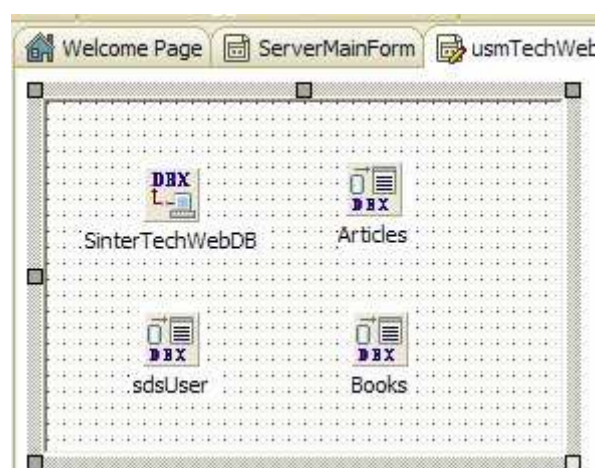
---

由於前面的文章已經說明過如何開發 JSON 伺服器，因此在本文中將只說明 JSON 伺服器提供的服務。

首先讓我們建立一個 JSON 伺服器，這個 JSON 伺服器將提供服務給稍後 VCL For Web 建立的 Web 應用程式呼叫使用。



接著建立一個 ServerModule，並且在其中放入 dbExpress 元件，連結到範例 MS SQL Server:



下面是 ServerModule 中 TSQLDataSet 的特性設定值:

特性名稱	特性值
Name	Articles
CommandText	select * from "Articles" where PID = :PID order by ADate DESC
CommandType	ctQuery

特性名稱	特性值
Name	Books
CommandText	select * from "Books" where PID = :PID
CommandType	ctQuery

特性名稱	特性值
Name	sdsUser
CommandText	select count(*) from "Users" where Name = :Name and Password = :Password
CommandType	ctQuery

在這個範例 JSON 伺服器中提供了三個服務:

```
function GetArticles(PID : Integer) : TDataSet;
function GetBooks(PID : Integer) : TDataSet;
function ValidateUser(AUserName : string; APassword : string) : Boolean;
```

**GetArticles** 方法可提供用戶端藉由產品 ID(參數 PID)來查詢相關的文章資料，**GetBooks** 方法可根據產品 ID(參數 PID)來查詢書籍相關的資料，而 **ValidateUser** 則可根據用戶端傳遞來的使用者名稱和密碼來查詢使用者是否為系統中合法的使用者。

下面分別是這些方法的實作程式碼:

```
001 function TdsmTechWeb.GetArticles(PID : Integer): TDataSet;
002 begin
003     Self.SINTERTECHWEBDB.Connected := True;
004     Self.Articles.Active := False;
005     Self.Articles.ParamByName('PID').AsInteger := PID;
006     Self.Articles.Active := True;
007     Result := Self.Articles;
008 end;
009
```

```

010 function TdsmTechWeb.GetBooks(PID: Integer): TDataSet;
011 begin
012     Self.SINTERTECHWEBDB.Connected := True;
013     Self.Books.Active := False;
014     Self.Books.ParamByName('PID').AsInteger := PID;
015     Self.Books.Active := True;
016     Result := Self.Books;
017 end;
018
019 procedure TdsmTechWeb.SetupDatabaseWorks;
020 begin
021     //
022 end;
023
024 function TdsmTechWeb.ValidateUser(AUserName, APassword: string):
Boolean;
025 begin
026     Self.SINTERTECHWEBDB.Connected := True;
027     Self.sdsUser.Active := False;
028     Self.sdsUser.ParamByName('Name').AsString := AUserName;
029     Self.sdsUser.ParamByName('Password').AsString := APassword;
030     Self.sdsUser.Active := True;
031     Result := (Self.sdsUser.Fields[0].AsInteger = 1) ;
032 end;

```

這些實作程式碼都非常的簡單，唯一需要說明的是，由於上面的每一個方法都需要重新開啓一次和資料庫的連結，因此爲了執行效率，開發人員應該開啓 `dbExpress` 的連結池功能。

另外需要注意的地方就是這些方法如何從用戶端取得需要的參數，稍後我們在實作 Web 應用程式時會說明。

現在編譯並且執行這個 JSON 伺服器並且準備實作 Web 應用程式來使用它。

## 使用 VCL For Web 開發 CodeGear 技術網站

---

要使用 VCL For Web 開發出類似圖 1 的 Web 應用程式，我們可以使用 VCL For Web 提供的樣版功能來進行，VCL For Web 的樣版功能可以讓開發人員結

合 Web 美工人員設計的網頁結果和程式碼執行的結果於一體，如此一來就可以開發出又美觀又可具備動態內容的網頁內容。

要使用 VCL For Web 的樣版功能非常的簡單，我們只需要使用下面的步驟即可：

1. 由美工人員使用 Web 設計軟體來設計 Web 頁面 HTM 或是 HTML 檔案，
2. 在步驟 1 設計的 Web 頁面中需要由 VCL For Web 應用程式程式碼產生的動態內容的地方，可由美工設計人員使用如下的格式來暫時表示：

