**التعامل مع البيانات على الإنترنت**

**Accessing Data on the Web**

يمثّل الإنترنت شبكة تجاريّة ضخمة، تسمح للمنتجين بالإعلان عن منتجاتهم وخدماتهم.. هذا هو ما يجعل معظم تطبيقات الإنترنت تتعامل مع قواعد البيانات، التي تحتوي على السلع والأصناف والأسعار والخدمات.... إلخ.

وفي هذا الفصل، سنتعلّم كيف نبني تطبيق إنترنت يتعامل مع قواعد البيانات.

* [**ربط أدوات الإنترنت بالبيانات Data-Bound Web Controls:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\2-%20برمجة%20قواعد%20البيانات%20على%20النت\1-%20ربط%20أدوات%20الإنترنت%20بالبيانات.htm)
* [**ربط الأدوات بمصفوفة قائمة ArrayList:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\2-%20برمجة%20قواعد%20البيانات%20على%20النت\2-%20ربط%20الأدوات%20بمصفوفة%20قائمة.htm)
* [**الارتباط بمجموعات البيانات Binding to DataSets:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\2-%20برمجة%20قواعد%20البيانات%20على%20النت\3-%20الارتباط%20بمجموعات%20البيانات.htm)
* [**مشروع AuthorsBooks:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\2-%20برمجة%20قواعد%20البيانات%20على%20النت\4-%20مشروع%20AuthorsBooks.htm)
* [**شراء الكتب على الإنترنت:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\2-%20برمجة%20قواعد%20البيانات%20على%20النت\5-%20شراء%20الكتب%20على%20الإنترنت.htm)
* [**عرض الجدول الأساسي والجداول الفرعيّة Master-Detail:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\2-%20برمجة%20قواعد%20البيانات%20على%20النت\6-%20عرض%20الجدول%20الأساسي%20والجداول%20الفرعيّة.htm)

**ربط أدوات الإنترنت بالبيانات Data-Bound Web Controls:**

مثل أدوات الويندوز، يمكن ربط أيّ أداة من أدوات الإنترنت بمصدر للبيانات Data Source.. وبخلاف مربّع النصّ TextBox ومربّع الاختيار CheckBox، فإنّ الأدوات التالية تمثّل أهمّ أدوات عرض البيانات على الإنترنت، وذلك لقدرتها على عرض أكثر من عنصر:

|  |  |
| --- | --- |
| ListBox | القائمة. |
| DropDownList | القائمةالمنسدلة. |
| CheckBoxList | قائمة مربعات الاختيار.. وهي تعرض مجموعة من مربعات الاختيار معا كأداة واحدة، كنوع من تسهيل التصميم وكتابة الكود. |
| RadioButtonList | قائمة أزرار التحويل.. وهي تعرض مجموعة من أزرار التحويل معا كأداة واحدة. |
| DataGrid | جدول البيانات.. وفيه يتمّ عرض العناصر في أعمدة وصفوف. |
| DataList | قائمة البيانات.. وهي تعرض مجموعة من العناصر، كلّ عنصر في خانة مستقلّة. |
| Repeater | أداة العرض المتكرّر.. وهي تقوم بعرض العناصر في مجموعة متكرّرة من التراكيب، على حسب تصميمك أنت (يمكن أن يتكوّن كلّ عنصر من مجموعة مربعات نصّ ولافتات.. وبمجرّد تصميمك لعنصر واحد، سيتمّ تكرار هذا عرض التصميم لباقي العناصر.. سيتضح لك هذا أكثر عند تصميم قائمة البيانات DataList، فالخطوات متشابهة). |

ولربط أداة الإنترنت بمجموعة بيانات DataSet، يجب أن تجعل خاصيّة DataSource للأداة تشير إلى اسم مجموعة البيانات.. بعد ذلك عليك باستدعاء الوسيلة "ربط البيانات" DataBind الخاصّة بالأداة في حدث تحميل الصفحة Load.

ويمكن استدعاء الوسيلة DataBind للصفحة كلّها (Me.DataBind)، وذلك لملء كلّ الأدوات بالبيانات دفعة واحدة.

وتمتلك القائمة المنسدلة DropDownList ـ وما شابهها من باقي أنواع القوائم ـ خاصيتين إضافيّتين، هما:

**حقل بيانات النصّ DataTextField:**

وتمثّل اسم العمود الذي سيتمّ ملء القائمة به.

**حقل بيانات القيمة DataValueField:**

وغالبا ما يتمّ ربط هذه الخاصيّة بالمفتاح الأساسيّ Key Field للجدول.. وعندما يختار المستخدم عنصرا من القائمة المنسدلة، نستقبل نحن قيمة خاصيّة DataValueField المناظرة لهذا العنصر، وبذلك نحدد الصف المطلوب في الجدول.

وأعتقد أنّك تعرف بالفعل كيفيّة إنشاء مجموعة البيانات.. وإذا لم تكن تعرف، فعليك بمراجعة القسم الخاص [ببرمجة قواعد البيانات](file:///C:\\Users\\Administrator\\Desktop\\vb.net\\04-%20Databases%20قواعد%20البيانات\\17-%20تطبيقات%20قواعد%20البيانات.htm) في هذا الكتاب.. وكن مطمئنّا: لا يوجد اختلاف يذكر بين استخدام قواعد البيانات مع نماذج الويندوز واستخدامها مع نماذج الإنترنت.

**ربط الأدوات بمصفوفة قائمة ArrayList:**

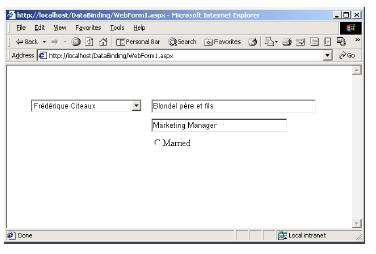
في هذا المثال لن نربط الأدوات بقاعدة بيانات!

إذن فبم سنربطها؟

لقد ذكرنا [من قبلُ](file:///C:\\Users\\Administrator\\Desktop\\vb.net\\04-%20Databases%20قواعد%20البيانات\\17-%20تطبيقات%20قواعد%20البيانات\\6-%20ربط%20البيانات.htm) أنّ بإمكاننا ربط الأداة بمصفوفة أو بمصفوفة قائمة ArrayList، وليس بسجلّ Record فحسب.

تعال نرى ذلك، من خلال تطبيق DataBinding.

صمّم الواجهة لتبدو كما في الصورة التالية:



وكما هو واضح، سيتمّ اختيار اسم الشخص من القائمة المنسدلة، حيث سيتمّ عرض بياناته في مربّعي النصّ وزر التحويل Radio Button.

