

VCL For Web和JSON應用程式

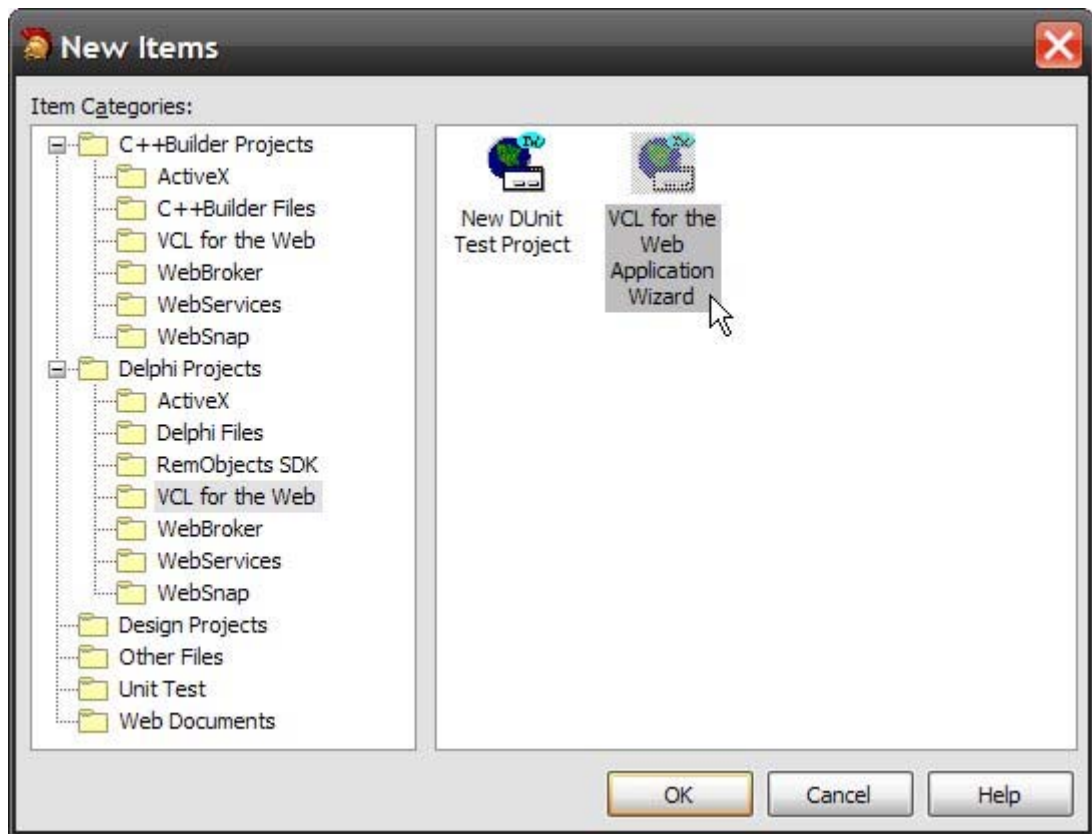
在前面的文章中我們已經說明了如何使用 Delphi 2009 建立分散式 JSON 應用程式，在本篇文章中我們將說明如何使用 VCL For Web 為 JSON 應用系統建立 Web 的用戶端應用程式。

在 JSON 應用系統中，一旦我們成功建立了 JSON 伺服器之後，我們就可以建立各種不同的用戶端來呼叫使用 JSON 伺服器提供的服務。VCL For Web 是 Win32 下建立 Web 應用程式最好用的框架之一，對於 Delphi/BCB 的開發人員來說，使用 VCL For Web 來開發 Web 應用程式是最方便，最快速的方法，因為 VCL For Web 的開發模式和 Delphi/BCB 開發人員使用 VCL 開發 Win32 應用程式非常的類似，開發人員只需要使用 VCL For Web 提供的元件，在 VCL For Web 的表單上以拖曳元件的 RAD 方式，再配合熟悉的 Delphi 或是 C/C++ 程式語言就可以完成開發 Web 應用程式的工作。對於更為進階，複雜的 Web 應用程式，VCL For Web 也提供了可結合 JavaScript 的能力來開發 AJAX 等型態的 Web 應用程式。

在本文中我們先從最簡單的 Web 應用程式來說明如何使用 VCL For Web 框架來開發 Web 應用程式，本文主要的目的是說明如何為 JSON 應用系統建立 Web 應用介面。