{%將由 VCL For Web 動態產生的內容%}

也就是說在 Web 頁面中暫時使用{%和%}包圍稍後將由 VCL For Web 程式碼動態產生內容的地方，例如在圖 1 中興德技術網提供了 Delphi 和 C++Builder 的技術文章，那麼我們可以在原本 Web 頁面中使用下面的方法暫時代表，稍後 VCL For Web 的樣版功能將取代下面的 {%iwtxtDelphiArticles%} 為實際 Delphi 文章的內容，而 {%iwtxtBCBArticles%}將被實際的 C++Builder 文章的內容取代：

```
<table cellSpacing="0" cellPadding="0" width="100%" border="0">
  <tbody>
    <tr>
      <td bgColor="#f0f0f0">
        <h3>Delphi</h3>
      </td>
    </tr>
    {%iwtxtDelphiArticles%}
    <tr>
      <td bgColor="#f0f0f0">
        <h3>C++Builder</h3>
      </td>
    </tr>
    {%iwtxtBCBArticles%}
  </tbody>
</table>
```

3. 在美工 Web 頁面檔案設計完成之後，交由開發人員使用 VCL For Web 來動態取代步驟 2 中使用{%和%}包圍的內容。開發人員首先使用

TIWTemplateProcessorHTML 元件載入美工 Web 頁面，再使用 VCL For Web 元件來取代{%和%}包圍的內容即可。

在下面的內容中我們將一一的說明如何實作完成上面的步驟。

## 步驟 1, 2 設計樣版美工 Web 檔案

---

一開始讓我們設計一個類似圖 1 的 Web 網頁，如下所示：



圖 2 樣版美工 Web 網頁

這是一個正常的 HTML 檔案，沒有任何特別的內容。由於我們希望上圖 Web 網頁中 Delphi 和 C++Builder 文章的內容能夠動態由資料庫中產生，而不是每次都需要修改 HTML 檔案的內容，因此讓我們先修改圖 2 樣版美工 Web 網頁如下：



圖 3 加入樣版標籤的樣版美工 Web 網頁

我們把圖 2 中 Delphi 和 C++Builder 文章的靜態內容，以及 Delphi 和 C++Builder 書籍的靜態內容都改成使用樣版標籤來取代，在下面我只列出在 Web 網頁 HTML 檔案中修改的部份：

```

001 <table cellSpacing="0" cellPadding="0" width="100%" border="0">
002 <tbody>
003 <tr>
004 <td bgColor="#f0f0f0">
005 <h3>Delphi</h3>
006 </td>
007 </tr>
008 {%iwtxtDelphiArticles%}
009 <tr>
010 <td bgColor="#f0f0f0">
011 <h3>C++Builder</h3>
012 </td>
013 </tr>
014 {%iwtxtBCBArticles%}
015 </tbody>
016 </table>
017 <div align="right">&nbsp;

```

```

018
019 </div></div>
020 <div class="space"
021 style="MARGIN-TOP: 30px; FONT-SIZE: 1em; LINE-HEIGHT: 20px;
FONT-FAMILY: Arial, Helvetica, sans-serif">
022 <h2 style="PADDING-TOP: 10px" align="center"><span
class="style10">李維老師書籍下載
023 </span>
024 </h2>
025 <table cellSpacing="0" cellPadding="0" width="100%" border="0">
026 <tbody>
027 <tr>
028 <td bgColor="#f0f0f0">
029 <h3>Delphi</h3>
030 </td>
031 </tr>
032 {%iwtxtDelphiBooks%}
033 <tr>
034 <td bgColor="#f0f0f0">
035 <h3>C++Builder</h3>
036 </td>
037 </tr>
038 {%iwtxtBCBBooks%}
039 </tbody>
040 </table>

