

PowerBuilder Tips :

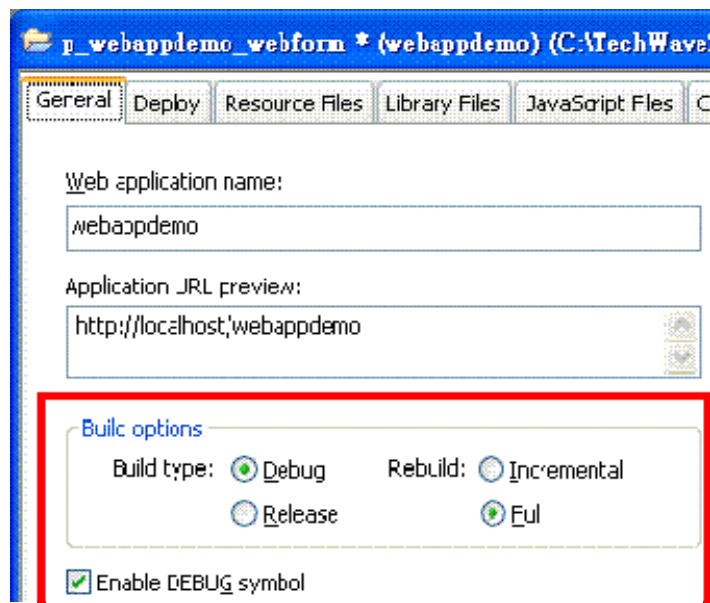
如何 Debug .NET WebForm 應用程式

使用 PB11 開發 .NET Web Form 除了開發流程簡易之外，PB 開發環境同時也提供 .NET Web 程式的偵錯(Debug)功能，.NET Web 偵錯使用的仍是 PB 傳統的偵錯畫面與偵錯功能，對於熟悉 PB 的程式設計師而言操作上絕不是什麼難事，只是這其中與 PB 傳統的偵錯方式還是有一些差異，如果沒有留意，恐怕反覆操作也難以找到程式開發中的錯誤，本期 Tips 將說明 .NET Web Form 偵錯的流程及需要注意的地方。

啓動 .NET Web Form Debug

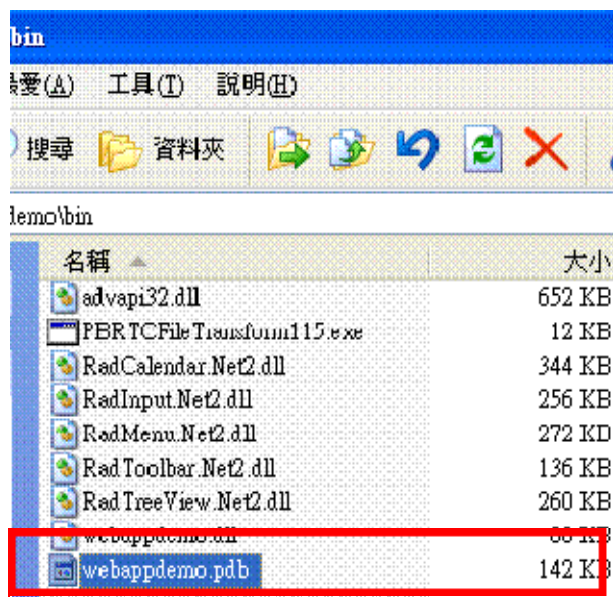
- **Debug 與 Enable Debug symbol 屬性**

啓動 .NET Web Form 偵錯功能的第一個動作，必須將 Project 中的 Build type 屬性設定為 "Debug" (如下圖一所示)，然後執行 Project 程式的佈署功能，這個步驟重要的原因是部署過程會產生一個副檔名為 PDB(Program Database file)的偵錯輔助檔案，少了這個檔案 .NET 應用相關的偵錯的功能便無法進行。



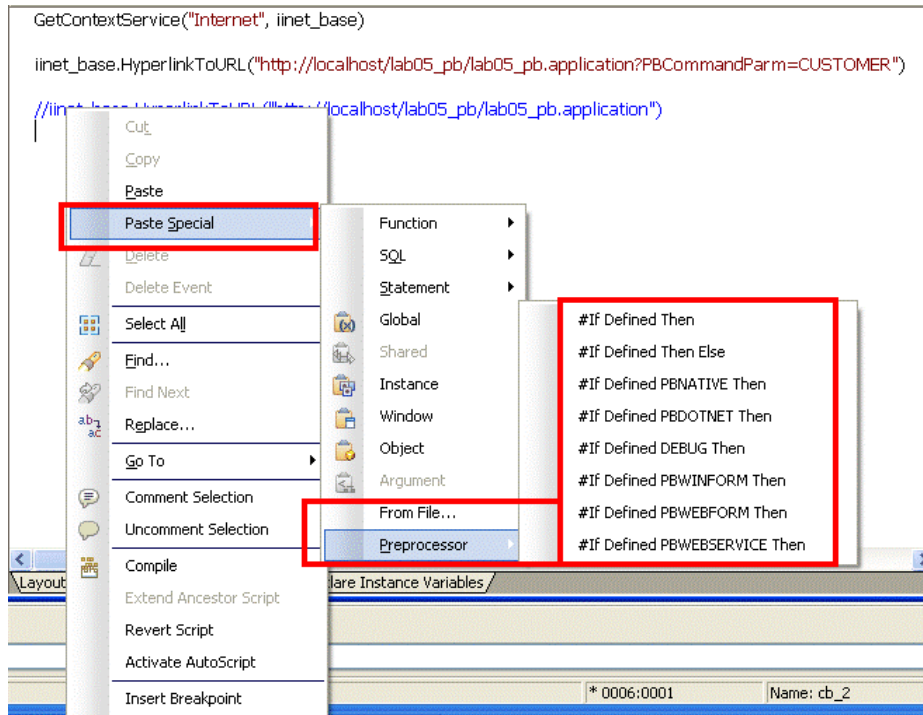
圖一、設定 .NET Web Form Project 的偵錯屬性，以便偵對 .NET Web 應用進行偵錯

