Displaying data from databases using different ASP.NET controls

Asist. Dragoş Sebastian CRISTEA
Catedra de Contabilitate şi Informatică Economică, Universitatea "Dunărea de Jos" Galaţi

This paper presents three Asp.Net controls used to render data stored inside a database. It's structure contains four parts: a brief introduction about ASP.NET followed by three segments each describing a certain type of control. Every theoretical aspect is further explained by a practical example written in C# as programming language. The back-end layer is represented by the "pubs" database which is bundled with every Sql Server database management system. The logic of the applications presented is not dependent on the data sources and we choose a sql server database and not an Oracle or DB2 one, because of the tight integration between products from the same vendor (Microsoft).
Keywords: *datasource, controls, dataset, templates.*

Introducere

ASP.NET reprezintă noua tehnologie de programare pentru Internet dezvoltată de compania Microsoft. Lucrul cu baze de date reprezintă o componentă fundamentală a oricărui astfel de mediu de programare. În acest sens noua tehnologie ASP se bazează integral pe noua platformă .NET, beneficiind astfel de toate avantajele care derivă din aceasta: un set nou de limbaje de programare care pot fi folosite, un model arhitectural simplificat şi în acelaşi timp eficientizat bazat pe noi clase şi obiecte, o nouă strategie de abordare a aplicaţiilor web care au ca nivel back-end o bază de date etc.

Acest articol nu se doreşte a fi o prezentare a tehnologiei ASP sau a platformei .NET. În schimb îşi propune să prezinte modurile în care informaţia dintr-o bază de date poate fi prezentată către utilizatorii finali. Vom exemplifica astfel trei controale folosite în afişarea informaţiei, respectiv controalele de tip lista, repeater şi dataList.

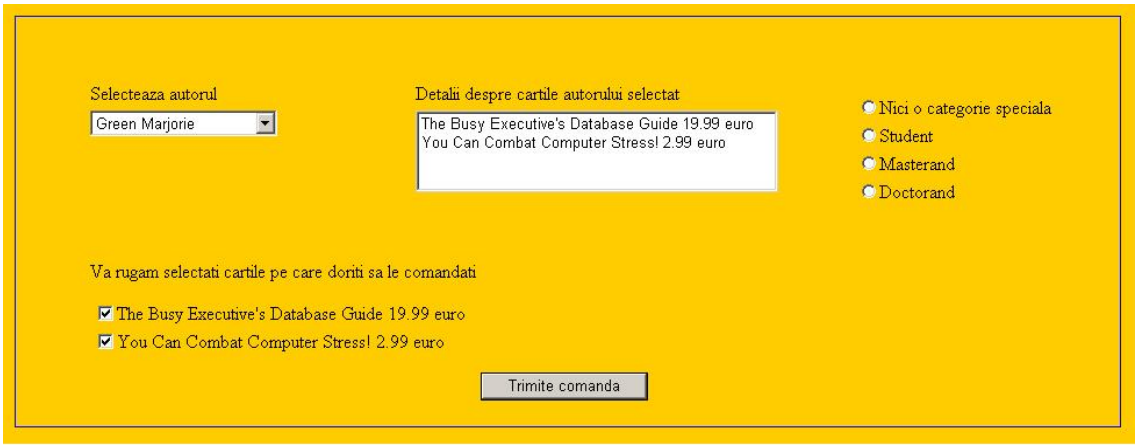
Majoritatea acestor controale moştenesc clasa ListControl. Acest lucru înseamnă că ele vor avea funcţionalităţi asemănătoare, urmând să se exemplifice diferenţele care apar în ceea ce priveşte eventualele proprietăţi suplimentare sau evenimente. Clasa List Control este o clasă de bază abstractă, nu poate fi instanţiată în mod direct. Ea defineşte proprietăţile, metodele, evenimentele specifice oricărui control de tip listă. Proprietăţile expuse de această clasă permit specificarea sursei de date(**datasource**), a numelui tabelei din care se extrag date (**datamember**). Toate obiectele afişate în controale de tip listă sunt stocate în colecţia numită Items. Ca şi eveniment principal menţionăm **SelectedIndex Changed()** care este declanşat la selectarea unui element din listă. Pentru folosirea acestor controale trebuie specificat "spaţiul de nume" care le conţine, repectiv: *System. Web.UI.WebControls*

Controalele de tip listă

Oricare dintre aceste controale poate fi folosit pentru a permite utilizatorilor să selecteze una sau mai multe valori dintr-o listă de posibile opţiuni. ASP.NET Framework conţine următoarele controale de tip listă: DropDownList, ListBox, RadioButtonList, CheckListBox. Toate aceste controale moştenesc clasa ListControl. Aceasta înseamnă că odată înţeles mecanismul funcţionării unuia dintre ele acesta va putea fi folosit şi asupra celorlalte, cu mici variaţii în ceea ce priveşte proprietăţile expuse.