```

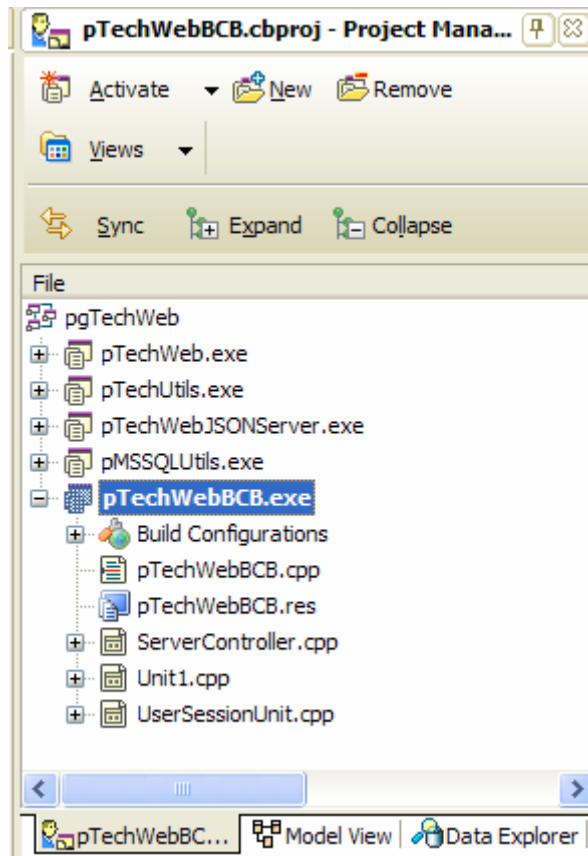
從上面的程式碼中我們可以看到，我們使用：

行數	樣版標籤	目的
008	{%iwtxtDelphiArticles%}	動態產生 Delphi 文章內容
014	{%iwtxtBCBArticles%}	動態產生 C++Builder 文章內容
032	{%iwtxtDelphiBooks%}	動態產生 Delphi 書籍內容
038	{%iwtxtBCBBooks%}	動態產生 C++Builder 書籍內容

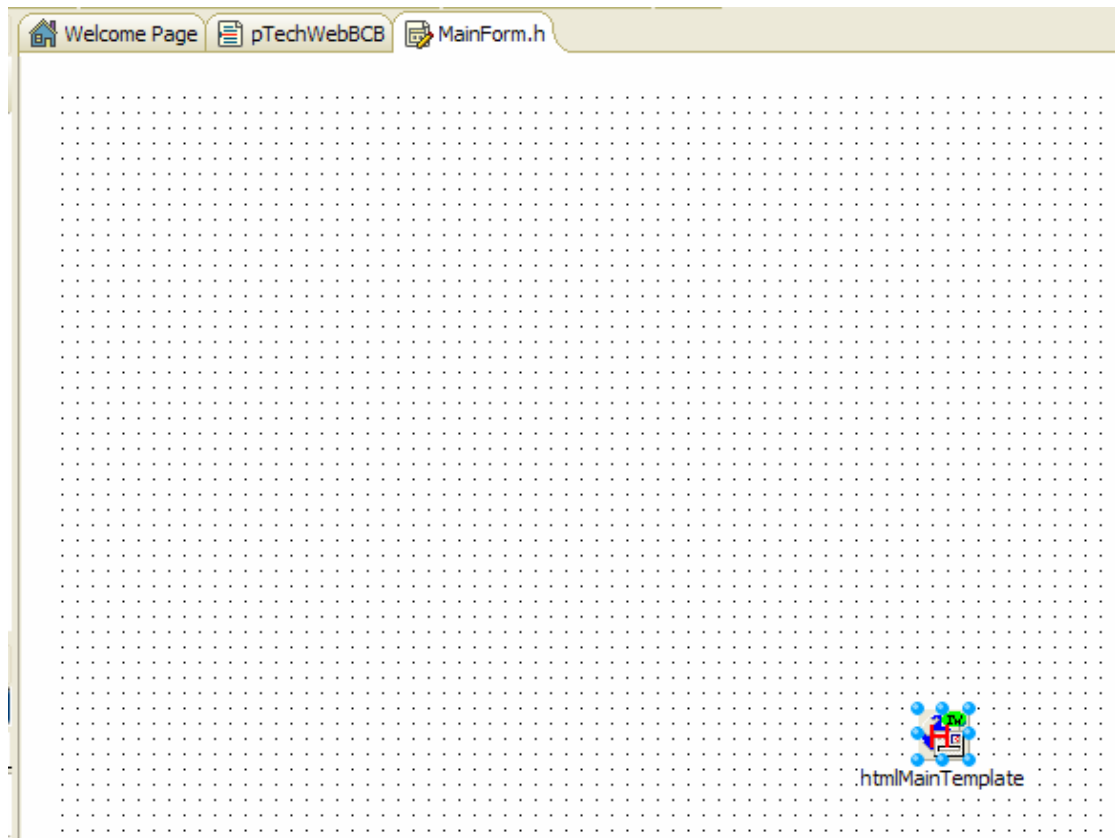
因此上面的程式碼中我們只是把原先靜態的 HTML 內容改由樣版標籤來取代，因為稍後我們將使用 VCL For Web 的程式碼直接再取代樣版標籤。

