Database and Website Programming in your Hands

https://www.facebook.com/yourcommands

دورة في ADO.NET

تمام قمر الدين

تحية طيبة لكل محبيني ومحبي صفحتي على الفيسبوك ولكل من قام بدعمي لايصال هذا الكتاب لمتناول أيديكم.

أقدم هذا الكتاب لكم بأبسط صورة باحثا عن السهولة في وصول المعلومات إليكم بأقرب طريقة مراعيا الظروف الصعبة التي يمر بها أي شخص فيكم.

أنا مثلكم في ايام سلفت كنت أتعب في الدراسة والحصول على المعلومات من مصادر متعددة وقد كان أغلبها معقدا في الشرح ويحوي على الكثير من الأمور التي أنا ست بحاجة إليها وقد قمت بمراعاة الكثير من الأمور لكي تكون المعلومة جيدة للجميع.

الكتاب بسيط جدا ولا يتناسب مع المتقدمين في مجال البرمجة وإنما هو للمبتدئين والمتوسطين في الخبرة.

أهدي الكتاب لكم أنتي ولأهلي الذين تعبوا لكي أصبح انسانا متعلما ومثقفا وبذلوا في ذلك الغالي والنفيس وأهديه لمحبوبة قلبي وشريكتي في حياتي وأهديه لكل أصدقائي وأصحابي ورواد صفحتي المتواضعة على الفيس بوك.

ولن أطيل عليكم وسأبدأ بالشرح بعد الفهرس مباشرة.

الفهرس

دورة في - ADO.NET الدرس الأول

البداية عنADO.NET !!!!

هي عبارة عن مجموعة فئات أو صفوف Classes مشمولة موجودة ضمن فضاء الأسماء System.Data غرضها الوصول غلى مصادر البيانات سواء كانت أكسس أو أوراكل أو سيكوال سيرفر أو غيرها مما يعني أنك قادر على الوصول إلى مصادر البيانات أو قواعد البيانات.

لماذا تدعى ADO.NET بهذا الاسم ؟

لقد أخذت ADO.NET اسمها من تقنية سابقة تدعى ADO و هي اختصار للكلمات ActiveX data Object و تمثل ADO مجموعة من الأصناف المستخدمة في لغات البرمجة السابقة كالفيجوال بيزك 6 المستخدمة للوصول إلى البيانات في قواعد البيانات العلائقية و مصادر البيانات الغير علائقية أيضا و قد اعتمدت مايكروسوفت على هذه التسمية لكي تشير أنADO.NET هي التقنية المفضلة للوصول إلى البيانات من قبل مبرمجيNET.
تخدم ADO.NET نفس الأغراض التي تخدمها ADO و لكن بأسلوب محدث و أسهل و غرضي التوجه أكبر من ذي قبل و ذلك للتكيف مع إطارNET.

الإختلافات بين ADO.NET و ADO !!!!

أولا ADO :

1- مصممة للعمل في بيئة متصلة باستمرار مع قاعدة البيانات.
2- تستخدم الكائن RecordSet للاحتفاظ بمجموعة بيانات واحدة.
3- تحتوي على أنواع متعددة من المؤشرات Cursors المستخدمة لأغراض مختلفة حيث أن لكل مؤشر من المؤشرات الإمكانيات الخاصة به.
4- تخزين البيانات في هيئتها الثنائية وبالتالي يكون ارسال عبر جدران الحماية صعب جدا وإضافة لذلك تعتبر غير مفيدة في الأنظمة الغير داعمة للـ ADO .
5- تستهلك الكثير من مصادر النظام وذلك لكونها متصلة دائما بقاعدة البيانات أثناء عمليات المعالجة.

قابلية التوسع

إن تقنية ADO.NET قابلة للتوسع و التطور فهي توفر إطار عمل لمزودي بيانات NET.بحيث يمكننا إنشاء مزودات جديدة للبيانات تمكننا من الوصول إلى أي مصدر من مصادر البيانات لقد تم دمج مزودين للبيانات في ADO.NET أحدهما للوصول إلى مصادر البيانات في OLE DB و الآخر موجه للوصول للبيانات في قواعد بيانات SQL Server أما بالنسبة لقواعد البيانات العلائقية الأخرى مثل Accessو Oracle و مصادر البيانات غير العلائقية فأنها تستخدم مزود OLE DB هناك أبضا مزود جديد تم طرحه مؤخرا و هو مزود ODBC للبيانات و الذي يسمح لنا الوصول إلى أنظمة أخرى لقواعد البيانات التي تدعم تقنية. ODBC

ثانيا ADO.NET :

1- صممت ADO.NET للعمل في بيئة غير متصلة مع قواعد البيانات على الرغم من أنها يمكنها أن تعمل في البيئة المتصلة.
2- تستخدم الغرض DataSet للاحتفاظ بعدة مجموعات من البيانات.
3- لا تستخدم المؤشرات لأنها أصلا تعمل في بيئة غير متصلة.
4- تخزين البيانات بصيغة XML وهي مصممة للارسال عبر جدران الحماية والشبكات دون مشاكل كما يمكن قراءتها بصيغتها XML بسهولة من العديد من التطبيقات.
5- تعمل وكأنها نظام بيانات منفصل عن قاعدة البيانات فهي لا تتصل بقاعدة البيانات إلا عند الضرورة وبالتالي توفير في استهلاك موارد النظام

هل لاحظتم الإختلافات بينهما؟؟؟ اختلافات جوهرية بكل ما فيها .

تعنمد ADO.NET على شيئين أساسيين في عملها وهما:

- مزودات البيانات.
- مجموعات البيانات.

أولا - مزودات البيانات

كما قلنا سابقا تم دمج مزودي بيانات فيها وهما OLE DB, SQlServer والحديث هنا يطول جدا فهنالك تشعبات كثيرة ولكن من المهم معرفة المزودات الأساسية في ADO.NET حيث أننا عندما نتعامل مثلا مع قواعد بيانات أوراكل علينا لعلم أن مزود البيانات الخاص بها هو OLE DB أما عند التعامل مع قواعد بيانات SQLServer فمزود البيانات هو sqlServer .