والآن، علينا أن نملأ المصفوفة القائمة ArrayList بالبيانات اللازمة، حيث سيكون كلّ عنصر من عناصرها نسخة من الخليّة Contact.. تعال نعرّف هذه الخليّة أولا:

**Class Contact**

**Dim \_ID As Integer**

**Dim \_name As String**

**Dim \_title As String**

**Dim \_company As String**

**Dim \_married As Boolean**

**Property ID() As Integer**

**Get**

**ID = \_ID**

**End Get**

**Set(ByVal Value As Integer)**

**\_ID = Value**

**End Set**

**End Property**

**Property Name() As String**

**Get**

**Name = \_name**

**End Get**

**Set(ByVal Value As String)**

**\_name = Value**

**End Set**

**End Property**

**Property Title() As String**

**Get**

**Title = \_title**

**End Get**

**Set(ByVal Value As String)**

**\_title = Value**

**End Set**

**End Property**

**Property Company() As String**

**Get**

**Company = \_company**

**End Get**

**Set(ByVal Value As String)**

**\_company = Value**

**End Set**

**End Property**

**Property Married() As Boolean**

**Get**

**Married = \_married**

**End Get**

**Set(ByVal Value As Boolean)**

**\_married = Value**

**End Set**

**End Property**

**Public Overrides Function ToString() As String**

**Return (\_name)**

**End Function**

**End Class**

بعد ذلك سنعرّف المصفوفة القائمة، ونملأها بنسخ من الخليّة التي أنشأناها:

**Dim Contacts As New ArrayList()**

**Dim c As New Contact()**

**c.ID = 0**

**c.Name = "Maria Anders"**

**c.Company = "Alfreds Futterkiste"**

**c.Title = "Sales Representative"**

**c.Married = False**

**Contacts.Add(c)**

واضح أنّنا اكتفينا ببيانات شخص واحد للإيضاح، ولكنّ الكود الموجود في التطبيق يضيف العديد من الأشخاص.

والآن، سنربط القائمة المنسدلة بالمصفوفة القائمة:

**If Not Me.IsPostBack Then**

**DDListName.DataSource = Contacts**

**DDListName.DataTextField = "Name"**

**DDListName.DataValueField = "ID"**

**DDListName.DataBind()**

**End If**

لاحظ استخدامنا للشرط، لنتأكّد من أنّ هذه العمليّة ستتمّ مرّة واحدة فقط، وليس في كلّ مرّة يقوم فيها الخادم بتحديث الصفحة.

أعرف أنّك ستسألني في استغراب:

- ولكن لماذا ملأنا المصفوفة القائمة خارج جملة الشرط؟.. ألا يعني هذا أنّ المصفوفة سيتمّ ملؤها من جديد في كلّ مرّة يتمّ فيها تحديث الصفحة؟

- بلي يعني هذا.

- إذن فهو كود عقيم، ويستهلك الكثير من الوقت.

- هذا صحيح.. بل وسيتمّ تحديد أوّل عنصر في القائمة في كلّ مرّة يتمّ فيها إعادة ملئها!

- عجبا.. فلم إذن نفعل هذا رغم عيوبه؟

- لأنّ هذا هو الإنترنت يا صديقي.. إنّ تطبيقك يعمل على الخادم، حيث يمكن أن يطالب مئات المستخدمون بعرض صفحتك في نفس اللحظة.. لهذا فمن الخطإ أن يتمّ الاحتفاظ بكمّ ضخم من البيانات بين الاستدعاءات المختلفة Between Sessions، وإلا اختنق خادمك وساء أداؤه بسبب استهلاك ذاكرته.. لهذا يجب أن تصمم صفحتك، بحيث ينفض الخادم يده منها بعد تحميلها، دون أن يضطرّ للاحتفاظ بالكثير من المعلومات عنها.

- معقول فعلا.

- هذا هو نفس ما سنفعله مع مجموعة البيانات DataSet فيما بعد، حيث سنحضر البيانات من قاعدة البيانات، ونملأ بها الصفحة، ثمّ نغلق الاتصال ونتخلّص من مجموعة البيانات.

والآن عليك أن تجعل خاصيّة "إرجاع تلقائيّ" AutoPostBack للقائمة المنسدلة True، وذلك حتّى تقوم بإطلاق الحدث SelectedIndexChanged لإخبار الخادم كلّما تغيّر العنصر المحدّد فيها.

تعال نكتب كود هذا الحدث، حيث سنقوم بعرض بيانات الشخص في باقي الأدوات:

**Private Sub DDListName\_SelectedIndexChanged(sender As Object, e \_**

**As System.EventArgs) \_**

**Handles DDListName.SelectedIndexChanged**

**Dim selIndex As Integer = DDListName.SelectedIndex**

**txtCompany.Text = CType(Contacts(selIndex), Contact).Company**

**txtTitle.Text = CType(Contacts(selIndex), Contact).Title**

**radioMarried.Checked = \_**

**CType(Contacts(selIndex), Contact).Married**

**End Sub**

لاحظ أنّ مربعي النصّ وزرّ التحويل ليست مرتبطة بالبيانات، ولكنّنا نغيّر قيمها بأنفسنا من خلال الكود.

ولفهم أفضل لكيفيّة معالجة ASP.NET لعمليّة ربط البيانات، انظر للكود الذي تمّ إرساله للعميل.. اضغط F5 لتشغيل التطبيق، وعندما يظهر لك متصفّح الإنترنت Internet Explorer افتح القائمة الرئيسيّة "عرض" View، واضغط الأمر Source.. وفي صفحة كود HTML التي ستظهر، ابحث عن عنصر التنسيق SELECT.. ستجده كالتالي:

<SELECT NAME="DDListName" ID="DDListName" ONCHANGE="\_\_doPostBack(‘DDListName’,’’)"

LANGUAGE="javascript" STYLE="height: 22px; width: 196px; z-index: 101; left: 42px;

position: absolute; top: 60px">

<OPTION VALUE="0">Maria Anders</OPTION>

<OPTION VALUE="1">Ana Trujillo</OPTION>

<OPTION VALUE="2">Thomas Hardy</OPTION>

<OPTION VALUE="3">Fr&#233;d&#233;rique Citeaux</OPTION>

<OPTION VALUE="4">Elizabeth Brown</OPTION>...

وكما ترى: هذا الكود يتمّ تنفيذه على جهاز العميل، ليقوم بإنشاء قائمة منسدلة، ويملؤها بالبيانات.

هل لاحظت فرقا آخر بين هذا التطبيق ومناظره في تطبيقات الويندوز؟

ألا تشعر أنّ الكود الذي كتبناه في حدث تغيّر عنصر القائمة طويل بعض الشيء؟

لو كنّا نكتب تطبيق ويندوز، كنّا سنستخدم خاصيّة SelectedItem الخاصة بالقائمة المنسدلة، حيث تعيد لنا كائنا Object يحتوي على كلّ بيانات العنصر.. وكنّا سنستخدم جملة تحويل كالتالية للتعامل مع خصائص هذا الكائن:

**txtCompany.Text = CType(DDListName.SelectedItem, \_**

**Contact).Company**

إذن فلماذا لم نفعل ذلك هنا؟

آهٍ منك.. دائما تنسى أنّنا نتعامل مع صفحة ويب، تعتمد في الأساس على كود HTML، وهو غير مصمم للتعامل مع الكائنات.. فقط مع القيم.

**الارتباط بمجموعات البيانات Binding to DataSets:**

سنقوم الآن بإنشاء تطبيق WebBooks.

صمّم صفحة كما بالشكل:



حيث سنقوم بملء قائمة منسدلة DropDownList بأسماء المؤلّفين، وعند اختيار مؤلّف منها يظهر رقمه في اللافتة الأولى، وتظهر كتبه في قائمة أخرى.. وعند اختيار أيّ كتاب منها، يظهر رقمه في اللافتة الثانية.

[أنشئ موصل بيانات](file:///C:\Users\Administrator\Desktop\vb.net\04-%20Databases%20قواعد%20البيانات\17-%20تطبيقات%20قواعد%20البيانات\32-%20موصل%20البيانات.htm) DataAdapter من جدول المؤلفين في قاعدة بيانات الكتب التي أنشأناها سابقا بـ Access (بسحبه من متصفح الخوادم وإسقاطه على النموذج) وغيّر اسمه إلى DAAuthors، [وأنشئ مجموعة البيانات](file:///C:\\Users\\Administrator\\Desktop\\vb.net\\04-%20Databases%20قواعد%20البيانات\\17-%20تطبيقات%20قواعد%20البيانات\\4-%20إنشاء%20مجموعة%20البيانات.htm) DSAuthrs.