### 步驟 3 使用 VCL For Web 動態取代樣版內容

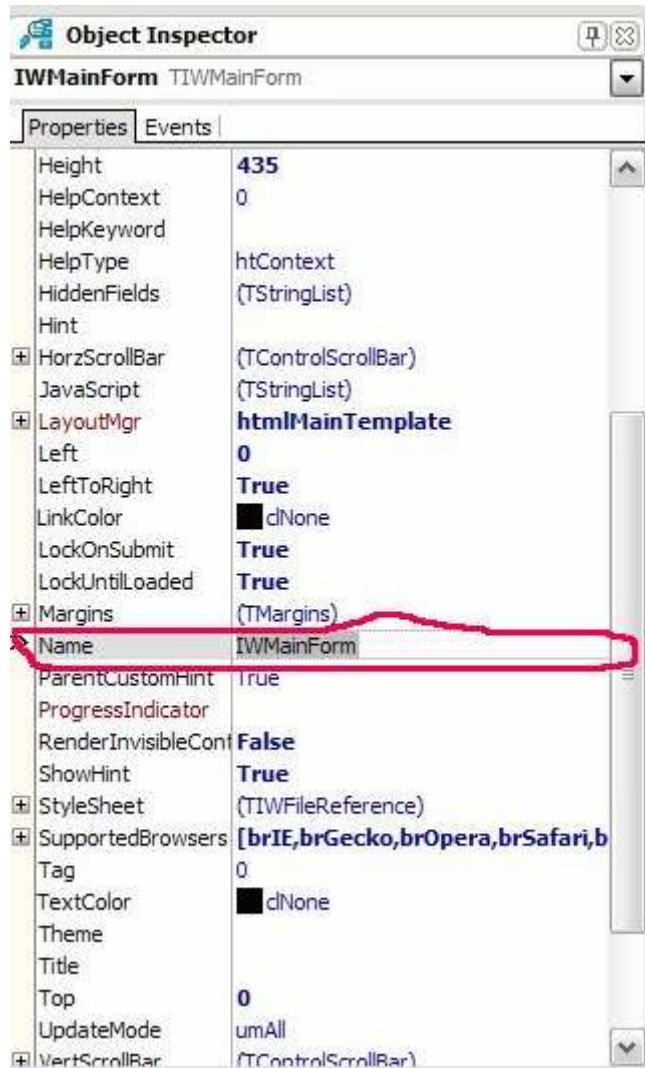
首先建立一個 VCL For Web 應用程式，接著如下所示把 VCL For Web 的主表單改名為 MainForm.pas：



開啓 `MainForm`，在其中放入 `TIWTemplateProcessorHTML` 元件，接著設定主表單的 `LayoutMgr` 特性為這個新加入的 `TIWTemplateProcessorHTML` 元件。`TIWTemplateProcessorHTML` 元件可以載入和表單同名的樣版 Web 網頁檔案，現在我們就可以使用這個元件載入我們在前面圖 3 修改的樣版 Web 網頁。



在這裡一般開發人員容易出錯的地方就是由於 `TIWTemplateProcessorHTML` 元件是載入和表單同名的樣版 Web 網頁檔案，因此我們設計的樣版 Web 網頁的 HTML 檔案名稱必須和 `TIWTemplateProcessorHTML` 元件所在的 VCL For Web 表單同名，否則 `TIWTemplateProcessorHTML` 是無法載入樣版 Web 網頁檔案。因此，現在讓我們使用物件檢視器把 `MainForm` 表單名稱設定為 `IWMainForm`，如下所示：



由於 TIWTemplateProcessorHTML 元件是位於名稱為 IWMainForm 的表單中，因此 TIWTemplateProcessorHTML 會自動載入名為『IWMainForm.HTML』的樣版 Web 檔案，因為這個原因，我們需要把前面圖 3 修改的樣版 Web 檔案儲存為 IWMainForm.HTML，並且需要儲存在 VCL For Web 應用程式的 Templates 子目錄之中。一般來說 VCL For Web 應用程式需要兩個子目錄來儲存不同目的檔案，下面的表格說明了這兩個子目錄的目的：

VCL For Web 子目錄	目的
Templates	儲存樣版 Web 網頁檔案
Files	儲存 Web 網頁其他需要的檔案，例如圖形檔案或是 JavaScript 檔等

下圖是這個範例 VCL For Web 應用程式的目錄，從下面的圖形中就可以看到在專案之下建立了 Files 和 Template 這兩個子目錄以儲存不同目的檔案：

Name	Ext	Size	Date	Attr
..		<DIR>	2009/02/10 19:33	----
.._history		<DIR>	2009/02/10 19:33	--h-
FILES		<DIR>	2009/01/20 20:04	----
Templates		<DIR>	2009/01/24 13:44	----
MainForm	cpp	951 b	2009/02/10 19:32	-a--
MainForm	dfm	3.8 k	2009/02/10 19:33	-a--
MainForm	h	1.1 k	2009/02/10 19:33	-a--
pTechWebBCB	cbproj	9.2 k	2009/02/10 19:29	-a--
pTechWebBCB	cpp	948 b	2009/02/10 19:29	-a--
pTechWebBCB	res	5.1 k	2009/02/10 19:27	-a--
pTechWebBCB.cbproj	local	1.1 k	2009/02/10 19:29	-a--
ServerController	cpp	954 b	2009/02/10 19:27	-a--
ServerController	dfm	1.2 k	2009/02/10 19:27	-a--
ServerController	h	0.9 k	2009/02/10 19:27	-a--
UserSessionUnit	cpp	720 b	2009/02/10 19:27	-a--
UserSessionUnit	dfm	100 b	2009/02/10 19:29	-a--
UserSessionUnit	h	665 b	2009/02/10 19:27	-a--