ثانيا - مجموعات البيانات

والمقصود بها مجموعة الأدوات التي تساعد في الاتصال بقاعدة البيانات وحفظ البيانات فيها وهي مثل:

- DataSet : وهو مكافئ لعمل RecordSet مع مجموعة تحسينات حيث يمكن له أن يقوم بتخزين أكثر من جدول من قاعدة البيانات أو نتيجة استعلام وهو كائن منفصل عن قاعدة البيانات.
- DataAdapter : وهي جسر يربط بين DataSet وقاعدة البيانات ويدعم أوامر , Select - Update - Delete - Insert لذا هو مزود للـ DataSet بالبيانات.
- DataReader : وهو كائن يستخدم لقراءة البيانات من قاعدة البيانات ويقوم بتخزينها على هيئة جداول بداخله ويقرأ كميات ضخمة من البيانات مثل تلك التي لا يمكن تخزينها بالذاكرة.
- DataRelation : يستخدم هذا الكائن لتمثيل العلاقات بين الجداول في قاعدة البياناتJOIN .
- Connection : وهو الاتصال الذي يتم انشاءه مع قاعدة البيانات لتبادل البيانات منها وإليها.
- Command : يسمح هذا الكائن للكائن DataAdapter بتطبيق الأوامر على قاعدة البيانات.

أرجو أن تكون هذه المقدمة مفيدة لكم في ADO.NET وأنكم استفدتم منها .

دورة في ADO.NET - الدرس الثاني

درس اليوم جميل جدا وهو الاتصال مع قاعدة بيانات، يمكن أن يتم ذلك بطريقتين إما من خلال wizard معالج البيانات الخاص بمصدر البيانات أو من خلال كتابة كود بلغة برمجة معينة.
بهذا الدرس سنقوم بشرح كيفية الاتصال بقاعدة البيانات من خلال wizard .

أصدقائي قوموا بفتح مشروع website جديد في الفيجوال استديو 2010 الخاص بكم أو اي اصدار آخر ولكن معظم عملي سيكون على اصدار 2010.

قوموا بإنشاء موقع فارغ تماما حتى نتعلم من الصفر، ستجدون فيه فقط ملف web.config .

الأن قوموا بإنشاء قاعدة البيانات الخاصة بنا ولتكن مثلا School.mdf ،من خلال النقر بالزر اليميني للفأرة فوق أرضية السيرفر ولنختر منها add new Item .

نختار ملف قاعدة بيانات جديد ونسميه بالاسم الذي اتفقنا عليه ومن ثم نضغ موافق.
الآن اصدقائي ستظهر لكم رسالة مفادها أن قاعدة البيانات يجب أن تكون في مجلد خاص اسمه App_Data والمجلد غير موجود لديك هل تريد انشاء هذا المجلد ؟

الرجاء هنا اختيار نعم وستجد انه تم انشاء المجلد وتم وضع قاعدة البيانات بداخله.
افتح قاعدة البيانات وانشئ جدول جديد بالنقر اليميني فوق مجلد Tables واختر Add New table ....

ضع فيه studID وهو المفتاح الرئيسي من النوع int وقابل للزيادة تلقائيا، واضف الحقل studName وهو نوع nvarchar50 والحقل registerDate من النوع date احفظ الجدول الجديد باسم Students.

ملاحظة : أخوتي هذا النموذج سنعمل عليه طيلة فترة دورتنا وهو قابل للتوسع بحسب الحاجة.

الآن نعود ونقوم بإنشاء ملف جديد وهو صفحة الانترنت التي سنعمل عليها، ضغطة يمينية على ارضية السيرفر ونختار add new item ونختار web form ونتركها بالاسم الافتراضي Default.aspx.

ليتم فتح الصفحة لدينا بوضع Source كود نضغط فوق Design حتى نتمكن من العمل معها.
حتى نقوم بإنشاء اتصال مع قاعدة البيانات الخاصة بنا نحتاج لأداة تقوم بالاتصال وأخرى لعرض

البيانات.

للاتصال سنستخدم sql data source وللعرض سنستخدم Grid View وهم عناصر تحكم في Asp.net.

تضع grid view جديد في الضفحة من خلال السحب والافلات لعنصر التحكم ومن قائمة مهام عنصر التحكم Grid view Tasks نختار من القائمة المنسدلة الخاصة بـ Choose Data Sourceنختار New Data Source .

ليفتح أمامنا معالج لإنشاء الاتصال مع قاعدة البيانات ، الآن نختار نوع قاعدة البيانات لدينا وهي SQL Database ونترك اسم مصدر البيانات كما هو ونضغط زر OK او موافق

الآن علينا اختيار الاتصال الخاص بنا وهو عادة موجود ضمن ملف web.confg ولكن بما أن مشروعنا جديد فالاتصال غير منشء بعد لذا تجد اسم قاعدة البيانات التي تريد الاتصال بها وهو سيقوم بانشاء الاتصال معها بنفسه ويقوم بحفظه ضمن ملف web.config تلقائيا.

نختار اسم قاعدة البيانات ونضغط زر next ليظهر لدينا مربع نصي يطالبنا بوضع اسم جديد للـ connection الذي سيتم انشاءه مع قاعدة البيانات وسيتم تخزينه في ملف web.config هنا نحدد الاسم الذي نرغب وأنا سأتركه بالاسم الافتراضي ConnectionString ونضغط زرNext .

ليظهر لدينا كافة الجداول بقاعدة البيانات ضمن قائمة منسدلة نختار الجدول الذي نرغب بعرض بياناته ليتم عرض كافة حقوله على هيئة Check box list أسفل منه نختار أيضا الحقول التي نرغب بعرضها ونضغط زر next لتظهر شاشة لاختبار الاتصال المنشأ وتنفيذ الاستعلام المناسب بحسب الحقول التي اخترناها ومن ثم نضغ زر الانهاءFinish .