建立 VCL For Web 應用程式

在 Delphi IDE 中點選 File | New | Other...功能表，然後選擇建立 VCL for the Web Application Wizard 圖像：



VCL For Web 精靈即顯示下列的對話盒詢問有關欲建立的 Web 應用程式的細節，例如專案名稱和專案儲存的目錄等。此外會詢問開發人員欲建立的 Web 應用程式型態，下面的表格簡單的說明了 VCL For Web 提供的三種型態的 Web 應用程式：

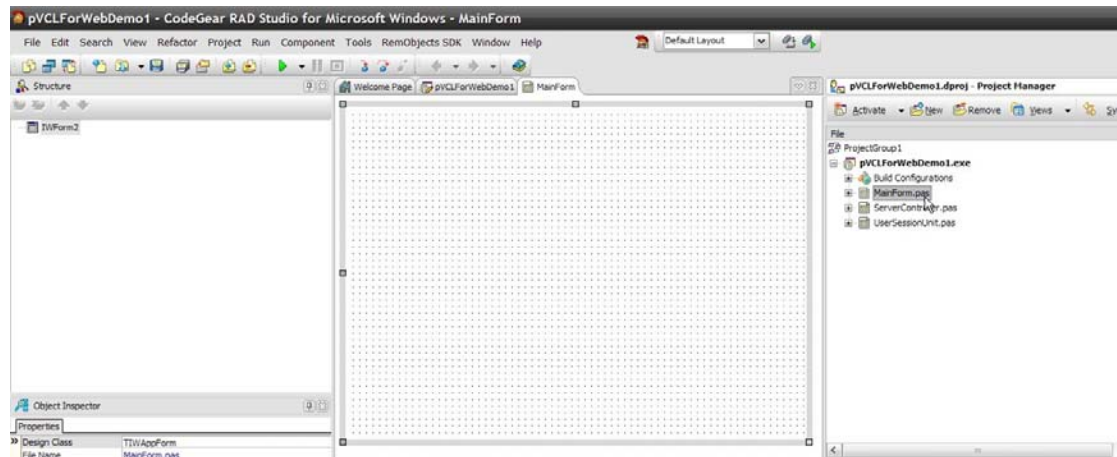
應用程式型態	說明
StandAlone	建立 VCL For Web 自屬的 EXE 型態的 Web 應用程式，VCL For Web 會執行一個 Launcher 應用程式再啟動瀏覽器並且執行 VCL For Web 應用程式。StandAlone 的應用程式主要是使用來開發之用，因為 StandAlone 的 Web 應用程式可以在 Delphi/BCB IDE 中進行除錯，開發人員應該在開發完畢之後再建立 Service 或是 ISAPI Extension 型態的 Web 應用程式以使用在實際工作環境中。
Service	把 Web 應用程式建立成 Window Service 的型態，一旦 Window 作業系統啟動之後 VCL For Web 應用程式便會自動啟動
ISAPI Extension	建立成 ISAPI 型態的 Web 應用程式，通常

是編譯成 dll 結尾的應用程式。ISAPI 型態的 Web 應用程式在 Window 環境中是執行速度最具效率的 Web 應用程式，因此開發人員在使用 VCL For Web 開發完畢之後應該再把 Web 應用程式編譯成此種型態的 Web 應用程式



此外在 Options 勾選盒群組中開發人員也可以選擇使用資料連結池來加快和增加 VCL For Web 應用程式的執行速度以及延展性，在內定上 VCL For Web 會選擇建立 User Session，稍後我們會說明什麼是 User Session。

現在讓我們接受 VCL For Web 設定的內定選項，點選 OK 按鈕之後，Delphi 即會建立 VCL For Web 的應用程式，在此專案中會包含三個項目，ServerController.pas，UserSessionUnit.pas 以及 Unit1.pas。讓我們把 Unit1.pas 改名為 MainForm.pas，