Exemplul pe care-l prezentăm pentru această categorie de controale reprezintă un posibil formular de comandă pentru una sau mai multe cărţi. Nu sunt prezente elemente vizând identitatea cumpărătorului deoarece acestea au fost preluate odată cu autentificarea acestuia.

Rezultatul mini-aplicaţiei este prezentat în figura următoare:



Pentru realizarea acesteia s-a folosit baza de date denumită "**pubs**" pe care o regăsim în orice instalare de SQL Server ca exemplu tutorial. S-a ales drept mediu de lucru platforma Visual Studio.NET (Professional). Primul pas este reprezentat de crearea unui nou proiect de tip "**aplicație web**", al cărui nume nu are o importanță în logica exercițiului. Odată cu proiectul nou creat este generat un prim fişier cu extensia **aspx** care reprezintă de fapt pagina web dinamică pe care o vom completa. O idee considerată a fi utilă, dar nu obligatorie, o reprezintă redenumirea fişierului cu un nume mai sugestiv (**listcontrol.aspx**). Următorul pas îl reprezintă definirea componentelor care alcătuiesc pagina respectivă. Pentru aceasta folosind bara de instrumente şi tehnica **drag and drop** vom plasa pe formular controalele prezentate în figura de mai sus:

- 3Labels(Label1,Label2,Label3)
- DropDownList(DropDownList
- ListBox(ListBox1)
- 1RadioButtonList(RadioButtonList1)
- CheckBoxList(CheckBoxList1),
- Button(Button1)

In paranteză au fost notați identificatorii care vor fi folosiți în cadrul programului.

Următoarea etapă este reprezentată de realizarea legăturii cu baza de date şi a operațiilor efectuate asupra acesteia.

Este necesar un obiect de tip sqlConnection având următorul string de conectare (**proprietateaConnectionString):workstation,id=[nume_statie];packet,size=4096; user id=sa;datasource=".";persist,security info=False;initial catalog=pubs,** el având rolul să realizeze conexiunea efectivă cu baza de date. Instrucțiunile SQL vor fi transmise prin intermediul obiectelor de tip sqlDataAdapter definite astfel: sqlDataAdapter1 având pentru comanda de tip select(SelectCommand), proprietatea CommandText: *"SELECT au_lname + ' ' + au_fname AS Nume FROM authors"*(datele extrase astfel vor popula DropDownList1) respectiv sqlDataAdapter2 cu proprietatea CommandText: *"SELECT titles.title + ' ' + CONVERT (varchar(20), titles.price) + ' euro' AS InfoCarte FROM authors INNER JOIN titleauthor ON authors.au_id = titleauthor.au_id INNER JOIN titles ON titleauthor.title_id = titles.title_id WHERE (authors.au_lname + ' ' + authors.au_fname = @Param2)"*. Se observă folosirea unui parametru în cadrul instrucțiunii de interogare, care va fi preluat din DropDownList1, (adică câmpurile FirstName, respectiv LastName concatenate). Odată definite obiectele de tip DataAdapter, se vor putea genera cu ajutorul lor seturile de date(obiecte DataSet) pe care le denumim: dataSet11 respectiv dataSet21.

Salvarea datelor în baza de date este realizată prin intermediul obiectului sqlCommand1 care are proprietatea CommandText: *"INSERT INTO comanda (numeautor, infocarte, observatii) VALUES (@var1, @var2, @var3)"* (toate controalele de lucru cu baze de date: SqlConnection, SqlCommand, SqlDataAdapter se regăsesc în toolbox accesând rubrica Data).