لنجد بالصفحة أنه تم انشاء عنصري تحكم هما sql datasource و grid view وجاهزين لعرض البيانات.

الآن شرحنا كيفية انشاء اتصال مع قاعدة البيانات لكن ماهي المعلومات الخاصة بالاتصال ضمن ملف web.config ؟؟؟؟

اذهب لهذا الملف وقم بفتحه وأقرأ التالي:

```
<connectionStrings>
<add name="ConnectionString" connectionString="Data
Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\School.mdf;Integrated
Security=True;User Instance=True"
providerName="System.Data.SqlClient" />
</connectionStrings>
<system.web>
<compilation debug="false" targetFramework="4.0" />
</system.web>
</configuration>
```

ماتم انشاءه هو القسم الخاص بـ connectionStrings فقط قمنا بإضافة اتصال جديد من خلال التاغ Add يحمل الاسم name="ConnectionString" ويحمل سلسلة الاتصال التالية : connectionString وهي مزودة بمجموعة من البيانات سأقوم بشرحها:

- Source : وهو نوع قاعدة البيانات الخاصة بنا وهي من نوع SQLEXPRESS .
- AttachDbFilename : ملف قاعدة البيانات المرفق حيث أنه موجود ضمن مجلد app_data وهو ما يسمى DataDirectory حيث أن اسم الملف هوSchool.mdf
- Security : تفعيل خيار حماية الاتصال true .
- Instance : انشاء Instance تم تفعيله Ture .

كما تم تزويد الاتصال بروفايدر providerName حيث أنه موجود ضمن فضاء الأسماء System.Data.SqlClient

دورة في ADO.NET - الدرس الثالث

سنقوم في هذا الدرس بالتعلم حول كيفية الاتصال مع قاعدة البيانات ولكن باستخدام كود السي شارب، وكود الـ VB.NET ولن أطيل عليكم من متطلبات العمل صفحة aspx وفيها grid view ليقوم بعرض البيانات وسنستخدم الكود التالي والذي سأقوم بشرحه سطر تلو الآخر :

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;

SqlConnection con = new SqlConnection(connection);

string SqlStatment = "select studID,studName,registerDate from students";
SqlCommand cmd = new SqlCommand(SqlStatment, con);
con.Open();
SqlDataReader dr = cmd.ExecuteReader();
GridView2.DataSource = dr;
GridView2.DataBind();
con.Close();
```

الآن سنبدأ بالسطر الأول والذي مهمته هو أن يقوم بإنشاء سلسلة اتصال مع قاعدة البيانات وهذه السلسلة تستخدم الاتصال الموجود في ملف web.config حيث أن الاتصال هذا مربوط مباشرة مع قاعدة البيانات وتكون عملية الانشاء من خلال تخزين سلسلة الاتصال في متحول نصي string :

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
```

حيث أن ConnectionString هو اسم الاتصال الموجود في ملف web.config

نقوم بعدها بإعداد الاتصال والذي سيستخدم السلسلة الاسبقة للولوج لملف web.config ومنه إلى قاعدة البيانات ويكون من النوع sqlconnection كما يلي :

```
SqlConnection con = new SqlConnection(connection);
```

حيث أنه يأخذ المتحول connection كوسيط يمرر له.

من ثم سنقوم بتخزين الاستعلام الخاص بنا ضمن متحول نصي لنقوم فيما بعد بتنفيذه لجلب البيانات من قاعدة لبيانات وهذا الاستعلام يجب أن يكون مكتوب وفق لغة sql النظامية ويجب مراعاة الدقة أثناء كتابة هذا الاستعلام ولتلافي الأخطاء أخوتي يفضل أن تقوموا بكتابته ضمن أحد محررا SQL والتعليمة تكون كما يلي :

```
string SqlStatment = "select studID,studName,registerDate from students";
```

أما الآن سنقوم بإنشاء أداة تنفيذ الاستعلام السابق والتي تستخدم الاتصال con كأداة للولوج لقاعدة البيانات وستقوم بتنفيذ الاستعلاك السابق وهي كما يلي :

```
SqlCommand cmd = new SqlCommand(SqlStatment, con);
```

نبدأ الأن بالعمل ونقوم بفتح الاتصال مع قاعدة البيانات من خلال التعلمية :

```
con.Open();
```

سنقوم الأن بقراءة البيانات وتخزينها ضمن الذاكرة على هيئة جدول ضمن متحول من نوع sqlDataReader كما يلي :

```
SqlDataReader dr = cmd.ExecuteReader();
```

حيث أن هذا المتحول يقوم بقراءة البيانات التي قام cmd بجلبها من قاعدة البيانات أثناء تنفيذه التابع ExecuteReader() والذي يعمل على جلب أكثر من سطر من قاعدة البيانات.

أما الآن فإن نتيجة الاستعلام أصبحت موجود بالذاكرة ضمن المتحول dr السابق ونحتاج لوسيلة للعرض.

كما قلنا في البداية لدينا grid view وهو سيقوم بعرض البيانات لذا سنقوم الآن بربط مصدر بيانات gridview1 مع المتحول الذي لدينا وهو ds كما يلي :

```
GridView2.DataSource = dr;
```

وسنقوم بعملية تثبيت الربط من خلال عمل Binding كما يلي:

```
GridView2.DataBind();
```

ولا تنسوا إغلاق الاتصال الذي قمتم بفتحه ببداية العمل :

```
con.Close();
```

ولكن ماهو عمل DataBind() ؟؟؟؟

إن هذا التابع يعمل على الوصل ما بين جدول موجود بالذاكرة ضمن متحول dataReader أو DataSet وبين عنصر تحكم مرئي للمستخدم مثل Grid View أو Details View أو غيرهم.

وإليكم الكود مكتوبا بلغة VB.NET ....

```
Dim connection As String = ConfigurationManager.ConnectionStrings("ConnectionString").ConnectionString
Dim con As SqlConnection = New SqlConnection(connection)
Dim SqlStatment As String = "select studID,studName,registerDate from students"
Dim cmd As SqlCommand = New SqlCommand(SqlStatment, con)
con.Open
Dim dr As SqlDataReader = cmd.ExecuteReader
GridView2.DataSource = dr
GridView2.DataBind
con.Close
```

دورة في ADO.NET - الدرس الرابع

التعامل مع DataAdapter و DataSet !!!!