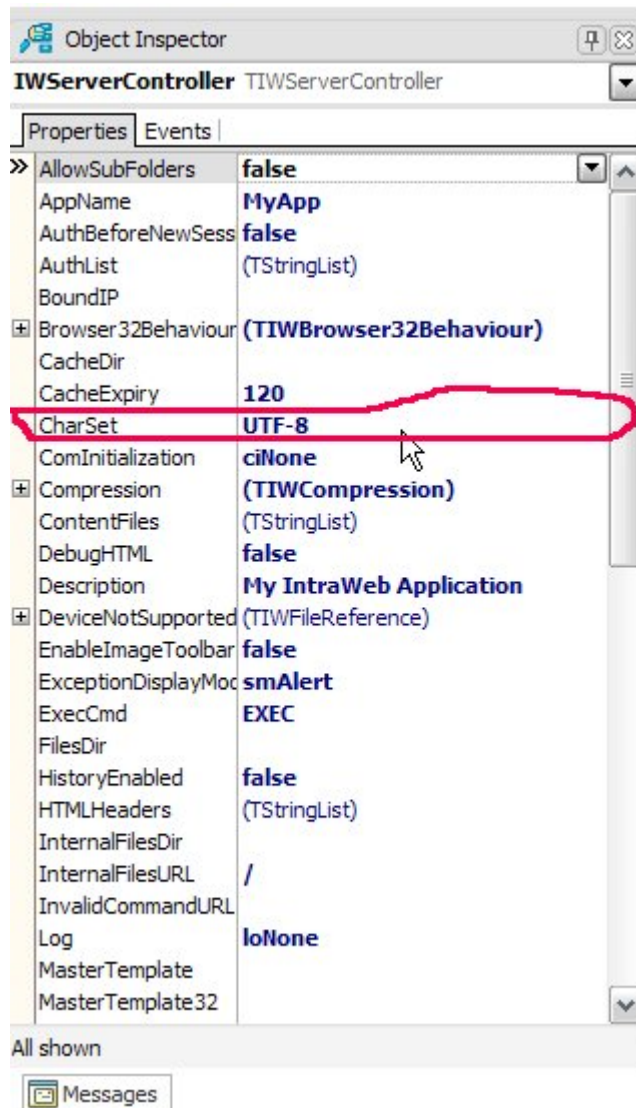


MainForm.pas 在 VCL For Web 中類似 VCL 應用程式中的主表單，開發人員可以藉由拖曳 VCL For Web 的元件來設計 **MainForm.pas**，就像在 VCL 應用程式中開發人員藉由拖曳 VCL 元件在表單中設計使用者介面一樣，只是當開發人員執行 VCL For Web 應用程式時，開發人員設計的 **MainForm.pas** 是執行在瀏覽器中的使用者介面，在 VCL For Web 應用程式中開發人員在 VCL For Web 表單設計使用者介面就是瀏覽器中會出現的使用者介面，VCL For Web 也是採用 WYSIWYW 和 RAD 的方式讓開發人員來開發 Web 應用程式，稍後我們即將在這個 VCL For Web 的表單中使用 VCL For Web 元件來設計 Web 圖形使用者介面。

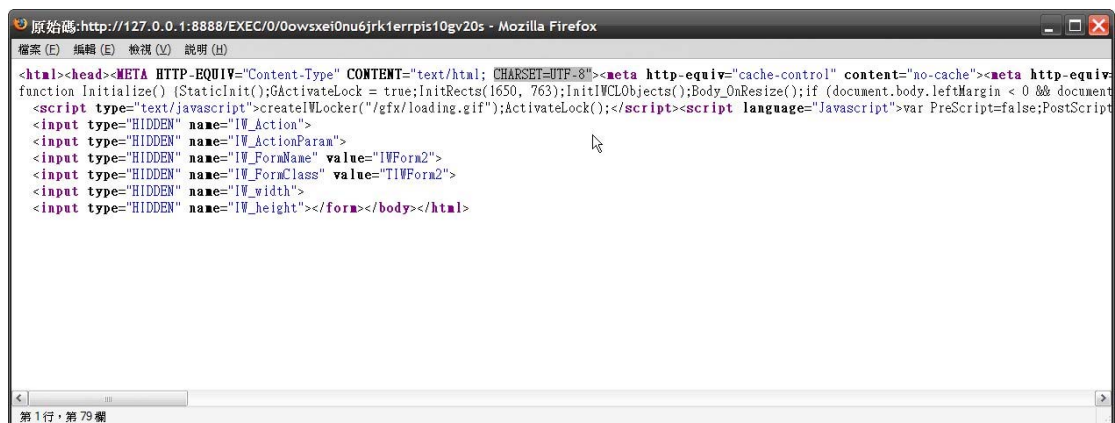
ServerController

在 VCL For Web 專案中的 **ServerController** 程式單元可以說是 VCL For Web 的控制中心，因為它控制了許多由 VCL For Web 框架開發出來的 Web 應用程式的執行特性，例如它可控制 VCL For Web 的應用程式是否使用 **Cookie**，**JavaScript**，使用的編碼機制，以及登錄驗證等重要功能，開發人員只需要在 **ServerController** 中使用物件檢視器來設定相關的特性即可輕易的完成這些重要的工作。