اتبع نفس الطريقة التي تعلّمناها في فصول قواعد البيانات.. لن أكرّر نفسي ها هنا!

ضع خصائص القائمة المنسدلة Authors كما يلي:

|  |  |
| --- | --- |
| الخاصية | القيمة |
| DataSource | DsAuthors1 |
| DataMember | Authors |
| DataTextField | Author |
| DataValueField | ID |

ثمّ أدخل الجمل التالية في حدث تحميل الصفحة:

**If Not Me.IsPostBack Then**

**DAAuthors.Fill(DsAuthors1, "Authors")**

**Authors.DataBind()**

**End If**

جرّب البرنامج الآن.. ستجد أنّ القائمة المنسدلة الأولى تعرض أسماء المؤلفين..

جميل..

علينا الآن أن نربط اللافتة بالمؤلف المحدّد في القائمة..

هنا لن يشبه الأمر ما فعلناه في تطبيقات الويندوز.. بل علينا أن ننشئ مجموعة بيانات أخرى، مبنية على جملة استعلام تعيد من قاعدة البيانات الصفّ الذي يمثّل المؤلف المحدّد في القائمة.

طيب.. أذكّرك بالخطوات في عجالة:

من صندوق الأدوات، اضغط الشريط الخاص بأدوات قواعد البيانات Data.

ضع على النموذج نسخة من الأداة OldbDataApapter..

سيظهر لك المعالج السحريّ.. امش مع [خطواته المعتادة](file:///C:\\Users\\Administrator\\Desktop\\vb.net\\04-%20Databases%20قواعد%20البيانات\\17-%20تطبيقات%20قواعد%20البيانات\\33-%20المعالج%20السحري%20لإعداد%20موصّل%20البيانات.htm)، واستخدم باني الاستعلام Query Builder لبناء جملة الاستعلام التالية:

**SELECT ID**

**FROM Authors**

**WHERE (ID = ?)**

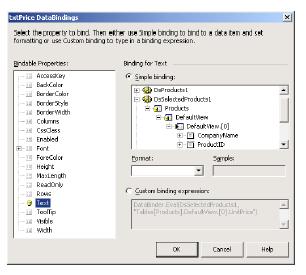
وأكمل خطوات المعالج.

سمّ هذا الموصّل DASelectedAuthor.

الآن يمكنك بناء مجموعة بيانات من هذه الموصل.. سمّها DSSelectedAuthor.

ثمّ حدّد اللافتة الأولى، وفي نافذة الخصائص اختر خاصية DataBindings (ستجدها أوّل خاصية) واضغط زرّ الانتقال في خانة القيمة المناظرة لها..

سيظهر لك مربّع الحوار التالي:



ستجد على اليمين شجرة تمثّل أسماء مجموعات البيانات المتاحة..

اختر المجموعة DSSelectedAuthor1 واعرض عناصرها.. أسدل العنصر Author، ومنه أسدل العنصر DefaultView، ومنه أسدل العنصر DefaultView.[0]..

وأخيرا اختر العنصر ID، واضغط OK.

**ملحوظة:**

لعلك تتساءل: كيف يمكنك القيام بعملية الربط هذه من الكود؟

لعلك لاحظت وجود تعبير طويل مكتوب في مربّع غير مفعّل يحمل العنوان Custom Binding.. هذا هو التعبير الذي سنستخدمه في كتابة الكود:

**LbAuthors.Text = DataBinder.Eval(DsSelectedAuthor1, \_**

**"Tables[Authors].DefaultView.[0].ID")**

إنّ الكائن DataBinder مكافئ للكائن DataBinding الذي استخدمناه في تطبيقات الويندوز.. والوسيلة Eval الخاصة به، تعيد قيمة من أحد حقول الجدول.

الآن علينا أن نُمدّ موصل البيانات بقيمة المعامل المجهول (اسم المؤلف).. هذا ما سنفعله في الحدث SelectedIndexChanged الخاص بالقائمة المنسدلة.. ولفعل هذا يجب أولا أن تغيّر خاصية AutoPostBack الخاصة بالقائمة إلى القيمة True.

كلّ ما سنفعله بعد ذلك، هو أن نستخدم العنصر المحدّد في القائمة، كمعامل لجملة التحديد في موصل البيانات، قبل أن نملأ مجموعة البيانات، ثم ننفذ ارتباط اللافتة بها:

**Private Sub Authors\_SelectedIndexChanged(ByVal sender As \_**

**System.Object, ByVal e As System.EventArgs) \_**

**Handles Authors.SelectedIndexChanged**

**DASelectedAuthor.SelectCommand.Parameters(0).Value = \_**

**Authors.SelectedItem.Value**

**DASelectedAuthor.Fill(DsAuthor1, "Authors")**

**LbAuthors.DataBind()**

**End Sub**

الآن لو جرّبت البرنامج، فستجد أنّ رقم كلّ مؤلف تختاره يعرض في اللافتة..

لكنّ عيبا ستلاحظه، هو أنّ اسم أوّل مؤلف يتمّ عرضه عند تحميل الصفحة، لكن دون عرض رقمه!

بسيطة.. استدع الحدث SelectedIndexChanged عند تحميل النموذج:

**If Not Me.IsPostBack Then**

**DAAuthors.Fill(DsAuthorsBooks1, "Authors")**

**Authors.DataBind()**

**Authors\_SelectedIndexChanged(Me, e)**

**End If**

والآن سنقوم بالمثل لملء قائمة الكتب..

أنشئ موصل بيانات جديدا بنفس الطريقة التي اتبعناها مع الموصل DASelectedAuthor ولكن سمّه هذه المرّة DABooks، وصمّم جملة الاستعلام الخاصة به كالتالي:

**SELECT Book, ID**

**FROM Books**

**WHERE (AuthorID = ?)**

ثمّ أنشيء مجموعة بيانات لهذا الموصل، وسمّها DsBooks.

وقم بضبط خصائص قائمة الكتب كالتالي:

|  |  |
| --- | --- |
| الخاصية | القيمة |
| DataSource | DsBooks1 |
| DataMember | Books |
| DataTextField | Book |
| DataValueField | ID |

تعرف الآن ما يجب فعله.. يجب أن نُمدّ المعامل الخاص بهذا الموصل بقيمة مناسبة (رقم المؤلف) كلّما تغيّر المؤلف المحدّد في القائمة.. ويمكن أخذ رقم المؤلف من خاصية DataValueField الخاصة بقائمة المؤلفين، أو من اللافتة LbAuthor.. عدّل الحدث SelectedIndexChanged ليصير كالتالي:

**Private Sub Authors\_SelectedIndexChanged(ByVal sender As \_**

**System.Object, ByVal e As System.EventArgs) \_**

**Handles Authors.SelectedIndexChanged**

**DASelectedAuthor.SelectCommand.Parameters(0).Value = \_**

**Authors.SelectedItem.Value**

**DASelectedAuthor.Fill(DsAuthor1, "Authors")**

**LbAuthors.DataBind()**

**DABooks.SelectCommand.Parameters(0).Value = LbAuthors.Text**

**DABooks.Fill(DsBooks1)**

**Books.DataBind()**

**End Sub**

لم تبقَ إلا خطوة أخيرة، وهي الخاصة بربط اللافتة التي ستعرض رقم الكتاب.. لن تختلف الخطوات عن تلك التي فعلناها مع لافتة رقم المؤلف.. فقط سنستخدم جملة الاستعلام التالية عند إنشاء موصل البيانات DASelectedBook:

**SELECT ID**

**FROM Books**

**WHERE (ID = ?)**

مع ضبط خصائص الربط بالطريقة المألوفة.. ولا تنس أن تغيّر خاصية AutoPostBack الخاصة بقائمة الكتب إلى القيمة True.