سنتعرف من خلال هذا الدرس التعامل مع كلا من data Adapter و data Set ونتعرف على كلا منهما وماذا يعمل كل واحد فيهم وكيف يمكن استغلالهما في الوصول إلى البيانات في قاعدة البيانات.

أولا DataAdapter :

هو جزء من مزود بيانات ADO.NET يعمل كصلة الوصل ما بين Dataset وبين مصدر البيانات datasource ويقوم هذا الغرض بتحميل البيانات من مصدر البيانات إلى DataSet الخاصة بنا. كما أنه يقوم بتعريف استعلامات SQL الخاصة مثل , Update , Insert , Select Deleteوذلك في مصدر البيانات الخاص بنا.

عند التعامل مع قواعد بيانات sql server فإن الغرض الخاص من DataAdapter هو SqlDataAdapter .

كما يمكن الوصول إليه من خلال فضاء الأسماء System.Data.SqlClient يعمل بشكل صريح هنا كصلة وصل بين DataSet و قاعدة االبيانات حيث يتم تعريفه بالشكل التالي :

:VB.Net

```
Dim adapter As New SqlDataAdapter
```

: C#

```
SqlDataAdapter adapter = new SqlDataAdapter();
```

ثانيا DataSet :

هو عبارة عن قاعدة بيانات علائقية بسيطة كما يمكن تشبيهها ولكن في الذاكرة حيث يمكن أن نقوم فيها بتخزين ناتج استعلام معين على هيئة جدول أو مجموعة جداول ويمكننا فيه الوصول إلى أي جدول وإلى أي سطر من أسطر الجداول الموجودة فيه.

تتميز بأنها تحتاج للاتصال بقاعدة البيانات مرة واحد أ وقت الحاجة فقط فهي تعمل بدون اتصال وتوفر جودة وسرعة بالوصول إلى البيانات، يمكن تعريف هذا الغرض كما يلي :

:VB.NET

```
Dim dataset As New DataSet()
```

:C#

```
DataSet dataset = new DataSet();
```

وإليكم الآن الكود الخاص بنا والذي سنعمل عليه ونقوم بشرحه كاملا:

```
string connection =
ConfigurationManager.ConnectionStrings["ConnectionString"].Connection
String;
SqlConnection con = new SqlConnection(connection);
string SqlStatment = "select studID,studName,registerDate from
students";
SqlDataAdapter da = new SqlDataAdapter(SqlStatment, con);
DataSet ds = new DataSet();
da.Fill(ds);
GridView2.DataSource = ds.Tables[0];
GridView2.DataBind();
```

في البداية وكما تعلمنا قمنا بتعريف سلسلة الاتصال مع قاعدة البيانات وقمنا بتخزين محتواها في متحول نصي connection واستخدمنا هذه السلسلة في تعريف الاتصال مع قاعدة البيانات con وقمنا بتجهيز الاستعلام الخاص بنا SqlStatment كما تعودنا في الدرس السابق:

```
string connection =
ConfigurationManager.ConnectionStrings["ConnectionString"].Connec
tionString;

SqlConnection con = new SqlConnection(connection);

string SqlStatment = "select studID,studName,registerDate from
students";
```

أما الآن سنقوم بتعريف dataAdapter وهو المسؤول كما قلنا عن صلة الوصل ما بين الغرض DataSet وقاعدة البيانات الخاصة بنا كما يلي:

```
SqlDataAdapter da = new SqlDataAdapter(SqlStatment, con);
```

حيث أننا نقوم بتمرير الاستعلام والاتصال له كوسيط دخل حتى يقوم بمهمة تنفيذ الاستعلام ويمكن أن نقوم عن DataAdapter أنه بديل عن sqlcommand ويعمل نفس عمله ولكن عند استخدامنا للـ DatSet لا يمكن استخدام sqlcommand لذا نقوم باستخدم هذا الغرض كبديل له.

ثم نقوم بتعريف DataSet لتخزين القيمة المعادة من الاستعلام ضمن جدول في DataSet بالذاكرة كما يلي :

```
DataSet ds = new DataSet();
```

ثم نقوم الآن بملء البيانات التي قام da بجلبها من قاعدة البيانات ونقوم بوضعها ضمن ds بالتعليمة التالية :

```
da.Fill(ds);
```

وبذلك تكون نتيجة الاستعلام مخزنة كليا ضمن DataSet حيث أن التابع Fill يقوم بنقل البيانات من da إلى ds أو بشكل أدق هي عملية تعبئة.

وأخيرا لعرض البيانات بشكل مرئي نقوم باستخدام Grid View لمشاهدة النتيجة:

```
GridView2.DataSource = ds.Tables[0];
```

```
GridView2.DataBind();
```

لاحظوا معي في السطر الأول

```
ds.Tables[0];
```

قمنا بتحديد الجدول الذي سيتم عرضه ضمن grid view حيث أن الجداول تخزن تبعا لعملية التعبئة لها ضمن ds ابتداءا من الصفر وبشكل متزايد بمقدر واحد، مع التأكيد أنه في حال لا يوجد سوى جدول واحد ضمن ds فيمكننا الاستغناء عن تحديد الجدول وتكون التعليمة بالشكل لتالي :

```
GridView2.DataSource = ds;
```

ملاحظة: dataset موجودة ضمن فضاء الأسماء System.Data إن لم تكن قد مت باستيراده فقم بذلك.

دورة في ADO.NET - الدرس الخامس

اليوم رح نتعلم كيف نستخدم DataReader لملء قائمة منسدلة من مصدر البيانات الخاص فينا لذا رح يكون عنا قائمة منسدلة أولا وDataReader بالخلفية ضمن الكود الخاص فينا رح يشتغل ورح نشرح مجموعة من الأمور لتصير واضحة للكل.