例如在本範例中讓我們在 **ServerController** 中設定使用 **UTF-8** 的編碼機制，在 Delphi 整合發展環境中開啓 **ServerController** 程式單元，在物件檢視器中設定它的 **CharSet** 為 **UTF-8** 即可：



一旦設定中 ServerController 的 CharSet 特性值之後，那麼當我們執行此 VCL For Web 應用程式，便可以在瀏覽器中檢查 Web 應用程式使用的編碼機制，從下圖中可以證明 Web 應用程式的確使用了 UTF-8 編碼機制。



在稍後討論 VCL For Web 的文章中會詳細討論 ServerController 的功能。

UserSessionUnit

VCL For Web 專案中的 UserSessionUnit 程式單元顧名思義可知它是提供對於 VCL For Web 應用程式每一個用戶端連結建立的物件，當使用者使用瀏覽器連結 VCL For Web 應用程式時，VCL For Web 框架便會為每一個使用者建立一個此物件。在實際的開發應用中，開發人員可以視 UserSessionUnit 為類似 C/S 應用程式中的資料模組物件，我們可以在 UserSessionUnit 中加入資料庫連結元件來連結資料庫，或是其他元件來提供每一個用戶端的使用者來使用。

因此在這個範例中，讓我們在 Delphi 整合發展環境中開啓 UserSessionUnit 程式單元，接著在其中放入 TSQLConnection 元件，設定它的特性值如下以連結前面文章實作的 JSON 伺服器：

特性	特性值
Driver	DataSnap
LoginPrompt	False

放入 TSqlServerMethod 元件，設定它的特性值如下

特性	特性值
SQLConnection	SQLConnection1
ServerMethodName	TDSServerModule1.GetSpeaker
Name	ssmSpeakers

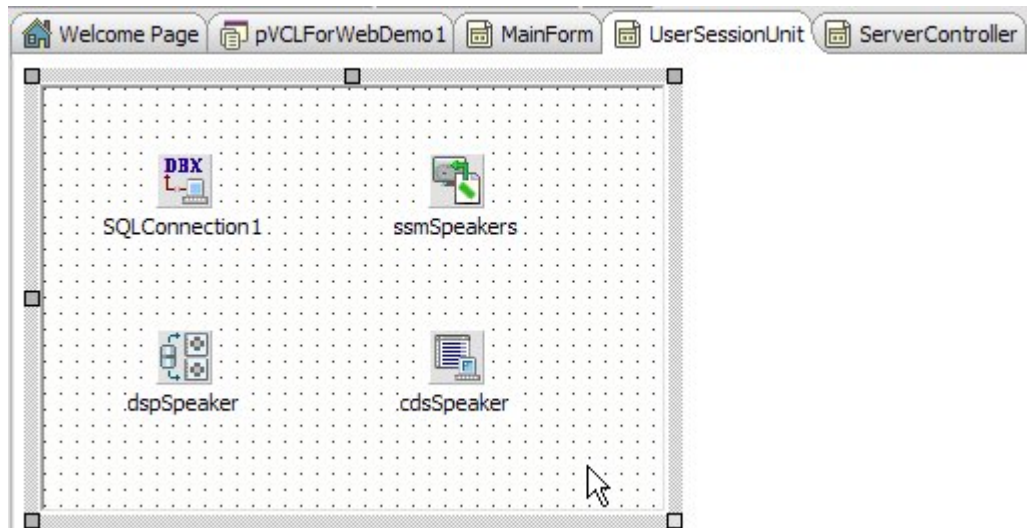
放入 TDataSetProvider 元件，設定它的特性值如下

特性	特性值
DataSet	ssmSpeakers
Name	dspSpeaker

放入 TClientDataSet 元件，設定它的特性值如下

特性	特性值
ProviderName	dspSpeaker
Name	cdsSpeaker

此時 UserSessionUnit 程式單元如下所示：



接著在 `UserSessionUnit` 程式單元的 `OnCreate` 事件處理函式中開啓 `SQLConnection1`:

```
procedure TIWUserSession.IWUserSessionBaseCreate(Sender: TObject);
begin
  Self.SQLConnection1.Connected := True;
end;
```