Codul propriu zis al aplicației este inclus în contextul a 3 evenimente(încărcarea paginii, selectarea unui autor în DropDownList res-

pectiv apăsarea butonului de tip Submit), exprimate prin următoarele metode:

```
private void Page_Load(object sender, System.EventArgs e)
{
if(!Page.IsPostBack)
{
RadioButtonList1.Items.Add("Nici o categorie speciala");
RadioButtonList1.Items.Add("Student");
RadioButtonList1.Items.Add("Masterand");
RadioButtonList1.Items.Add("Doctorand");
sqlConnection1.Open();
sqlDataAdapter1.Fill(dataSet11);
DropDownList1.DataSource=dataSet11;
DropDownList1.DataTextField="Nume";
DropDownList1.BorderColor=System.Drawing.Color.Blue;
DropDownList1.DataBind
ListBox1.Items.Add("Nici un autor selectat");
sqlConnection1.Close();
}
}
private void Page_Load(object sender, System.EventArgs e)
{
if(!Page.IsPostBack)
{
RadioButtonList1.Items.Add("Nici o categorie speciala");
RadioButtonList1.Items.Add("Student");
RadioButtonList1.Items.Add("Masterand");
RadioButtonList1.Items.Add("Doctorand");
sqlConnection1.Open();
sqlDataAdapter1.Fill(dataSet11);
DropDownList1.DataSource=dataSet11;
DropDownList1.DataTextField="Nume";
DropDownList1.BorderColor=System.Drawing.Color.Blue;
DropDownList1.DataBind();
ListBox1.Items.Add("Nici un autor selectat");
sqlConnection1.Close();
}
}
private void Button1_Click(object sender, System.EventArgs e)
{
sqlConnection1.Open();
foreach (ListItem objListItem in CheckBoxList1.Items)
{
if (objListItem.Selected)
{
sqlCommand1.Parameters["@var1"].Value=DropDownList1.SelectedItem.Text;
sqlCommand1.Parameters["@var2"].Value=objListItem.Text;
sqlCommand1.Parameters["@var3"].Value=
RadioButtonList1.SelectedItem.Text;
sqlCommand1.ExecuteNonQuery();
}
}
sqlConnection1.Close();
}
```

Controlul Repeater

Este un control folosit în special pentru afişarea unui set de înregistrări extras de o interogare asupra unei baze de date. Folosirea lui implică însă definirea unui template care să precizeze modul în care vor fi afişate datele extrase. Exemplul prezentat are ca rezultat afişarea tuturor editurilor şi a cărţilor pe care acestea le-au publicat înregistrate în baza de date. Figura 2 prezintă rezultatul exerciţiului.

Pentru această aplicaţie se va folosi un control **Repeater,** pe care îl găsim în toolbox în zona componentelor de tip **webforms.** Se adaugă o astfel de componentă pe formularul *aspx* folosind la fel ca în aplicaţia precedentă tehnica drag and drop. Logica aplicaţiei este următoarea: Va fi generat un obiect de tip dataset pe baza a doua obiecte dataadapter(sqlDataAdapter1, sqlDataAdapter2) care extrag toate informaţiile despre o editura conform interogării SQL: *"SELECT pub_id, pub_name, city, state, country FROM publishers* **"** respectiv ale unui autor: "*SELECT title_id, title, type, pub_id, price, advance, royalty, ytd_sales, notes, pubdate FROM titles*". Pasul următor

îl reprezintă definirea unui obiect de tip **dataview** care va conține titlurile cărților filtrate dupa editură. Codul aplicației îl regăsim în evenimentul Page_Load şi este următorul:

```
private void Page_Load(object sender, System.EventArgs e)
{
sqlDataAdapter1.Fill(dataSet11.titles);
sqlDataAdapter2.Fill(dataSet11.publishers);
Repeater1.DataSource = dataSet11;
Repeater1.DataMember = "publishers";
Repeater1.DataBind();
}
public DataView ShowTitles( object Pub_ID )
{
string strFilter = "Pub_ID=" + Pub_ID;
DataView dvTitles = new DataView(
dataSet11.Tables[ "Titles" ], strFilter,"title", DataViewRowState.CurrentRows );
return dvTitles;
}
```