بالدرس الثالث تعلمنا استخدام DataReader ولكن ماشرحنا كتير عنو وعن شغلو بهالدرس رح يكون مخصص إلو لنقرأ البيانات .

الكود الذي سنعمل عليه هو:

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].Connection String;
SqlConnection con = new SqlConnection(connection);
string SqlStatment = "select studID,studName,registerDate from students";
SqlCommand cmd = new SqlCommand(SqlStatment, con);
con.Open();
SqlDataReader dr = cmd.ExecuteReader();
while (dr.Read())
{
      ListItem li = new ListItem();
      li.Text =dr["studName"].ToString();
      li.Value= dr["studID"].ToString();
      DropDownList1.Items.Add(li);
}
dr.Close();
con.Close();
```

كالعادة رح نبدأ من السطور الأولى والتي قد اعتدنا عليها وهي انشاء سلسلة الاتصال connection وانشاء الاتصال con ووضع الاستعلام ضمن متحول نصي SqlStatment .

لنعمل مع الـ DataReader يجب أن نستخدم SqlCommand وهو قد أنشأناه باسم cmd نقوم بفتح الاتصال مع قاعدة البيانات :

```
con.Open();
```

ونقوم بتعريف DataReader خاص بنا وهو يحمل الاسم dr :

```
SqlDataReader dr = cmd.ExecuteReader();
```

يقوم الداتا ريدر بالوصول إلى قاعدة البيانات من خلال الكوماند cmd عندم نقوم بتنفيذ التابع ExecuteReader الخاص به حيث يقوم بالولوج إلى قاعدة البيانات ويقوم بتنفيذ الاستعلام وقراءة البيانات منها ويتم تخزين البيانات ضمنه على هيئة جداول وهي قابلة للقراءة فقط أي لا يمكن إجراء تعديلات عليه مثل DataSet وسنقوم بشرح ذلك ضمن جلسة لاحقة لتبيان بعض الفروقات بينهما.

الآن وبعد أن أصبحت البيانات جاهزة ضمن DataReader سنبدأ بالقراءة منه والكتابة ضمن القائمة المنسدلة، حيث يمكن ذلك من خلال الحلقة While ومن خلال استخدام التابع Read .

```
while (dr.Read())
```

هنا ضمن هذه الحلقة سيستمر dr بالقراءة من أول سطر الجدول المخزن لديه بالذاكرة وحتى آخر سطر فيه وذلك ضمن كل دورة من دورات while يقرأ سطر واحد فقط.

وحتى نقوم بملء القائمة المنسدلة نقوم بتعريف ListItem وهو عنصر واحد من عناصر القائمة المنسدلة يأخذ قيمتين وهما القيمة الظاهرة للمستخدم وهي الاسم Text والقيمة بالخلفية Value والتي لا تظهر للعين وهي قيمة مخفيه.

```
ListItem li = new ListItem();
li.Text =dr["studName"].ToString();
li.Value= dr["studID"].ToString();
```

لاحظو كيفية استخدام dr قمنا بذكر اسم الحقل الذي هو موجود أصلا بقاعدة البيانات ومضمن ضمن الاستعلام studID و studName ولا تنسو تحويل القيم إلى قيم نصية باستخدام التابع toString.

وأخيرا نقوم بإضافة العنصر li إلى القائمة المنسدلة من خلال التعليمة :

```
DropDownList1.Items.Add(li);
```

ومن ثم نقوم بإغلاق dr و con بعد الانتهاء من الحلقة.

```
dr.Close();
con.Close();
```

وهنا كود مكتوب بلغة VB.NET ليسهل على مستخدمي اللغة قراءتها :

```
Dim connection As String = ConfigurationManager.ConnectionStrings("ConnectionString").Conn
ectionString
Dim con As New SqlConnection(connection)
Dim SqlStatment As String = "select studID,studName,registerDate
from students"
Dim cmd As New SqlCommand(SqlStatment, con)
con.Open()
Dim dr As SqlDataReader = cmd.ExecuteReader()
While dr.Read()
      Dim li As New ListItem()
      li.Text = dr("studName").ToString()
      li.Value = dr("studID").ToString()
      DropDownList1.Items.Add(li)
End While
dr.Close()
con.Close()
```

دورة في ADO.NET - الدرس السادس الجزء الأول (تعمق بالمحتوى)

الفرق بين DataReader و الـ DataSet !!!!

: DataReader

1- يحتاج لإتصال طوال وقت عمله Connected mode.
2- البيانات ترسل في إتجاه واحد بشكل تدفق Streaming forward only .
3- للقراءة فقط، لايمكن التعديل على البيانات من خلاله.
4- لا يحتاج إلى ذاكرة كبيرة لانه لا يقوم بعمل تخزين للبيانات في الذاكرةbuffering .
5- يجب غلقه عند الإنتهاء منه لانه يقوم بحجز الاتصال ولا يسمح لأى شيء اخر بالقراءة من نفس الإتصال.

: Dataset

1- لا تحتاج إلى إتصال مستمر كي تعمل disconncted mode.
2- يمكن عرض البيانات بكافة الإتجاهات forward and backward.
3- يمكن القراءة والتعديل عليها.
4- تقوم بتخزين كل البيانات في الذاكرة.
5- يمكنها قراءة وكتابة من ملفات XML وليس فقط من قاعدة البيانات.
6- يمكنك بناء برنامج قاعدة بيانات كامل بإستخدام ملفات XML وبدون الحاجة للإستخدام أي قاعدة بيانات معروفة (لا ينصح للبرامج للبيانات الكبيرة).
7- يمكن أن يتم عمل serialization لذلك يمكن نقل البيانات من خلالها الى الانترنت أو أي جهاز اخر.

دورة في ADO.NET - الدرس السادس الجزء الثاني (تعمق بالمحتوى)

عناصر الاتصال بقاعدة البيانات !!!!

كما علمنا في الدرس السابقة فإن الاتصال بقاعدة البيانات يحتاج إلى سلسلة اتصال بقاعدة البيانات ثم اتصال مع قاعدة البيانات ثم أمر استعلام command الآن سنرى كيفية عمل هذه العناصر بطريقة أخرى.

أولا - سلسلة الاتصال بقاعدة البيانات

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
```