設計 Web 應用程式的圖形使用者介面

最後我們使用 `VCL For Web` 元件來設計 Web 圖形使用者介面，在整合發展環境中開啓 `MainForm.pas`，先點選 `File | Use Unit...` 功能表加入使用前面的 `ServerController` 和 `UserSessionUnit` 兩個程式單元:

```
implementation
uses UserSessionUnit, ServerController, Graphics;
```

接著放入 `TDataSource` 元件，設定它的特性值如下

特性	特性值
DataSet	IWUserSession.cdsSpeaker

放入 `TIWDBNavigator` 元件，設定它的特性值如下

特性	特性值
DataSource	DataSource1

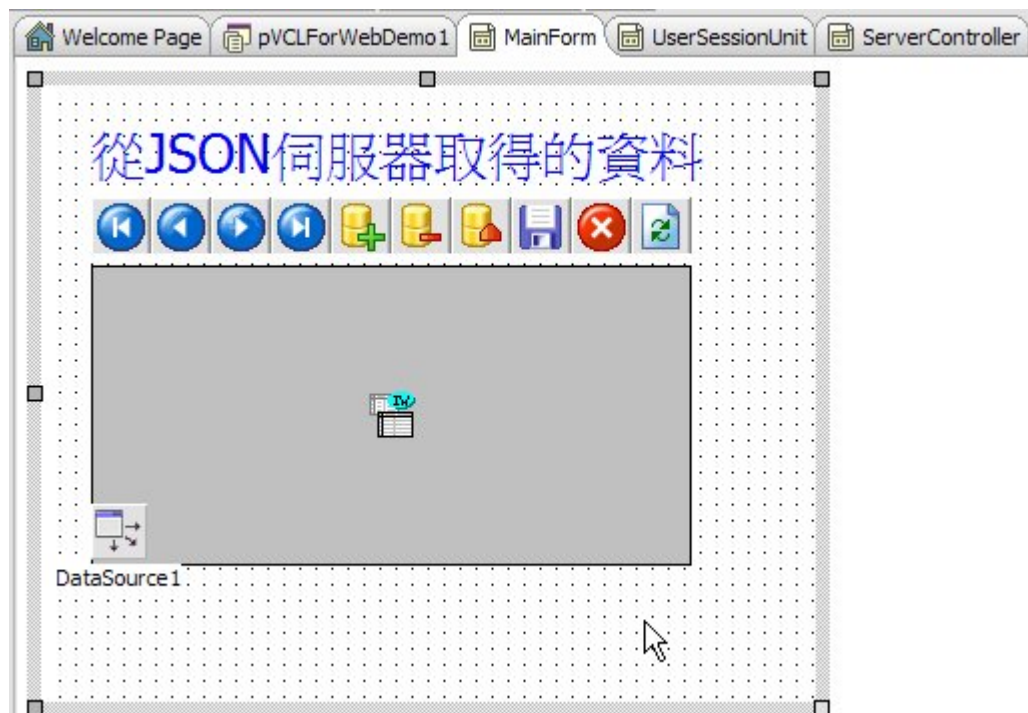
放入 `TIWDBGrid` 元件，設定它的特性值如下

特性	特性值
DataSource	DataSource1

放入 TIWLabel 元件，設定它的特性值如下

特性	特性值
Caption	從 JSON 伺服器取得的資料

此時 MaiForm 程式單元如下所示:



最後在 MainForm 程式單元的 OnRender 事件處理函式中撰寫如下的程式碼:

```
procedure TIWForm2.IWAppFormRender(Sender: TObject);
begin
  UserSession.cdsSpeaker.Active := True;
end;
```

VCL For Web 表單的 OnRender 事件會在它被瀏覽器存取時觸發，因此在這個事件處理函式中我們藉由開啓 UserSession 中的 TClientDataSet 從 JSON 伺服器中取得資料，但是上面的 UserSession 是什麼呢？