.NET Web Form 的 PDB 檔會被部署到 IIS 目錄下，應用程式主目錄下的 bin 目錄中，如下圖二所示：



圖二、PB .NET Web Form 偵錯專用的.PDB 檔

另外，我們先前介紹過，在 PB11 中加入了一個新的語法，稱為條件式編譯區塊(Conditional Compiler Area)，這是一個以"#IF define <條件> THEN"和"#END IF"所區隔的一個特別空間(如下圖三所示)，用意是當程式設計師在 PB 開發工具中撰寫到非 PB 原生的語法時，如呼叫.NET 的 Assembly 中的函式，放在這個區塊中就能避免存檔時被 PB 原生編譯器誤認為錯誤語法而造成困擾。



圖三、在程式中插入 Debug Symbol 來輔助.NET Web 應用偵錯的進行

在圖一中如果勾選 **Enable Debug Symbol** 屬性，則在程式中撰寫如下圖四的程式碼，則程式運作時，這個區塊中的訊息便會被 **show** 出來，圖五則是當程式執行時，**Debug Conditional Compiler Area** 中程式碼所作用的結果畫面。

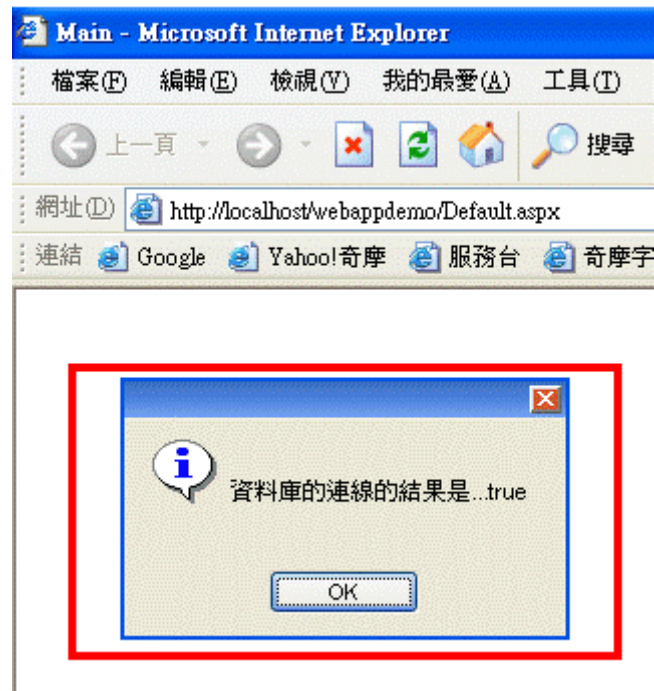
```

if sqlca.of_init_trans("ODBC", "EAS Demo DB V115", &
    "local", "dba", "sql") then
    sqlca.of_connect()
else
    close(this)
    return
end if

#if defined DEBUG then
    messagebox("", "資料庫的連線的結果是..." + string(sqlca.of_init_trans("ODBC", "EAS Demo DB V115", &
        "local", "dba", "sql")))
#endif

dw_contacts.settransobject(sqlca)
dw_contacts.retrieve()
    
```


圖四、在 conditional compiler area 中撰寫 Debug 相關的訊息



圖五、Enable Debug Symbol 執行的結果

Enable Debug Symbol 屬性最大的好處是程式設計師可以靈活使用它來搭配偵錯模式對 .NET Web 程式碼來進行偵錯，最主要的原因是 .NET 偵錯運作有些限制(筆者將在稍後做介紹)，這使得許多變數不見得能在程式設計師想看的時候出現，因此開啓這個模式，能夠讓整個偵錯更順利，且當 Enable Debug Symbol 屬性被取消，重新部署後，這些 Debug 條件式編譯區塊中的程式碼就自動失效，程式設計師無須一一刪除這些僅提供偵錯用的程式碼。