وفي الحدث SelectedIndexChanged الخاص بقائمة الكتب، اكتب ما يلي:

**Private Sub Books\_SelectedIndexChanged(ByVal sender As \_**

**System.Object, ByVal e As System.EventArgs) \_**

**Handles Books.SelectedIndexChanged**

**DASelectedBook.SelectCommand.Parameters(0).Value = \_**

**Books.SelectedItem.Value**

**DASelectedBook.Fill(DsSelectedBook1)**

**LbBooks.DataBind()**

**End Sub**

ولا تنس أن تستدعي هذا الحدث من حدث تغيّر العنصر المحدّد في قائمة المؤلفين، حتّى يعرض رقم أوّل كتاب لكلّ مؤلّف:

**Private Sub Authors\_SelectedIndexChanged(ByVal sender As \_**

**System.Object, ByVal e As System.EventArgs) \_**

**Handles Authors.SelectedIndexChanged**

**DASelectedAuthor.SelectCommand.Parameters(0).Value = \_**

**Authors.SelectedItem.Value**

**DASelectedAuthor.Fill(DsAuthor1, "Authors")**

**LbAuthors.DataBind()**

**DABooks.SelectCommand.Parameters(0).Value = LbAuthors.Text**

**DABooks.Fill(DsBooks1)**

**Books.DataBind()**

**Books\_SelectedIndexChanged(Me,e)**

**End Sub**

**ملاحظات:**

1- لعلك تذكر أنّ نفس هذا البرنامج بالضبط أنشأناه [في تطبيقات الويندوز](file:///C:\\Users\\Administrator\\Desktop\\vb.net\\04-%20Databases%20قواعد%20البيانات\\17-%20تطبيقات%20قواعد%20البيانات\\6-%20ربط%20البيانات\\3-%20ربط%20الأدوات%20الأكثر%20تعقيدا\\5-%20مشروع%20الكتب.htm) باستخدام موصل بيانات واحد ومجموعة بيانات واحدة، وقمنا بكلّ العمل بدون كتابة حرف واحد من الكود!

2- كان من الممكن أن نضع في اللافتتين قيمة رقم الحقل باستخدام خاصية Value الخاصة بالعنصر المحدّد في كل قائمة مباشرة!!.. ولكنّ هذا مثال للتعلّم.. وفي البرامج الحقيقيّة، لن تحتاج لعرض رقم الحقل، وستكون هناك حقول أخرى كثيرة ستعرض محتوياتها بنفس الطريقة التي اتبعناها هنا.

3- مع كلّ مرّة يتمّ تحديث الصفحة فيها، يتمّ إنشاء مجموعات البيانات وتحميلها مجدّدا منذ البداية.. هذا معناه أنّ أيّ تغيير في أيّ حقل (هذا غير ممكن في برنامجنا هذا ولكنّه ممكن في غيره طبعا) سيتمّ تجاهله بمجرّد الانتقال إلى أيّ حقل آخر، وليس فقط عند إغلاق التطبيق (كما كان في تطبيقات ويندوز).. أرجوك لا تنس أنّك تتعامل مع خادم Server، يمكن أن يفتح من نفس الصفحة الآلاف في نفس اللحظة.. لهذا فهو لا يحتفظ إلا بأقلّ قدر ممكن من المعلومات حول كلّ صفحة، وإلا اختنقت ذاكرته وتوقّف عن الاستجابة من فرط الضغط عليه..

إذن فكيف نتصرّف عندما نرغب في حفظ التعديلات إلى قاعدة البيانات؟

في هذه الحالة عليك بقراءة القيم التي تعرضها الأدوات على الصفحة واستخدام جمل SQL أو الإجراءات المخزنة Stored procedures لحفظها في قاعدة البيانات.

**مشروع AuthorsBooks**

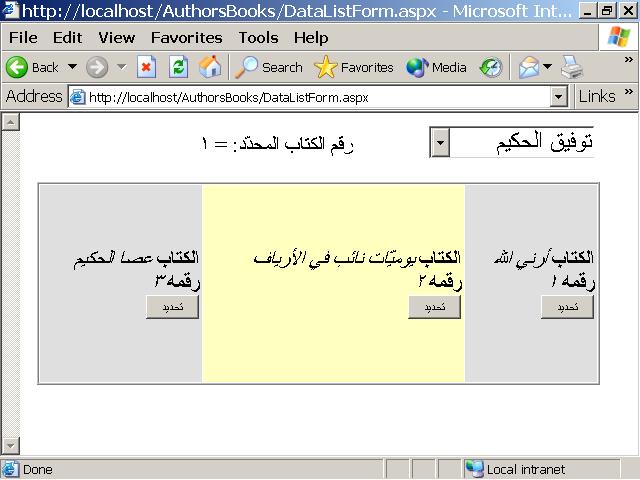
* [**فكرته:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\4-%20مشروع%20AuthorsBooks\1-%20فكرته.htm)
* [**تصميم قائمة البيانات DataList**:](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\4-%20مشروع%20AuthorsBooks\2-%20تصميم%20قائمة%20البيانات.htm)
* [**تصميم جدول البيانات:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\4-%20مشروع%20AuthorsBooks\3-%20تصميم%20جدول%20البيانات.htm)

**فكرته:**

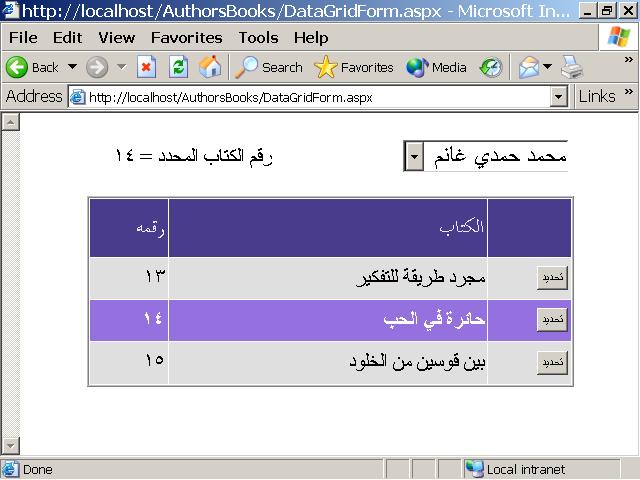
في هذا المقطع سننشئ تطبيق إنترنت يسمح للمستخدم باختيار المؤلف، لتظهر كتبه، مرّة في الأداة DataList، ومرّة في الأداة DataGrid.

هذا المشروع سيحتوي على نموذجين، وعليك أن تغير نموذج بدء التشغيل كلّما أردت أن تستخدم أحدهما.

هذا هو النموذج الذي يستخدم قائمة البيانات DataList:



وهذا هو الذي يستخدم جدول البيانات DataGrid:



إنّ زرّ التحديد في بداية كلّ صفّ، يسمح لك بتحديد هذا الصفّ (في هذه الحالة يتمّ عرضه بخطّ سميك وبلون خلفيّة مختلف).