事實上 UserSession 即是宣告在 UserSessionUnit 程式單元中 TIWUserSession 物件，前面說過，VCL For Web 會為每一個用戶端建立一個 TIWUserSession 物件，這在 Web 應用程式中可以視為使用者公用資料的儲存地，因此開發人員可以在 TIWUserSession 類別中宣告任何需要使用的全域資

料，接著在 VCL For Web 應用程式中就可以藉由 `UserSession` 來取得 `TIWUserSession` 物件，再存取其中的全域資料即可。

而 `UserSession` 則是宣告在 `ServerController` 程式單元中的全域函式，在下面的 `ServerController` 程式單元原始程式中的 038~041 行可以看到 `UserSession` 函式回傳代表 VCL For Web 應用程式的 `WebApplication` 物件的 `Data` 特性值再轉變型態為 `TIWUserSession` 物件，而在 043~047 行我們可以看到 `TIWServerController` 類別會建立 `TIWUserSession` 物件並且儲存到 `WebApplication` 物件的 `Data` 特性之中。

```
001  unit ServerController;
002
003  interface
004
005  uses
006      SysUtils, Classes, IWServerControllerBase, IWBaseForm, HTTPApp,
007      // For OnNewSession Event
008      UserSessionUnit, IWApplication, IWAppForm;
009
010  type
011      TIWServerController = class(TIWServerControllerBase)
012          procedure IWServerControllerBaseNewSession(ASession:
013              TIWApplication;
014              var VMainForm: TIWBaseForm);
015      private
016
017      public
018      end;
019
020
021      function UserSession: TIWUserSession;
022      function IWServerController: TIWServerController;
023
024  implementation
025
026  {$R *.dfm}
027
028  uses
```

```

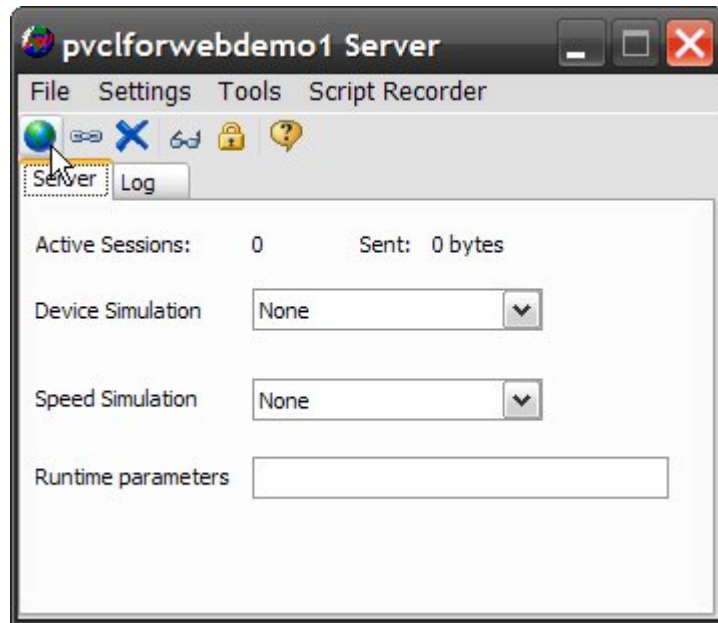
029     IWSInit, IWGlobal;
030
031     function IWServerController: TIWServerController;
032     begin
033         Result := TIWServerController(GServerController);
034     end;
035
036
037
038     function UserSession: TIWUserSession;
039     begin
040         Result := TIWUserSession(WebApplication.Data);
041     end;
042
043     procedure TIWServerController.IWServerControllerBaseNewSession(
044         ASession: TIWApplication; var VMainForm: TIWBaseForm);
045     begin
046         ASession.Data := TIWUserSession.Create(nil);
047     end;
048
049
050     initialization
051         TIWServerController.SetServerControllerClass;
052
053     end.