而在 **Templates** 子目錄中儲存了名為 **IWMainForm.html** 的樣版網頁，這也就是前面圖 3 修改的樣版 Web 網頁：

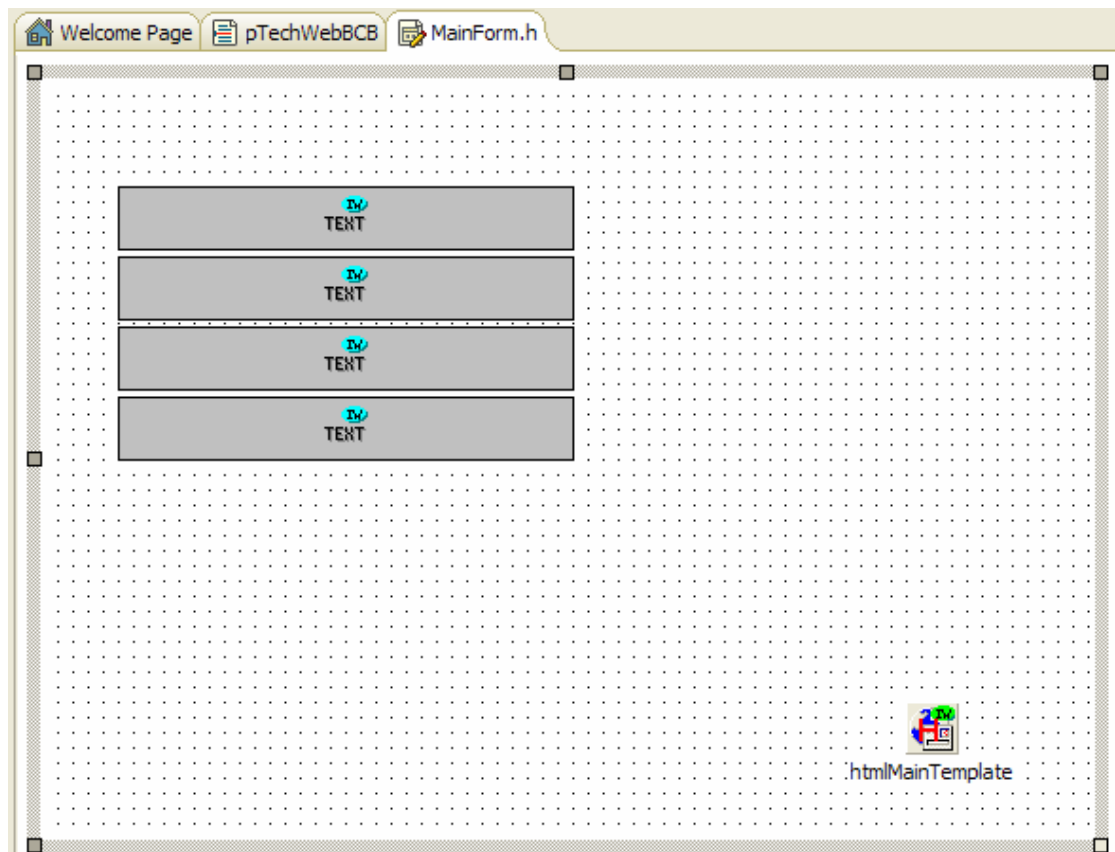
Name	Ext	Size	Date	Attr
..		<DIR>	2009/02/10 19:39	----
.._history		<DIR>	2009/01/20 20:04	--h-
IWMainForm	html	3.9 k	2009/01/24 13:44	-a--

下圖則是 **Files** 子目錄，其中儲存了樣版 Web 網頁使用的圖形檔案：

Name	Ext	Size	Date	Attr
..		<DIR>	2009/01/20 20:04	----
CG_technique_banner	gif	23.7 k	2009/01/18 16:39	-a--
icon_pdf	gif	1.0 k	2009/01/18 16:39	-a--
icon_zip	gif	599 b	2009/01/18 16:39	-a--