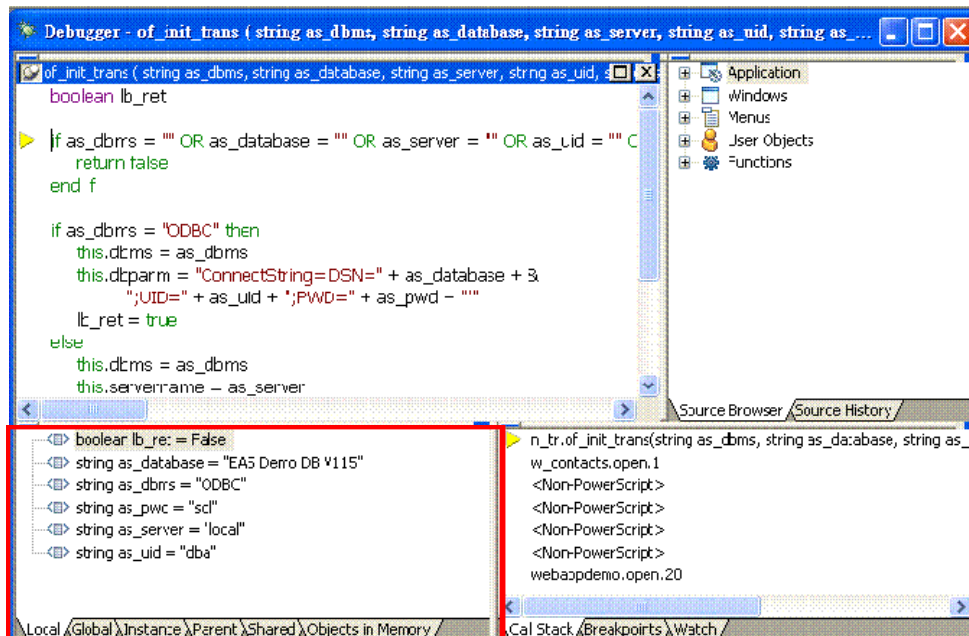
- 設定斷點，啟動偵錯模式開始偵錯：

接下來按下偵錯功能鈕() 啟動偵錯模式，並設定斷點，斷點設定上這裡有一點必須注意，.NET Web Form 的偵錯功能是所謂的 Single-stepping between events，簡單的說就是只有設定斷點的 Event 才會被偵錯，若沒有設定斷點的 event，PB .NET WebForm 的偵錯功能不會因“下一步”功能的作用而進入偵錯，這與 Native PB 是不一樣的。

舉個範例來說，程式設計師在 Application object 的 open event 最後一行，寫了一行程式：open(w_main)，用來開啓 w_main 視窗。若他要進行偵錯功能，在 Application object 的 open event 中某處下了斷點，並在稍後進行偵錯。基本上，偵錯功能執行到這個斷點，則會停止，進行偵錯，若持續偵錯執行到 open(w_main) 指令，然後點選下一步，若此程式是既有 Native PB 版本，則偵錯功能會進入 w_main 視窗的 Open Event，並且一步一步持續偵錯。若此程式是 PB .NET WebForm，則偵錯功能就會一直執行，直到下一個中斷點為止，所以假如沒有在 w_main

視窗的 Open Event 設定中斷點，則此 Open Event 就不會被偵錯到。

因此，程式設計師必須在每一個想要偵錯的 events 中下斷點，但 event 中呼叫到的 function 則無此限制。而當偵錯開始時，程式中的變數便會出現在偵錯 painter 下的子視窗中(如下圖六所示)。



圖六、NET Web Form Debug 視窗

其他限制與注意事項：

.NET Web Form 偵錯還有一些與傳統 PB 偵錯的差異如下：

1. 不支援 object in Memory 的顯示。
2. 被宣告的 Local 變數在未被執行到以前是不會被顯示的。
3. 當程式設計師關閉 Web 頁面後，偵錯模式仍然還是正在進行中，這與傳統 PB 偵錯的 Window 關閉會自動停止偵錯的特性不同。
4. 如果 IIS6 上最大的 worker processes 數目被設定大於 1 時，則無法進行 Web 應用的 debug，這是因為偵錯程式無法確認被偵錯中的 process 到底是新建立的還是由 worker processes pool 中被拿出來循環使用的。
5. 當有 PBLs 共用的狀況在.NET Web 應用上發生時，如果有兩個程式設計師先後對這個 Web 進行 debug，第二順位偵錯的程式設計師會被警告有另一個程式設計師已正在對該 Web 進行偵錯，且偵錯無法進行。
6. Web Form 的偵錯進行時無法支援某個被部署到遠端 IIS 上元件的偵錯。
7. 更詳細的內容請參照 Compiled HTML Help file 下的 Chapter 17 Compiling and Debugging 說明

結論

本次 PB Tips 介紹了 .NET Web Form 的偵錯如何進行及相關限制，其實 .NET Window Form 與 .NET Web Service 的偵錯也是大同小異，希望本文能帶給 PB 程式設計師在開發 .NET 相關應用時一點小小的幫助。