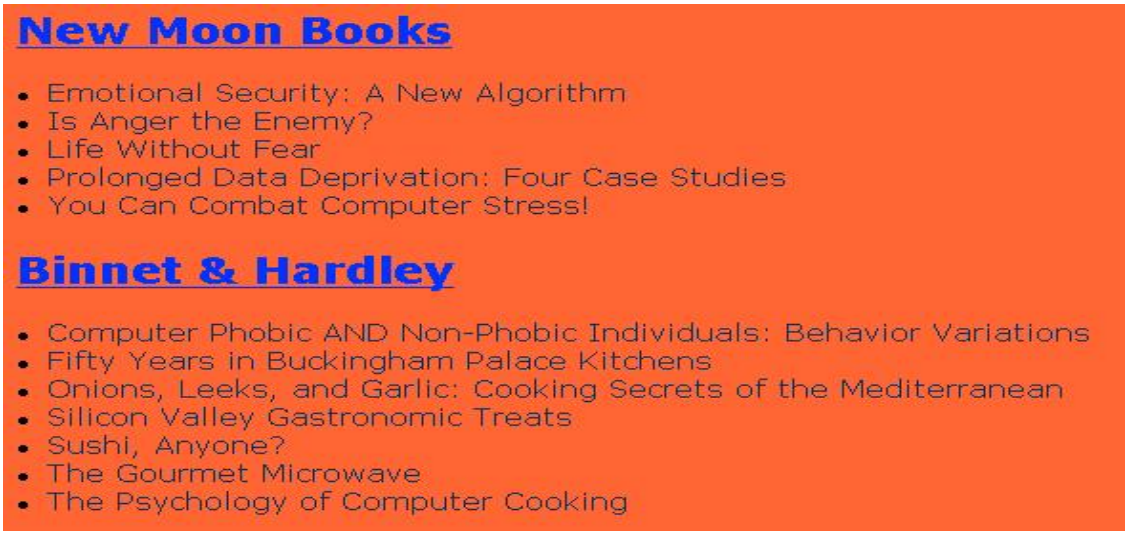


Fig. 2. Rezultatul exercițiului

Aşa cum se observă din codul de mai sus nu există nici o legătură între metoda **ShowTitles()** şi controalele existente pe formular. Aceasta va fi folosită în continuare în definirea şablonului pentru controlul repeater. Acesta va fi scris în codul html al paginii, între etichetele <asp:repeater> şi </asp:repeater>.

```
<ItemTemplate>
<h2><font face="verdana" color=#0033ff><u><%#
DataBinder.Eval( Container,
"DataItem.pub_name" )%></u></font></h2>
<asp:Repeater DataSource='<%# ShowTitles(
DataBinder.Eval( Container, "DataItem.pub_ID"
) )%>' Runat="server" ID="Repeater2">
<ItemTemplate>
<li>
<font face="verdana" color=#003366
size=3><%# DataBinder.Eval( Container,
"DataItem.title" )%></font>
</li>
</ItemTemplate>
</asp:Repeater>
</ItemTemplate>
```

Eticheta <ItemTemplate> defineşte începutul unui şablon care va fi aplicat pentru fiecare înregistrare din repeater. Ca element de noutate s-a dorit a fi subliniat faptul că în cadrul unui astfel de şablon poate fi definit un al doilea control de tip repeater care va afişa conținutul obiectului dataview creat în momentul încărcării paginii. Metoda **Eval()** permite filtrarea acelor atribute care le dorim a fi afişate, în cazul în care au fost extrase mai multe din baza de date.

Controlul DataList

Procedura de bază pentru "legarea" unui control DataList la o bază de date este în principiu aceeaşi ca şi în cazul unui Repeater. Este folosită proprietatea **DataSource** pentru a indica sursa de date respectiv metoda DataBind() pentru a-i copia conținutul în control.

Aplicația prezentată ca exemplu își propune folosirea controlului de tip DataList pentru realizarea unui meniu care să prezinte cărțile publicate la o editură pe care utilizatorul o va selecta. Figura următoare prezintă rezultatul, în condițiile în care a fost selectată editura "Binnet & Hardley".

Vom denumi fişierul (formularul) de lucru AfisareMenu.aspx. Primul pas se referă la afişarea obiectelor din meniu, respectiv numele editurilor din baza de date. Pentru aceasta sunt necesare următoarele obiecte configurate astfel: **sqlConnection**(definit exact ca în exemplele anterioare), **sqlData Adapter**(SelectCommand: "*SELECT pub_id, pub_name, city, state, country FROM publishers*"), DataSet(UnTyped). Odată aceste controale configurate va trebui definit un obiect care să permită afişarea datelor; vom folosi în acest caz un DataList. Pentru ca acesta să poată afişa datele cerute se vor configura şabloanele care se vor folosi. Aceasta se realizează folosind meniul de context pentru controlul DataList, selectând din acesta optiunea **EditTemplate.** În cadrul şablonului aferent unui element al listei (**Item Template**) se va adăuga un control Link Button care va avea proprietatea Data Bindings(**Custom Binding Expression**) definită astfel: "*DataBinder.Eval(Container, "DataItem.pub_name")*". De fapt în cadrul acestei expresii se precizează faptul că butonul va afişa ca text dintre toate atributele unei edituri doar numele acesteia.