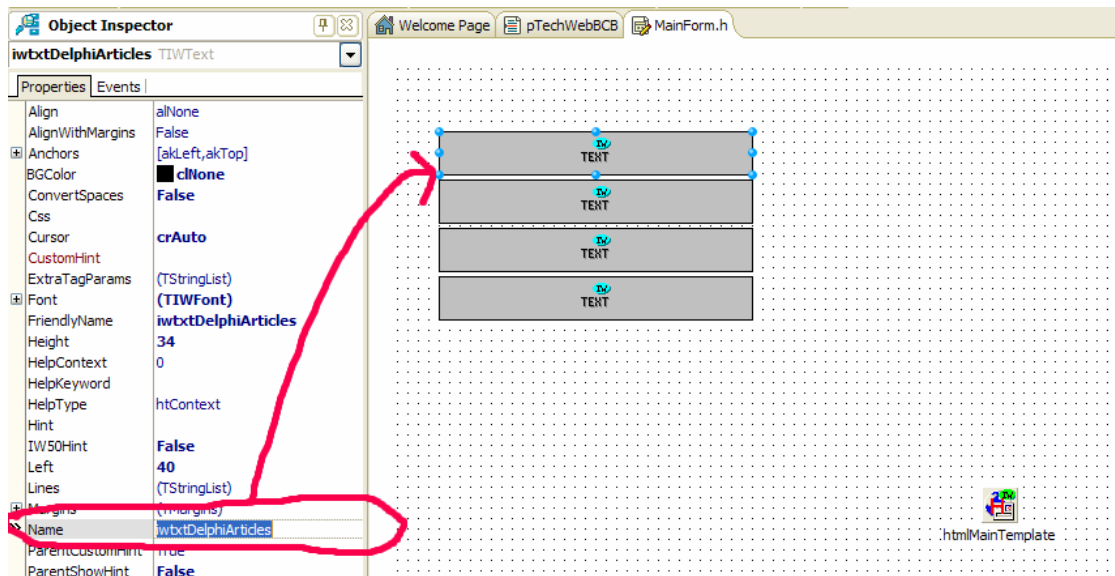
備註：由於 **C++Builder** 會把編譯的執行檔產生在 **Debug** 或是 **Release** 子目錄中，因此 **Files** 和 **Templates** 子目錄必須和執行檔位於同一層的目錄中才能夠正確的執行。

正確的完成了上述的步驟之後我們就可以開始開發範例 Web 應用程式了，首先在主表格中放入四個 **TIWText** 元件如下所示：



爲什麼使用 **TIWText** 元件？因爲我想使用這些 **TIWText** 元件動態產生的內容取代樣版 Web 網頁中使用的樣版標籤，**TIWText** 元件可以讓開發人員在它的 **Lines** 特性中產生動態內容，然後我們再藉由 **VCL For Web** 的樣版功能把 **TIWText** 的 **Lines** 特性值取代樣版 Web 網頁中使用的樣版標籤。要如此做，我們需要把 **TIWText** 元件的名稱設定爲想取代的樣版標籤相同的名稱。例如在前面我們使用了樣版標籤 `{% iwtxtBCBArticles%}` 來暫時代表 **C++Builder** 的文章內容，因此我們需要設定 **TIWText** 元件的名稱爲 `iwtxtBCBArticles`，如此一來在 **VCL For Web** 應用程式執行時就會把這個 **TIWText** 的 **Lines** 特性值取代樣版 Web 網頁中 `{% iwtxtBCBArticles %}` 地方的內容。

由於前面樣版 Web 網頁中使用了四個樣版標籤，因此讓我們在主表單中放入四個 **TIWText** 元件：



並且分別設定這四個 TIWText 元件的特性如下：

特性	說明
Name	iwtxtDelphiArticles
RawText	True
WantReturns	True

特性	說明
Name	iwtxtBCBiArticles
RawText	True
WantReturns	True

特性	說明
Name	iwtxtDelphiBooks
RawText	True
WantReturns	True

特性	說明
Name	iwtxtBCBBooks
RawText	True
WantReturns	True

從上面的設定我們可以看到每一個 TIWText 元件的名稱都設定為它要取代的樣版標籤相同的名稱，至於 RawText 特性的意思是由於我們希望 TIWText

元件的 **Lines** 特性值直接取代樣版 Web 網頁中樣版標籤的內容，因此我們需要直接產生 HTML 的內容，設定 **TIWText** 的 **RawText** 特性值為 **True** 代表我們要求 **VCL For Web** 直接把 **TIWText** 的 **Lines** 特性值中的內容拷貝和取代到樣版 Web 網頁中樣版標籤的內容，不需要 **VCL For Web** 進行任何的處理，這也就是說我們會直接在 **TIWText** 的 **Lines** 特性值中產生相關的 HTML 內容和標籤等字串值，稍後我們便會看。

設定好 **TIWText** 元件之後，我們就可以在表格的 **OnRender** 事件中撰寫如下的程式碼：

```
void __fastcall TIWMainForm::IWAppFormRender(TObject *Sender)
{
    取得文章資料();
    取得研討會資料();
    取得書籍資料();
}
```

**VCL For Web** 表單的 **OnRender** 事件會在 Web 表單顯示在瀏覽器之前被呼叫，因此在這個事件處理函式中我們呼叫三個函式分別取得文章，研討會和書籍資料並且取代樣版標籤以顯示動態內容。

先看看取得文章資料函式的實作程式碼：

```
void TIWMainForm::取得文章資料()
{
    //For Delphi
    執行取得文章資料(1, this->iwtxtDelphiArticles);
    //For C++Builder
    執行取得文章資料(2, this->iwtxtBCBArticles);
}
```