ابدأ بتكوين الموصلين DAAuthors و DABooks.. الأوّل يحتوي على جدول المؤلفين.. والثاني يحتوي على الكتب الخاصة بمؤلف محدد.. (استرشد بالمشروع السابق).

ثمّ أنشئ مجموعتي البيانات منهما.

طبعا تلاحظ وجود القائمة المنسدلة التي ستعرض أسماء المؤلفين كما في المشروع السابق، والتي سنملؤها أثناء تحميل الصفحة بنفس الكود الذي استخدمناه سابقا:

**If Not Me.IsPostBack Then**

**DAAuthors.Fill(DsAuthors1, "Authors")**

**Authors.DataBind()**

**Authors\_SelectedIndexChanged**(**Me**, e)

**End If**

**Private Sub Authors\_SelectedIndexChanged(ByVal sender \_**

**As System.Object, ByVal e As System.EventArgs) \_**

**Handles Authors.SelectedIndexChanged**

**DABooks.SelectCommand.Parameters(0).Value = \_**

**Authors.SelectedItem.Value**

**DABooks.Fill(DSBooks)**

**DataGrid1.DataBind()**

**End Sub**

لا جديد في هذا الكود إذن..

فدعنا إذن نوجّه اهتمامنا لتصميم الأداتين:

**تصميم قائمة البيانات DataList**:

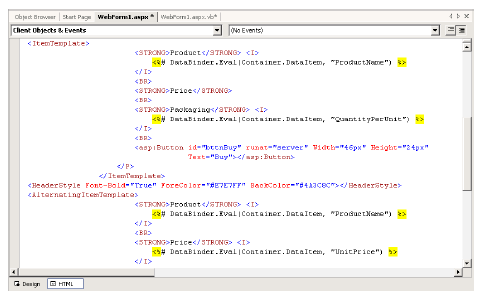
اضغط بزرّ الفأرة الأيمن على الأداة ، واختر القائمة الفرعيّة Edit Templates، ومنها اضغط الأمر Item Templates.

الآن سيتحوّل شكل الأداة ليسمح لك بتحرير كلّ عنصر في القائمة.. اكتب الأسماء، التي تريدها.

كما يمكنك أن تضيف الأزرار التي تريدها (لو واجهت صعوبة في ذلك، فيمكنك أن تضع الزرّ على الصفحة، ثمّ تقصّه Cut وتضغط بزرّ الفأرة الأيمن على القائمة DataList في وضع التحرير، ثمّ تضغط الأمر Paste).

أو يمكنك أن تستخدم كود HTML لتصميم العنصر، كالتالي:

اضغط شريط HTML الموجود أسفل الصفحة.. سيظهر لك كود الصفحة:



ابحث عن العناوين التي أدخلتها ("الكتاب"، "رقمه"):

ستجده كالتالي:

<ItemTemplate>

<STRONG>الكتاب</STRONG><BR>

</ItemTemplate>

<FooterStyle ForeColor="Black" BackColor="#C6C3C6"> </FooterStyle>

<HeaderStyle Font-Bold="True" ForeColor="#E7E7FF" BackColor = "#4A3C8C"> </HeaderStyle>

<AlternatingItemTemplate>

<P><STRONG>الكتاب</STRONG></P>

<P><STRONG>رقمه</STRONG></P>

</AlternatingItemTemplate>

عدّل هذا الجزء من الكود ليصبح كما يلي:

<ItemTemplate>

<STRONG>الكتاب</STRONG>

<I><%# DataBinder.Eval(Container.DataItem, "Book") %></I>

<BR><STRONG>رقمه</STRONG>

<I><%# DataBinder.Eval(Container.DataItem, "ID") %></I>

<BR><asp:Button id="bttnSeect" runat="server" width="53px" height="24px"

text="تحديد"></asp:Button>

</ItemTemplate>

<HeaderStyle Font-Bold="True" ForeColor="#E7E7FF" BackColor="#4A3C8C">

</HeaderStyle>

<AlternatingItemTemplate>

<STRONG>الكتاب</STRONG>

<I><%# DataBinder.Eval(Container.DataItem, "Book") %></I>

<BR><STRONG>رقمه</STRONG>

<I><%# DataBinder.Eval(Container.DataItem, "ID") %></I>

<BR><asp:Button id="bttnSelect" runat="server" width="53px" height="24px"

text="تحديد"></asp:Button>

</AlternatingItemTemplate>

لاحظ ما يلي:

- أنّنا استخدمنا عنصر التنسيق <BR> لإضافة سطر جديد، بدلا من العنصر </P> الذي يمثّل فقرة جديدة، وذلك لأنّ الأخير يضيف سطرا زائدا غير مرغوب.

- أنّنا أضفنا الكود المناسب بعد كلّ عنوان، (الكتاب، رقمه) ليتمّ كتابة قيمة الحقل المناظر من قاعدة البيانات.. هذا هو الكود الذي يضيف اسم الكتاب:

<%# DataBinder.Eval(Container.DataItem, "Book") %>

- <AlternatingItemTemplate> هو تنسيق الخانات الزوجيّة في القائمة، حيث يمكنك أن تعرض العناصر الفرديّة بتنسيق معيّن، والعناصر الزوجيّة بتنسيق آخر.

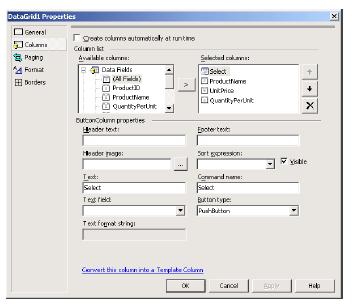
بعد أن تفرغ من تحرير قالب القائمة، عد إلى تصميم الصفحة واضغط الأمر End Template Editing من القائمة الموضعيّة.

**تصميم جدول البيانات:**

نريد أن نضيف أزرار تحديد في خانات العمود الأوّل..

اضغط بزرّ الفأرة الأيمن على الجدول، واختر الأمر Property Builder.. وفي مربّع الحوار الموضّح بالشكل، اختر عنصر الأعمدة Columns.. أزل علامة الاختيار من أمام الخيار Create Columns Automatically At Runtime.

بعدئذ أضِف العنصر Button Column من قائمة الأعمدة المتاحة Available Columns إلى قائمة الأعمدة المختارة Selected Columns، وذلك بضغط الزرّ الذي يحمل الرمز "<"..



اجعل لهذا العنصر الخصائص التالية:

|  |  |
| --- | --- |
| Text | تحديد |
| CommandName | Select |
| HeaderText | تحديد |

عندما سيتمّ ضغط زرّ التحديد في أيّ صفّ، سيتمّ إطلاق الحدث ItemCommand.. اكتب في هذا الحدث ما يلي:

**Private Sub DataGrid1\_ItemCommand(ByVal source As Object, \_**

**ByVal e As DataGridCommandEventArgs) \_**

**Handles DataGrid1.ItemCommand**

**Dim itm As Integer = e.Item.ItemIndex()**

**Dim keys As DataKeyCollection = DataGrid1.DataKeys()**

**Label1.Text = " رقم الكتاب المحدد = " & keys(itm).ToString**

**End Sub**

يمثّل الكائن e.Item الصفّ المحدّد في الجدول.. ويمثّل الكائن DataKeys مجموعة كلّ رموز الجدول keys.