```

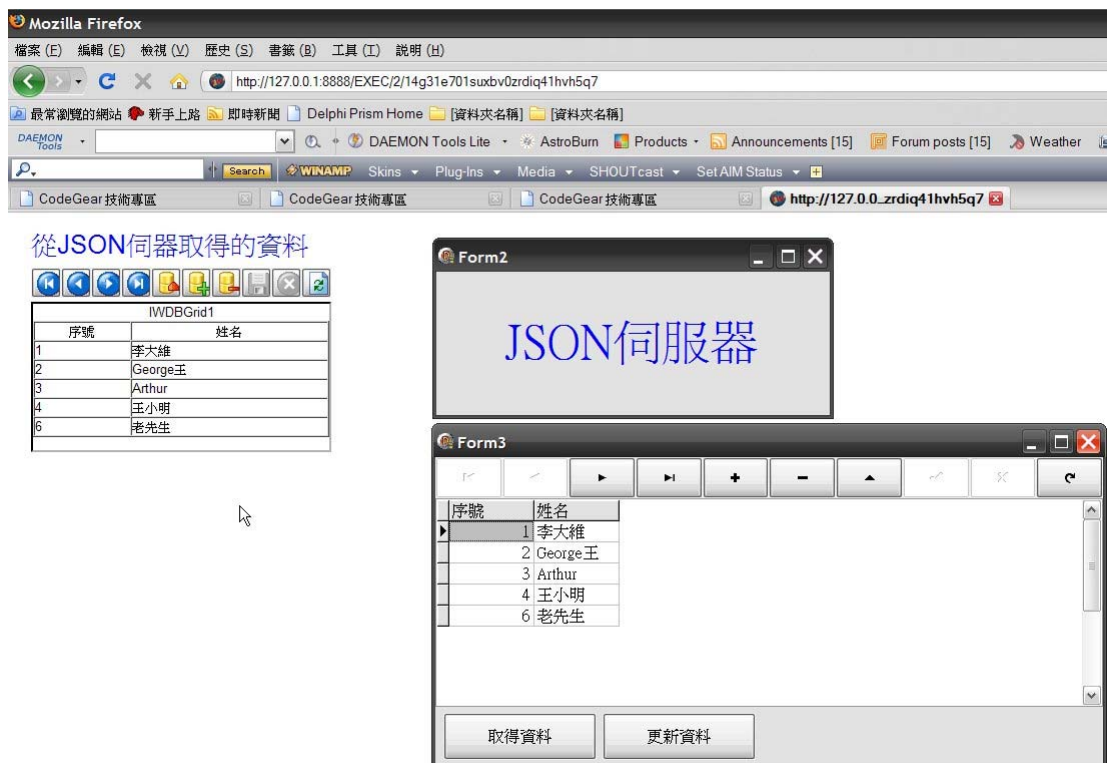
現在這個範例 **VCL For Web** 應用程式已經完成了，它使用 **Delphi 2009** 提供的 **DataSnap** 元件連結到 **JSON** 伺服器，又使用 **VCL For Web** 元件來建立 **Web** 圖形使用者介面來呈現 **JSON** 伺服器傳遞回來的資料。

現在執行這個範例 **VCL For Web** 應用程式，當然您必須先啟動前面文章中實作的 **JSON** 伺服器，那麼 **VCL For Web** 首先會啟動它的 **Launcher**，接著點選這個 **Launcher** 上方工具列中第一個按鈕，它就會啟動您的內定瀏覽器並且執行範例 **VCL For Web** 應用程式。

範例 **VCL For Web** 應用程式之所以會先啟動這個 **Launcher** 是因為我們在前面建立的 **VCL For Web** 應用程式型態是 **StandAlone** 的 **EXE** 的 **Web** 應用程式，如果我們是選擇建立 **Service** 或是 **ISAPI** 的型態，那麼就不需要這個 **Launcher** 了。



當您的瀏覽器啓動之後，它便會執行範例 VCL For Web 應用程式，您就可以在瀏覽器中看到類似如下的執行畫面了。



從上圖中我們就可以看到在瀏覽器中看到的資料和 JSON 用戶端應用程式看到的資料是一致的。

在結束本文之前讓我們再說明一下，VCL For Web 元件提供了豐富的功能可以讓開發人員客製化圖形使用者介面，就像使用 VCL 元件一樣，例如我們只需要回到 MainForm 表單，再設定 TIWDBGrid 的特性值如下

特性	特性值
RowCurrentColor	cIWebYELLOW
RowAlternateColor	cIWebMISTYROSE

那麼當我們點選瀏覽器中的 TIWDBNavigator 元件時，目前的資料就會以黃色來顯示了。



讀者可以在 VCL For Web 系列 1.zip 檔案中找到本文討論的範例 VCL For Web 應用程式。