由於目前只有 **Delphi** 和 **C++Builder** 的技術文章，假設 **Delphi** 的文章在資料庫中是以 1 做為 ID，2 是 **C++Builder** 的技術文章，因此取得文章資料分別以不同的文章 ID 呼叫執行取得文章資料方法實際的從資料庫中取得文章資料。

執行取得文章資料方法首先在 005 行開啓和 JSON 伺服器的連結，呼叫開啓文章資料表方法並且傳遞文章 ID 給它以便從資料庫中取得文章資料，最後進入迴圈一一的從文章資料表中取出文章資料並且在 012 行使用必要的 HTML 標籤和文章名稱組成的字串加入到 **TIWText** 元件的 **Lines** 特性中。

```
001 void TIWMainForm::執行取得文章資料(const int IID, TIWText
```

```

*aTxtCtrl)
002  {
003      UnicodeString sResult;
004
005      開啓資料庫連結();
006      try
007      {
008          開啓文章資料表(iID);
009          aTxtCtrl->WantReturns = true;
010          while (!UserSession()->cdsArticles->Eof)
011          {
012              sResult = "<tr><td><strong>■</strong>" +
UserSession()->cdsArticles->Fields->FieldByName("Name")->AsStri
ng;
013              aTxtCtrl->Lines->Add(sResult);
014              UserSession()->cdsArticles->Next();
015          }
016      }
017      __finally
018      {
019          關閉文章資料表();
020          關閉資料庫連結();
021      }
022  }

```

還記得前面我們把 **TIWText** 的 **RawText** 特性值設定為 **True** 嗎？就是因為我們在上面的程式碼中自行使用 **HTML** 標籤而不需要 **VCL For Web** 為我們處理，因此我們需要設定 **RawText** 特性值設定為 **True**。

開啓資料庫連結和關閉資料庫連結方法只是開啓和關閉連結到 **JSON** 伺服器的 **TSQLConnection** 元件而已。

```

void TIWMainForm::開啓資料庫連結()
{
    UserSession()->sqlcnTechWeb->Connected = true;
}

void TIWMainForm::關閉資料庫連結()
{
    UserSession()->sqlcnTechWeb->Connected = false;
}

```

```
}
```

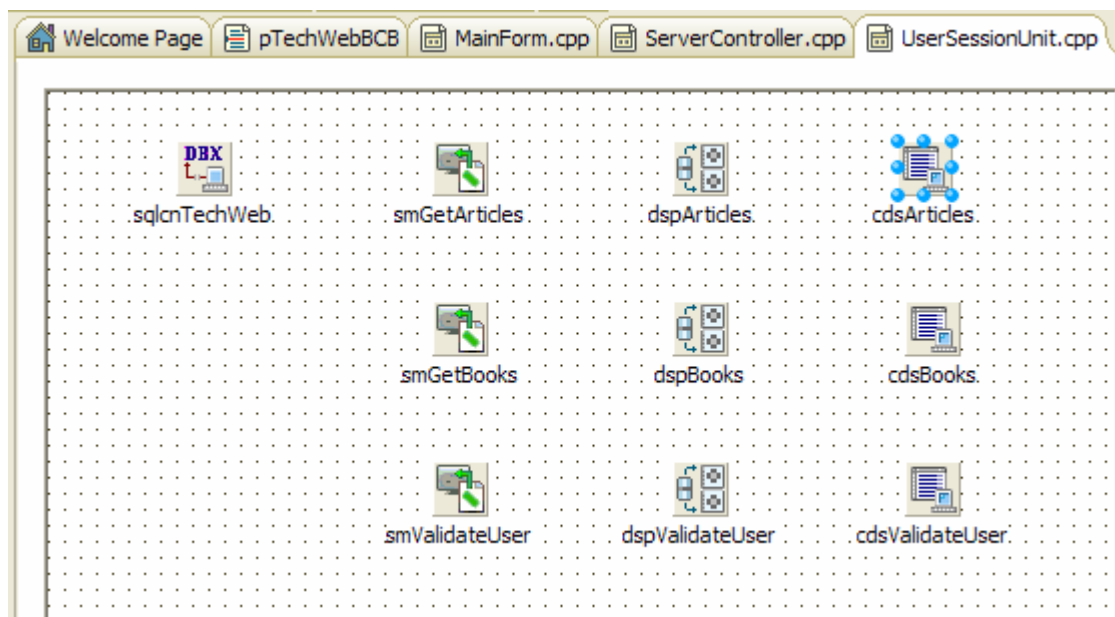
至於開啓文章資料表也很簡單，它只是把文章 ID 傳遞給 TClientDataSet 元件中的參數而已。

```
void TIWMainForm::開啓文章資料表(int iID)
{
    UserSession()->cdsArticles->Params->ParamByName("PID")->AsInteger =
iID;
    UserSession()->cdsArticles->Active = true;
}
```