وهناك طريقة أخرى لفعل نفس الأمر كالتالي:

**Private Sub DataGrid1\_SelectedIndexChanged(ByVal sender As \_**

**System.Object, ByVal e As System.EventArgs) Handles \_**

**DataGrid1.SelectedIndexChanged**

**Label1.Text = " رقم الكتاب المحدد= " & \_**

**DataGrid1.SelectedItem.Cells(2).Text**

**End Sub**

**شراء الكتب على الإنترنت**

سننشئ هنا تطبيق BookBasket.. وهو تطبيق جيّد للتدريب على عدد من المفاهيم.. وستجده بإذن الله شيّقا، ونافعا عند بناء تطبيقات مماثلة.

* [**نموذج اختيار الكتب:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\5-%20شراء%20الكتب%20على%20الإنترنت\1-%20نموذج%20اختيار%20الكتب.htm)
* [**تقسيم مجموعات البيانات الضخمة إلى صفحات Paging Large DataSets:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\5-%20شراء%20الكتب%20على%20الإنترنت\11-%20تقسيم%20مجموعات%20البيانات%20الضخمة%20إلى%20صفحات.htm)
* [**نموذج المشتريات:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\5-%20شراء%20الكتب%20على%20الإنترنت\2-%20نموذج%20المشتريات.htm)
* [**نموذج الشراء:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\20-%20برمجة%20قواعد%20البيانات%20على%20النت\5-%20شراء%20الكتب%20على%20الإنترنت\3-%20نموذج%20الشراء.htm)

**نموذج اختيار الكتب:**

يبدأ تطبيقنا بصفحة تسمح للمستخدم باختيار الكتب ووضعها في سلّة التسوّق shopping basket.. وحتّى لا نعرض كلّ أسماء الكتب (أنت تعرف أنّ التطبيقات الحقيقيّة تتعامل مع مئات الآلاف من الكتب)، فسنسمح للمستخدم بكتابة أيّ جزء من اسم الكتاب (في التطبيقات الحقيقيّة يمكن تصنيف الكتب تبعا لموضوعها).



وكما ترى، يحتوي العمود الأخير على روابط hyperlinks لوضع الكتاب المناظر في كلّ صفّ في سلة المشتريات (لو كان الكتاب موجودا في السلة بالفعل، فسيتمّ زيادة عدد النسخ المشتراة منه).

أنشيء موصل البيانات DABooks الذي يستخدم جملة التحديد التالية:

**SELECT Books.Book, Books.ID, Authors.Author**

**FROM Books INNER JOIN**

**Authors ON Books.AuthorID = Authors.ID**

**WHERE (Books.Book LIKE ?)**

لاحظ أنّنا نستخدم المعامل LIKE، وذلك حتّى نسمح للمستخدم باستخدام العلامة "%" كبديل لأيّ عدد من الحروف، والعلامة "\_" كبديل لحرف واحد.. وبذلك يستطيع البحث عن اسم الكتاب بمرونة.

**ملحوظة:**

يمكنك تعديل جملة التحديد لتصبح كالتالي:

**SELECT TOP 100 Books.Book, Books.ID, Authors.Author**

**FROM Books INNER JOIN**

**Authors ON Books.AuthorID = Authors.ID**

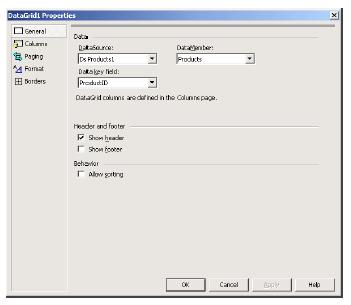
**WHERE (Books.Book LIKE ?)**

وبهذا تتلافى الحالة التي يكتب فيها المستخدم في مربّع النصّ العلامة "%" فقط، ممّا يعمل على تحميل كلّ الكتب من قاعدة البيانات عبر الإنترنت (في التطبيقات الحقيقيّة قد يصل هذا لمئات الآلاف من الكتب) وهو ما لن يكون عمليّا على الإطلاق.. لهذا فسنكتفي بتحميل أوّل 100 كتاب فقط.. وعلى المستخدم أن يكتب عددا أكبر من الحروف للحصول على النتيجة المطلوبة.

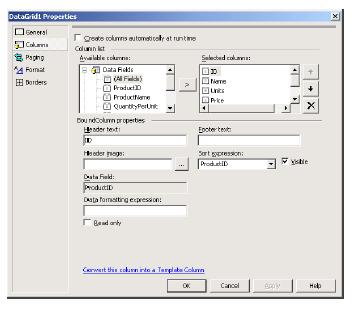
اضبط خصائص ربط البيانات الخاصة بجدول البيانات DataGrid كما تعوّدنا..

واجعل للخاصيّة AutoGenerateColumns القيمة False، وذلك حتّى ننشئ نحن أعمدة الجدول بطريقتنا.

من القائمة الموضعيّة الخاصة بجدول البيانات، اضغط الأمر Property Builder.. في مربّع الحوار ستظهر لك خصائص ربط البيانات، ضمن العنصرGeneral .



انتقل إلى العنصر Columns، وتأكّد من أنّ الاختيار Create Columns Automatically At Runtime غير مفعّل.



لقد رأينا سابقا كيف يمكننا إنشاء الأعمدة وتحديد خصائصها بواسطة مربّع الحوار هذا.. لاحظ أنّ لدينا هذه الأنواع من الأعمدة:

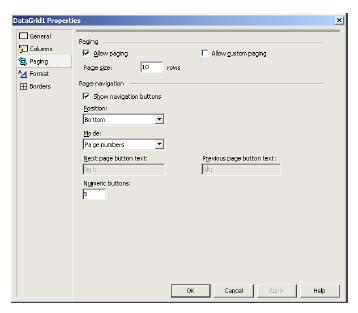
|  |  |
| --- | --- |
| أعمدة قاعدة البيانات | وذلك باختيار All Fields أو باختيار اسم الحقل الذي تريده. |
| عمود أزرار  Button Column | وتحته الأنواع التالية:  زرّ التحديد Select.  أزرار التحديد والتحديث والإلغاء Edit, Update, Cancel.  زرّ الحذف Delete. |
| عمود وصلات فائقة  Hyperlink Column | بدلا من الزرّ، يمكنك وضع وصلة فائقة في خانات العمود، للقيام بالمهمّة المطلوبة عند الضغط عليها. |
| عمود مخصّص  Template Column | هل تذكر كيف حرّرنا قائمة البيانات.. يمكنا فعل المثل مع هذا النوع من الأعمدة، وبنفس الطريقة (ضغط العمود بالفأر واختيار الأمر Edit Template من القائمة الموضعيّة)، لإضافة الأزرار والنصوص وغيرها. |

أضف عمود وصلات فائقة، واجعل نصّ الوصلة: شراء.

انتقل الآن إلى العنصر Paging.. في هذه الخيارات يمكنك تحديد كيفيّة عرض السجلات عندما تكون أعدادها كبيرة.

يمكنك تفعيل الخيار "السماح بتعدّد الصفحات" Allow Paging، مع قدرتك على اختيار أقصى عدد للسجلات يظهر في كلّ صفحة (وليكن 5).. في هذه الحالة ستظهر في أسفل جدل البيانات، أرقام الصفحات التي يمكن عرضها للمستخدم، حيث ستبدو جميع هذه الأرقام كوصلات فائقة (ما عدا رقم الصفحة المعروضة حاليّا)، وعند ضغط أيّ رقم، يتمّ تحميل الصفحة المناظرة له.