هذه السلسة كما نشاهد تقوم بالاتصال بملف web.config وتقوم بتجهيز الاتصال ببياناته لماذا قمنا بوضعها ضمن متحول نصي ألا يمكننا اسنادها مباشرة للاتصال sqlconnection؟؟؟؟

الجواب : نعم يمكننا ذلك وتتعد الطرق حتى نقوم بإسناد سلسلة الاتصال للاتصال الخاص بها كما في ما يلي:

الطريقة الأولى :

نقوم بتعريف اتصال ونمرر لهذا الاتصال سلسلة الاتصال كمتحول نصي :

```
SqlConnection con = new SqlConnection(connection);
```

الطريقة الثانية:

يمكننا أن نقوم بإنشاء سلسلة الاتصال داخل القوسين كوسيط للاتصال

```
SqlConnection con = new SqlConnection( ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString);
```

الطريقة الثالثة:

يمكننا أن نقوم بتعريف الاتصال من دون تمرير سلسلة الاتصال ومن خلال الخاصية ConnectionString الموجودة في المتحولات من نوعSqlConnection يمكننا اسناد هذا الاتصال كمتحول نصي.

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
SqlConnection con = new SqlConnection();
```

```
con.ConnectionString = connection;
```

الطريقة الرابعة:

يمكننا اسناد سلسلة الاتصال مباشرة للخاصية ConnectionString .

```
con.ConnectionString = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
```

ثانيا – الاستعلام والأمر command و الاتصال معا

يمكن أيضا تمرير الاستعلام والاتصال بأكثر من طريقة للمتحولات من نوع SqlCommand وتختلف الطرق ولكن كلها مثل بعضها تؤدي نفس النتيجة ولكن لو كان لدي أكثر من استعلام أريد تنفيذه بنفس الاتصال أو أكثر من اتصال يمكننا التبديل بينهم باستخدام الخصائص كما نرى في درسنا هذا ومن الطرق المستخدمة هنا ما يلي:

الطريقة الأولى:

وهي تخزين الاستعلام كالمعتاد في متحول نصي وانشاء اتصال ما ومن ثم اسناد الاتصال والاستعلام كوسيط دخل للأمر sqlcommand كما يلي

```
SqlConnection con = new SqlConnection(connection);
string SqlStatment = "select studID,studName,registerDate from students";
SqlCommand cmd = new SqlCommand(SqlStatment, con);
```

الطريقة الثانية :

وهي اسناد الاستعلام مباشرة للمتحول من نوع sqlcommand كما يلي:

```
SqlConnection con = new SqlConnection(connection);
SqlCommand cmd = new SqlCommand("select studID,studName,registerDate from students", con);
```

الطريقة الثالثة:

وهي استخدام خصائص المتحولات من نوع SqlCommand لاسناد قيم الخصائص.

```
string SqlStatment = "select studID,studName,registerDate from students";
SqlCommand cmd = new SqlCommand();
cmd.Connection = con;
cmd.CommandText = SqlStatment;
```

الطريقة الرابعة:

اسناد مباشر للاستعلام ضمن CommandText .

```
SqlCommand cmd = new SqlCommand();
cmd.Connection = con;
cmd.CommandText   =   "select   studID,studName,registerDate   from
students";
```

ثالثا – الاتصال Connection

إن الاتصال هام جدا ويمكن لنا أن نقوم باختباره للتأكد من حالة الاتصال والتحكم به حيث أن الاتصال con له عدة حالات وهي:

- Broken : الاتصال محطم أو مفقود وهي تحصل بعد فتح الاتصال ووقوع مشكلة ما في الاتصال فيصبح معطل وغير قادر على الوصول إلى قاعدة البيانات يتم حل مشكلة الاتصال بإغلاقه وإعادة فتحه من جديد.
- Closed : وهي الحالة الطبيعية بعد أن ننتهي من العمل مع قاعدة البيانات حيث أنه يتوجب علينا اغلاق الاتصال مع قاعدة البيانات كي لا يسبب عبئا على موارد السيرفر.
- Connecting : وهي حالة (جاري الاتصال مع قاعدة البيانات) حيث يكون فيها الاتصال قيد التوصيل.
- Executing : وهي حالة الاتصال (جاري تنفيذ الاستعلام) عندما يكون قيد تنفيذ استعلام معين.
- Fetching : وهي حالة الاتصال (جاري جلب البيانات) أي يتم الآن استيراد البيانات من قاعدة البيانات.
- Open : الاتصال مع قاعدة البيانات مفتوح.

يمكن الوصول لهذه الحالات بكتابة التعليمة التالية:

```
con.State= ConnectionState.***
```

مع استبدال النجم *** بإحدى الحالات السابقة والتي ستظهر لك في القائمة في الفيجوال استديو

دورة في ADO.NET - الدرس السابع (تعمق بالمحتوى)

الصف SqlCommand !!!!

إن هذا الصف يمكن أن نقوم بإنشاء مثيل منه (متحول) يعمل كأدارة لتنفيذ الاستعلام الخاص بنا سواء كان الاستعلام نصي أو عبارة عن إجرائية مخزنة (storde Procedure) .

يمكن الوصول إلى هذا الصف من خلال فضاءات الأسماء كما يلي :

```
System.Data.SqlClient.SqlCommand
```

ويمكن انشاء مثيل من هذا الصف كما يلي:

```
SqlCommand cmd = new SqlCommand();
```

أولا - التابع البناء Constructor !!!!

يملك المتحول من هذا الصف ثلاث توابع بناء Constructors حيث يمكن لنا أن نقوم بإنشاءه من دون تمرير وسطاء كما يلي:

```
SqlCommand cmd = new SqlCommand()
```

أو مع تمرير وسيط واحد وهو الاستعلام أو اسم الإجرائية المخزنة

```
SqlCommand cmd = new SqlCommand(String)
```

أو مع تمرير وسيطين وهما الأول الاستعلام أو اسم الإجرائية المخزنة والثاني هو الاتصال مع قاعدة البيانات

```
SqlCommand cmd = new SqlCommand(String, SqlConnection)
```

أو يمكننا تمرير ثلاث وسطاء له وهم الأول الاستعلام أو اسم الإجرائية المخزنة والثاني الاتصال مع قاعدة البيانات والثالث وهو المناقلة SqlTransaction .