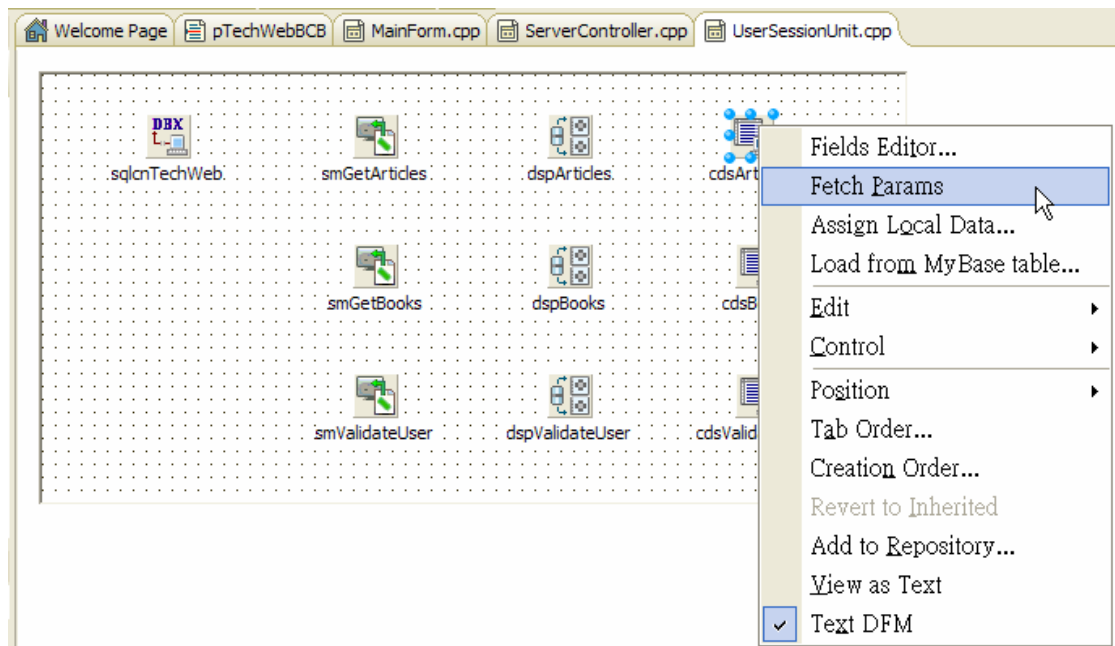
但是 cdsArticles 的參數又是從那麼來的？還記得前面的 JSON 伺服器嗎？它使用了如下的 SQL 從文章資料表中選取資料：

```
select * from "Articles" where PID = :PID order by ADate DESC
```

因此在這個 VCL For Web 應用程式中的 UserSessionUnit 中使用了如下的元件連結 JSON 伺服器(如果讀者不瞭解這些元件和設定，請參考 VCLForWeb 和 JSON 應用程式一文)：



我們只需在設計時期要點選 cdsArticles 元件，點選滑鼠右鍵，再選擇 Fetch Params 選項，如下所示，那麼只要此時 JSON 伺服器是在執行中，那麼 cdsArticles 元件便會自動從 JSON 伺服器中取得正確的參數資訊。



好了，瞭解了取得文章資料方法的實作原理之後，取得書籍資料方法也差不多，我就不多贅述了，我只列出它的實作程式碼如下：

```
void TIWMainForm::取得書籍資料()
{
    //For Delphi
    執行取得書籍資料(1, this->iwtxtDelphiBooks);
    //For C++Builder
    執行取得書籍資料(2, this->iwtxtBCBBooks);
}
void TIWMainForm::執行取得書籍資料(const int iID, TIWText *aTxtCtrl)
{
    UnicodeString sResult;

    開啓資料庫連結();
    try
    {
        開啓書籍資料表(iID);
        aTxtCtrl->WantReturns = true;
        while (!UserSession()->cdsBooks->Eof)
        {
            sResult = "<tr><td><strong>■</strong>" +
UserSession()->cdsBooks->Fields->FieldByName("Name")->AsString;
            aTxtCtrl->Lines->Add(sResult);
        }
    }
}
```

```
UserSession()->cdsBooks->Next();
}
}
__finally
{
    關閉書籍資料表();
    關閉資料庫連結();
}
}
```

現在我們就可以編譯並且執行這個範例 VCL For Web 應用程式了，下面的就是此時這個範例 VCL For Web 應用程式在 FireFox 中執行的結果：



各位可以看到這個結果和圖 1 興德提供的 CodeGear 技術網非常的類似的，只是少了文章和範例圖像以及連結，各位可以自行修改加入上這些少了的元素，非常簡單的。