ويمكنك أن تختار طريقة أخرى غير أرقام الصفحات، وذلك باختيار Next, Previous Buttons من القائمة المنسدلة Mode، حيث يعمل ذلك على عرض زرّي التالي والسابق، بدلا من أرقام الصفحات.



بقي عنصران، هما Format و Borders.. ومن خلالهما يمكنك تنسيق مظهر جدول البيانات (مظهر الصفوف الأحاديّة والزوجيّة) ومظهر إطاراته.

والآن ننتقل لكود زرّ البحث.. لا أعتقد أنّ به جديدا:

**DABooks.SelectCommand.Parameters(0).Value = TextBox1.Text**

**DABooks.Fill(DsBooks1, "Books")**

**DataGrid1.DataBind()**

**DataGrid1.Visible = True**

عند ضغط زرّ الشراء في أيّ صفّ، علينا أن نضيف رقم الكتاب ID إلى سلة المشتريات.. وفي تطبيقنا هذا ستكون السلّة عبارة عن كائن من النوع Session، حيث سنضيف متغيّرا باسم رقم الكتاب، ونجعل قيمته مساوية لعدد مرات طلب هذا الكتاب.

وسنكتب الكود في الحدث ItemCommand والذي يستجيب لضغط واصلة الشراء كالتالي:

**Private Sub DataGrid1\_ItemCommand(ByVal source As Object, e As \_**

**DataGridCommandEventArgs) \_**

**Handles DataGrid1.ItemCommand**

**' تأكّد أنّ الحدث لم يُستدعَ بسبب ضغط أرقام الصفحات**

**If e.Item.ItemType = ListItemType.Pager Then Exit Sub**

**Dim sItemID As String = e.Item.Cells(0).Text**

**If Session(sItemID) Is Nothing Then ' أوّل مرّة يشترى هذا الكتاب**

**Session(sItemID) = 1**

**Else ' زيادة عدد النسخ المشتراة من هذا الكتاب**

**Session(sItemID) += 1**

**End If**

**End Sub**

طبعا من الأفضل أن تقوم بتعديل هذا الكود، لاستخدام الملفات المؤقتة Cookies، وذلك حتّى توضع كلّ المعلومات على جهاز المستخدم، بدلا من خنق الخادم (راجع الفصل السابق)، مع ملاحظة أنّ هذا قد أدّى عندما جرّبته، لحدوث شعوذة في عرض الصفحة، حيث يتمّ تحويل محاذاتها لتظهر من اليسار لليمين بدون سابق إنذار!!!!

بقيَ أن نكتب كود الزرّ الذي يعرض سلّة المشتريات.. منتهى البساطة:

**Response.Redirect("BasketForm.aspx")**

**تقسيم مجموعات البيانات الضخمة إلى صفحات Paging Large DataSets:**

ليس من المنطقيّ عرض مئات السجلات في صفحة واحدة.. ليس فقط لبطء تحميلها، ولكن أيضا لصعوبة تصفّحها.

لهذا كان الحلّ هو تقسيم هذه السجلات إلى مجموعة صفحات، كما رأينا في نموذج اختيار الكتب.

ولكن حذارِ: فلو زاد عدد الصفحات عن حدّه، لتعذّر على المستخدم الوصول إلى السجلّ المطلوب، إلا بفتح كلّ الصفحات تباعا!

لهذا، فإنّ عليك أن تطالب المستخدم بأن يكون أكثر تحديدا في بحثه، حتّى لا تحتوي نتيجة البحث على أكثر من 100 صفّ ـ مثلا، تعرض كلّ عشرة منها في صفحة.

ولكي تستخدم خاصية التقسيم إلى صفحات paging في جدول البيانات DataGrid، فإنّ عليك أن تجعل للخاصيّة AllowPaging القيمة True، واضعا للخاصيّة PageSize العدد الذي تراه ملائما من السجلات، ليتمّ عرضه في كلّ صفحة.

كما أنّ خاصيّة PagerStyle تسمح لك باختيار مظهر الجزء الخاصّ بعرض أرقام الصفحات، وهي عبارة عن كائن له العديد من الخصائص التي تتحكّم في ألوان وخطوط هذا الجزء.

وفي كلّ مرّة تعرض فيها صفحة أخرى، سيتمّ إطلاق الحدث PageIndexChanged.. هذا هو الكود الذي استخدمناه لعرض الصفحات في نموذج اختيار الكتب:

**Private Sub DataGrid1\_PageIndexChanged(source As Object, e \_**

**As DataGridPageChangedEventArgs) \_**

**Handles DataGrid1.PageIndexChanged**

**DsBooks1.Clear()**

**DABooks.SelectCommand.Parameters(0).Value = TextBox1.Text**

**DABooks.Fill(DsBooks1, "Books")**

**DataGrid1.DataSource = DsBooks1.Books**

**DataGrid1.CurrentPageIndex = e.NewPageIndex**

**DataGrid1.DataBind()**

**End Sub**

وتمثل الخاصيّة e.NewPageIndex الصفحة الجديدة التي اختارها المستخدم.. وعليك أن تستخدمها مع خاصيّة CurrentPageIndex، للانتقال إلى الصفحة الجديدة.. مع ملاحظة أنّك لو وضعت قيمة هذه الخاصيّة بـ -1، فسيتمّ إلغاء الانتقال للصفحة الجديدة.

وكما تلاحظ، فإنّ عليك إعادة ملء مجموعة البيانات.. فلا الخادم ولا العميل يحتفظان بها، حفاظا على ذاكرة الخادم، وحفاظا على حداثة البيانات الواردة للعميل (لا تنس أنّ هناك مستخدمين آخرين يتعاملون مع قاعدة البيانات في نفس اللحظة بالحذف والإضافة والتعديل).. هنا يجب أن تلاحظ أنّ البيانات المطلوبة كلّها يتمّ تحميلها مجدّدا مع كلّ صفحة، ولكنّ جدول البيانات لا يعرض منها فقط إلا الصفحة المختارة.. للأسف لا توجد جملة SQL تستطيع اختيار العدد المطلوب من نتائج البحث، إلا جملة Top فقط، وهي لن تفيد إلا في عرض الصفحة الأولى فقط!!.. وهو ما قد يشكّل عبئا ضخما على أداء البرنامج، في حالة التعامل مع أعداد كبيرة من السجلات.. هذا هو السبب الذي جعلنا نستخدم التعبير TOP 100 لوضع حدّ أقصى لعدد السجلات المعادة.

**نموذج المشتريات:**

هذا النموذج يعرض فقط الكتب المشتراة، دون أن يسمح للمستخدم بتعديل أيّ بيانات.



وبإمكان المستخدم ضغط زرّ للخلف Back الخاص بمتصفّح الإنترنت، للعودة لصفحة اختيار الكتب.