```
SqlCommand cmd = new SqlCommand(String, SqlConnection, SqlTransaction)
```

ثانيا - أهم الخصائص لهذا الصف !!!!

- CommandText : يمكن من خلال هذه الخاصية أنقوم بتعيين أو الحصول على الاستعلام الذي سيتم تنفيذه أو اسم الإجرائية المخزنة التي سيتم الوصول إليها من خلال المتحول.

```
cmd.CommandText = "select studID,studName,registerDate from students";
```

- وعندما يكون الاستعلام هو عبارة عن إجرائية مخزنة يتم وضع اسن الإجرائية مكان الاستعلام :

```
cmd.CommandText = "MyProcedure";
```

- CommandTimeout : ويمكن من خلال هذه الخاصية تعيين أو الحصول على الوقت الذي سيتم الانتظار فيه حتى يتم انهاء التعليمة عند حدوث خطأ ما.

```
cmd.CommandTimeout = 5;
```

ويجب أن تكون قيمة TimeOut رقم صحيح int .

- CommandType : وهنا يتم تحديد نوع الاستعلام المنفذ هل هو استعلام نصي أو إجرائية مخزنة ففي حال كان الاستعلام نصي وهو الحالة الافتراضية لهذه المتحولات يكون كما يلي:

```
cmd.CommandType = CommandType.Text;
```

وفي حال كان الاستعلام هو عبارة عن إجرائية مخزنة يكون كما يلي:

```
cmd.CommandType = CommandType.StoredProcedure;
```

- Connection : وهنا يتم تحديد الاتصال الذي سيستخدمه المتحول حتى يقوم بتنفيذ مهامه.

```
cmd.Connection = con;
```

- Parameters : وفيه يتم تمرير البراميترات الخاصة بالاستعلام لدينا.

```
cmd.Parameters.Add(ParameterValue);
```

وسيكون هنالك ملحق لهذه التعليمة ضمن الدرس

- Transaction : وهنا يتم تحديد المناقلة التي سيتم استخدامها في الاستعلام.

```
SqlTransaction tran = new SqlTransaction();
cmd.Transaction = tran;
```

وسألحق شرح بسيط عن ما هي Transaction .

ملحق 1 :

ما هو البارامیتر وكيف يمكن إضافتة ؟؟؟

كما نعلم البارامیتر هو عبارة عن قيمة يتم تمريرها للاستعلام لتحقيق شرط معين مثلا :

```
Select studID, studName from Students where studID=@studID
```

ففي مثالنا السابق البارامیتر هو @studID ويجب تمرير قيمته للاستعلام وهي كما يلي:

```
cmd.Parameters.Add("@studID", SqlDbType.Int);
cmd.Parameters["@studID"].Value = StudentIDValue;
```

أو يمكن اسناد قيمة ذلك البارامیتر مباشرة كما يلي:

```
cmd.Parameters.Add("@studID", SqlDbType.Int).Value = studentIDValue;
```

ملحق 2 :

ماهي Transaction ؟؟؟؟

المناقلة وهي عبارة مجموعة من الاستعلامات يتم تنفيذها مع بعضها البعض فإما تنفذ جميعها معا وإلا لن يتم تنفيذ شيء وهنا مثال عليها:

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;

SqlConnection con = new SqlConnection(connection);
string SqlStatment1 = "insert statment1";
string SqlStatment2 = "insert statment2";
SqlTransaction tran = new SqlTransaction();
SqlCommand cmd = new SqlCommand("", con, tran);
try
{
     tran = con.BeginTransaction();
     con.Open();
     cmd.CommandText = SqlStatment1;
     cmd.ExecuteNonQuery();
     cmd.CommandText = SqlStatment2;
     cmd.ExecuteNonQuery();
     tran.Commit();
}
catch (Exception EX)
```

```
{
     tran.Rollback();
}
con.Close();
```

تم الإعلان عن بدأ tran بالتعليمة

```
tran = con.BeginTransaction();
```

حيث أن بدأ المناقلة مرتبط بالاتصال التي ستعمل عليه المناقلة وفي حال تم العمل بنجاح يتم تثبيت العمل بالتعليمة:

```
tran.Commit();
```

وإلا يتم التراجع من خلال التعليمة:

```
tran.Rollback();
```

ثالثا - أهم الطرق Method المستخدمة في SqlCommand !!!!

- Cancel : لإلغاء الـ Command الحالية.
- Clone : إنشاء نسخة من Command الحالية ومطابقة لها تماما.
- ExecuteNonQuery : لتنفيذ استعلام لا يعيد قيمة مثل استعلامات Update,Delete,Insert .
- ExecuteReader : لتنفيذ استعلام يعيد جدول كامل له عدة اسطر وعدة حقول.
- ExecuteScalar : لتنفيذ استعلام يعيد قيمة وحيدة أي خلية واحدة من جدول م في قاعدة البيانات.

دورة في ADO.NET - الدرس الثامن (تعمق بالمحتوى)

الصف sqlConnection !!!!

هذا الصف مسؤول عن انشاء متحولات تقوم بالاتصال بقاعدة البيانات ومن ثم تقوم بتنفيذ محتوى معين لاستعلام وهو ضروري عند الاتصال بأي قاعدة بيانات.

أولا - التابع البناء Constructor !!!!