În general cheia străină dintr-o tabelă este reprezentată de o cheie primară a unei alte tabele care a migrat pentru a realiza o legătură. Precizăm aceasta deoarece o metoda foarte eficientă de a realiza un astfel de formular în care dorim să se vizualizeze date cu legături semantice reprezentate la nivelul bazei de date prin diferite tipuri de legături, este de a prelua valoarea cheii primare a elementului selectat şi de a realiza o filtrare dupa valoarea ei asupra elementelor unui alt set de date. În cazul problemei noastre vom prelua valoarea cheii primare a elementului selectat din DataList şi o vom folosi pentru filtrarea datelor dintr-un al doilea DataSet care va conține titlurile cărților apărute la diferite edituri. De asemenea vom mai folosi valoarea cheii astfel preluată pentru a modifica dinamic valoarea unui Label pentru a afişa editura care a fost selectată(a se observa figura de mai sus).

Evenimentul OnLoad al paginii va conține codul necesar încărcării meniului cu numele editurilor:

```
private void Page_Load(object sender, System.EventArgs e)
{
if (!Page.IsPostBack)
{
sqlDataAdapter1.Fill(dataSet1);
DataList1.DataSource=dataSet1;
DataList1.DataKeyField = "pub_id"; //incarcarea colectiei DataKey asociata controlului
DataList1
DataList1.DataBind();
} }
```

Controlul DataList prezintă o colecţie specială numită **DataKeys**, care poate fi automat încărcată cu valori din coloană cheie a sursei de date.
Titlurile cărţilor vor fi afişate într-un control DataList2 care va conţine în şablonul de tip Item un label cu proprietatea DataBinding definită astfel: "DataBinder.Eval(Container, "DataItem.title")". Pentru că se doreşte afişarea cărţilor atunci când se acţionează asupra butonului de tip link vom defini evenimentul ItemCommand al controlului DataList care va transmite ca parametru către Data Adapter2 valoarea cheii primare pentru editura selectată, acest lucru generând afişarea doar a acelor titluri publicate la editura respectivă. Codul asociat evenimentului Item Command este următorul:

```
private void DataList1_ItemCommand(object source,
System.Web.UI.WebControls.DataListCommandEventArgs e)
{
//identificarea elementului selectat din DataList1
DataList1.SelectedIndex = e.Item.ItemIndex;
//deschiderea conexiunii cu baza de date
sqlConnection1.Open();
//transmiterea catre parametrul cu numele @param2 din sqlDataAdapter2(care
contine titlurile din baza de date) a valoarii cheii primare a editurii selec-
tate
sqlDataAdapter2.SelectCommand.Parameters["@Param2"].Value=
(string)DataList1.DataKeys[ e.Item.ItemIndex ];
sqlDataAdapter3.SelectCommand.Parameters["@Param2"].Value=
(string)DataList1.DataKeys[ e.Item.ItemIndex ];
sqlDataAdapter2.Fill(dataSet31);
sqlDataAdapter3.Fill(dataSet51.publishers);
DataList2.DataSource=dataSet31;
DataList2.DataBind();
Label3.DataBind();
sqlConnection1.Close();
}
```

Concluzii

Mediul ASP.NET oferă dezvoltatorilor de aplicaţii web un set complet de controale pentru afişarea informaţiei. Complexitatea acestora derivă atât din numărul impresionant de atribute pe care le posedă cât şi din modul în care sunt integrate în platforma .NET. Din punct de vedere al dificultăţii în utilizare situaţia am putea spune că este relativă, în sensul că variază de la un nivel care nu implică un volum mare de cod(acesta fiind generat automat de IDE) până la configurarea totală a acestora de către programator, în cazul aplicaţiilor complexe.
Aspectul cel mai important al mediului ASP.NET îl reprezintă integrarea conceptelor şi tehnicilor folosite într-un model de programare. Astfel realizarea unei aplicaţii web, bazată pe formulare web implică folosirea aceloraşi concepte valabile şi în cazul programelor windows standard; în cazul de faţă putem spune că diferenţele între modul de folosire a controalelor de afişare a datelor folosite în exemplele web de mai sus nu diferă de modul în care acestea ar fi fost folosite în cazul în care dezvoltam o aplicaţie standard.

Bibliografie

1. James Foxall, Wendy Haro-Chun, Sams Publishing March 2002 – Teach yourself C # in 24 hours
2. Stephen Walter, Sams Publishing December 2002, ISBN 0-672-32476-8, ASP.NET Kick Start
3. Joe Martin, Brett Tomson, Teora Press – Introducere in Asp.NET
4. Mickey Williams, Microsoft Press, 2002 – Microsoft Visual C#.NET
5. http:// msdn.microsoft.com