لاحظ أنّ محتويات الكائن Session لن تزول إلا بإغلاق الصفحة نهائيّا.. ربّما عليك أن تفكّر في إضافة زرّ يسمح للمستخدم بمحو هذه البيانات.. أو عليك تطوير صفحة اختيار الكتاب بإضافة عمود لجدول البيانات، به أزرار أو وصلات، عند الضغط عليها يتمّ حذف رقم الكتاب.. فليكن اسم الأمر الجديد RemoveItem.. هذا هو الكود الذي يجب أن تضيفه لبداية الحدث ItemClicked:

**If e.CommandName = "RemoveItem" Then**

**If Not Session(sItemID) Is Nothing Then**

**Session.Remove(sItemID)**

**End If**

**Exit Sub**

**End If**

أضِف للنموذج موصل البيانات DABooks.. يمكنك أن تضيف له أمر التحديد في وقت التصميم كما هو معتاد، بناء على جملة تحديد تعيد كلّ الكتب ذات الأرقام الواردة في مقطع IN (ستضع هذه الأرقام كمعاملات طبعا):

**SELECT ID, Book FROM Books WHERE ID IN (?,?,?)**

المشكلة هي أنّك لا تستطيع أن تعرف عدد المعاملات المطلوبة!

لقد جرّبت أن أستخدم معاملا واحدا، وأمنحه الأرقام تفصل بينها علامة "," فاكتشفت أنّ VB يزيل هذه العلامات عند التنفيذ، ليتعامل مع الأرقام كعدد واحد!

هنا يمكنك أن تستخدم مجموعة المعاملات Parameters Collection الخاصة بأمر التحديد الخاصّ موصّل البيانات، لإضافة المعاملات في وقت التنفيذ.. أظنّك أصبحت محترفا في التعامل مع المجموعات:

**DABooks.SelectCommand.Parameters.Add("Par1",** \_

**Data.OleDb.OleDbType.Integer)**

عموما.. دعنا من مسألة المعاملات هذه.. ولنعد لطريقة مألوفة لنا في VB، وهي تشبيك النصوص.. انظر لهذا الكود:

**Dim cmd As String, ck As Object**

**cmd = "SELECT SELECT ID, Book FROM Books WHERE ID IN ("**

**For Each ck In Session**

**cmd & = ck.ToString & ", "**

**Next**

**cmd = cmd.Substring(0, cmd.Length - 2) & ")"**

بهذه الطريقة يمكننا تكوين جملة SQL بدون استخدام المعاملات.

ولكن انتبه: لو لم يختر المستخدم أيّ كتاب، فسينتج مقطع IN بقوسين فارغين، وهذا سيسبب خطأ عند التنفيذ.. لهذا فلا بدّ من التحقٌّ من هذه الحالة كالتالي:

**If cmd.EndsWith(", ") Then**

**cmd = cmd.Substring(0, cmd.Length - 2) & ")"**

**' ………………..**

**Else**

**lblBasket.Text = "لم يتمّ اختيار أيّ كتب"**

**End If**

وهذه صيغة بديلة لنفس الكود السابق، بدلا من استخدام الجملة التكراريّة:

**Dim S(Session.Count - 1) As String, Exp As String**

**Session.CopyTo(S, 0)**

**Exp = Join(S, ", ")**

**If Trim(Exp) <> "" Then**

**cmd &= Exp & ")"**

**' ………………..**

**Else**

**lblBasket.Text = "لم يتمّ اختيار أيّ كتب"**

**End If**

والكود التالي (سيوضع مكان النقاط في الكود السابق) يختصّ بفتح قاعدة البيانات وإحضار السجلات (باستخدام قارئ البيانات DataReader) وعرضها في اللافتة على شكل جدول باستخدام تنسيق HTML:

**OleDbConnection1.Open()**

**DABooks.SelectCommand.CommandText = cmd**

**DABooks.SelectCommand.CommandType = CommandType.Text**

**Dim Reader As Data.OleDb.OleDbDataReader**

**Reader = DABooks.SelectCommand.ExecuteReader**

**Dim BookName As String, BookID As Integer**

**Dim table As String = "<table border>"**

**Table &= "<tr>"**

**table &= "<td><b>اسم الكتاب</b></td><td><b>عدد النسخ</b></td>"**

**While Reader.Read**

**Dim No As Integer**

**table &= "<tr><td>"**

**BookName = Reader.Item("Book")**

**BookID = Reader.Item("ID")**

**table &= BookName**

**No = Session(CStr(BookID))**

**total += No**

**table &= "</td><td align=right>" & Trim(No)**

**End While**

**table &= "</tr></tsble>"**

**table &= "<b>سلة المشتريات:" & "</b>"**

**table &= "<BR><b> عدد النسخ المشتراة =" & total.ToString**

**table &= "</b></BR>"**

**lblBasket.Text = table**

**OleDbConnection1.Close()**

الآن علينا أن نكتب كود زرّ الشراء، الذي يقود المستخدم لصفحة إدخال بياناته وطريقة الدفع:

وسنرسل لهذه الصفحة عدد الكتب التي تمّ شراؤها، لحساب التكلفة (افرض جدلا ـ جدلا يعني ـ أنّ كلّ الكتب متساوية الأثمان).

هنا يتعيّن علينا تكوين عنوان الصفحة الجديدة بحيث يحتوي على معامل، كالتالي:

**BuyForm.aspx?Total=15**

هذا هو الكود الذي يفعل هذا:

**Response.Redirect( "BuyForm.aspx?Total=" & \_**

**Server.UrlEncode(total) )**

**نموذج الشراء:**

في هذا النموذج سيُدخل المستخدم المعرّف الخاص به ID.. (نظرا لأنّ قاعدة البيانات التي لدينا ليس فيها جدول للعملاء، فسنستخدم جدول المؤلفين مجازا!)

مع اختيار طريقة دفع النقود.



عندما يضغط المستخدم زرّ Login، سنستعيد السجلّ الذي يحمل الرقم الموجود في مربّع النصّ، ونعرض اسم المشتري في مربّع النصّ الثاني.. عندئذ سيتمّ تفعيل زرّ الشراء، للسماح للمستخدم بضغطه.

فإذا لم يكن الرقم صحيحا، فسيتمّ تحذير المستخدم:

**DABooks.SelectCommand.Parameters(0).Value = TextBox1.Text**

**DABooks.Fill(DsBooks1)**

**If DsBooks1.Books.Rows.Count <> 1 Then**

**lblError.Visible = True**

**Else**

**lblError.Visible = False**

**Dim Address As String = DsBooks1.Books.Item(0).Author**

**txtAddress.Text = Address**

**bttnShip.Enabled = True**

**CustomerID.Value = TextBox1.Text**

**End If**

ولا تنس أن تعرض عدد النسخ المشتراة في اللافتة، عند تحميل النموذج:

**lblTotal.Text = "عدد النسخ المشتراة = " & Request.QueryString("Total")**

**عرض الجدول الأساسي والجداول الفرعيّة Master-Detail:**



يعرض جدول البيانات الأيمن أسماء المؤلفين (5 في كلّ مرّة)، بينما يعرض الجدول الآخر كتب المؤلفّ المحدّد (لا تستخدم هذه الطريقة في العرض مع الجداول الضخمة.. أعتقد أنّك تعرف السبب.. أو استخدم عناصر بحث لتقليص عدد السجلات المعروضة ـ مثل حروف من اسم المؤلف أو الكتاب أو أيّ تفاصيل أخرى).

لا أعتقد أنّ هذا التطبيق يحتاج لأيّ شرح.. عليك أن تفحصه في مجلّد برامج هذا الفصل.

مع ملاحظة أنّه في التطبيقات الحقيقيّة، سيكون هناك جداول أخرى، مثل جدول تفاصيل الكتب (تاريخ التأليف، عدد الصفحات، الناشر، المضمون... إلخ).. في هذه الحالة عليك أن تضع عمود أزرار لجدول الكتب، لتعرض تفاصيل الكتاب المحدّد في جدول بيانات ثالث.. وهكذا.