يمكن تعريف هذا المتحول بثلاث طرق وهي :

- SqlConnection() : وفيها يمكن أن نقوم بتعريف الاتصال بدون أي وسيط ممر له.

```
SqlConnection con = new SqlConnection();
```

- SqlConnection(String) :وفيه يمكن تعريف متحول للاتصال مع تحديد سلسلة الاتصال الخاصة بهذا الاتصال.

```
SqlConnection con = new SqlConnection(connection);
```

- SqlConnection(String, SqlCredential) : وفيها يتم تعريف الاتصال مع تمرير سلسلة الاتصال وبطاقة تعريف SqlCredential وهي تتضمن اسم مستخدم وكلمة مرور لن نأتي على ذكرها الأن.

```
SqlConnection con = new SqlConnection(String, SqlCredential) ;
```

ثانيا – خصائص الصف :

من أهم الخصائص التي يتمتع بها هذا الصف هي :

- ConnectionString : وهي تقوم بتحديد سلسلة الاتصال الخاصة بالاتصال بقاعدة البيانات وذلك من خلال البيانات الموجودة في Web.confg .

```
con.ConnectionString = connection;
```

- State : وهي تقوم بتحديد الحالة للاتصال مفتوح ام مغلق.

```
con.State;
```

ثالثا – الطرق الخاصة بالصف :

يتمتع هذا الصف بمجموعة من الخصائص الهامة وهي :

- BeginTransaction : لبدأ مناقلة ما وقد قمنا بشرح المناقلات في الدرس السابق
- Close : إغلاق الاتصال مع قاعدة البيانات
- Open : فتح الاتصال مع قاعدة البيانات

دورة في ADO.NET - الدرس التاسع (تعمق بالمحتوى)

الصف SqlDataAdapter !!!!

يعمل هذا الصف مثل الصف sqlCommand تماما ولكن مجموعة من الفروقات حيث أنه يحتوي على عبارات select ,Update, Delete, insert ويمكن لنا إجراء مقارنة بين الصفين ببساطة كما يلي:

SqlCommand :

- يقوم بتنفيذ التعليمات الخاصة به من خلال تنفيذ الاستعلام أو الإجراية المخزنة على قاعدة البيانات مباشرة.
- يمكن أن يتم انشاءه لتنفيذ أي استعلام T-SQL سواء كان ادخال أو تعديل او حذف او جلب من خلال تمرير الاستعلام.
- ويستخدم مع SqlDataReader حيث لا يمكن استخدم DataAdapter معها أبدا.
- يجب أن نقوم بفتح وإغلاق الاتصال يدويا.

SqlDataAdapter :

- يتطلب الاتصال مع قاعدة البيانات والاستعلام لملء البيانات في DatSet أو DataTable حيث يجري عليها التعديلات ومن ثم يقوم بحفظها في قاعدة البيانات.
- يمكن أن يتم تمرير استعلام select اما مباشرة من خلال constructor التابع الباني أي تمريره كوسيط دخل أو من خلال متحول sqlcommand يحتوي على استعلام select لنقوم بإسناده للأدبتر الخاص بنا.
- ان SqlDataReader هو داخليا يتضمن استخدام DataAdapter .
- الاتصال مع قاعدة البيانات يتم فتحه وإغلاقه تلقائيا.

أولا - التابع البناء constructor :

يمكن تعريف مثيل من هذا الصف بأربع طرق وهي :

- SqlDataAdapter() :وفيه يمكن للمستخدم أن لا يقوم بتمرير أي وسيط للمتحول الجديد كما يلي :

```
SqlDataAdapter da = new SqlDataAdapter();
```

- SqlDataAdapter(SqlCommand) : وفيه يتم تمرير متحول من نوع SqlCommand يحتوي على الاستعلام الذي نرغب بتنفيذه كما يلي:

```
SqlCommand cmd = new SqlCommand(SqlStatment, con);
```

```
SqlDataAdapter da = new SqlDataAdapter(cmd);
```

- SqlDataAdapter(String, SqlConnection) :
- وفيه يتم تمرير استعلام Select على أساس أنه نص عادي وتمرير الاتصال الذي سيتم استخدمه.

```
SqlDataAdapter da = new SqlDataAdapter("select * from students",con);
```

- SqlDataAdapter(String, String) :
- وفيه يتم تمرير استعلام Select وسلسلة الاتصال بقاعدة البيانات مباشرة على أساس بيانات نصية:

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
SqlDataAdapter da = new SqlDataAdapter("select * from students",connection);
```

كما يمكن بهذه الحالة تمرير سلسلة الاتصال كما هي في web.config .

ثانيا – أهم خصائص الصف :

- DeleteCommand : ويحتوي على استعلام Delete الخاص بالمتحول :

```
da.DeleteCommand.CommandText = "Delete from students where studID=1";
```

- InsertCommand : ويحتوي على استعلام insert :

```
da.InsertCommand.CommandText = "insert into students (studName,regDate) values('New Name', 2/4/2014)";
```

- SelectCommand : ويحتوي على استعلام Select :

```
da.SelectCommand.CommandText = "select studID,studName,registerDate from students";
```

- UpdateCommand : ويحتوي على استعلام Update :

```
da.UpdateCommand.CommandText = "Update students set studName='New Name' where studID=1";
```

ثالثا – أهم التوابع

- Fill(DataSet) : حيث يتم تمرير DataSet له كوسيط دخل ليقوم بملئها ببيانات الاستعلام الذي قمنا بتمرير للـ DataAdapter .
- Fill(DataTable) : نمرر له DataTable ليقوم بملئه ببيانات الاستعلام الخاص بنا.
- Update(DataRow[]) : يقوم بحفظ التغيرات لاستعلامات Update, Delete, Insert التي تم اجراءها على مصفوفة DataSet التي قمن بجلب DataRow منها.
- Update(DataSet) : يتم فيها حفظ التغيرات لاستعلامات Update, Delete, Insert التي تم اجراءها على الـ DataSet .

رابعا – أهم أحداث الصف :

- FillError : يحث هذا الحدث عندما يحدث خطأ عند تنفيذ التابع Fill.
- RowUpdated : يحدق بعد أن يتم حفظ التغيرات على قاعدة البيانات.
- RowUpdating : يحدث قبل أن يتم الانتهاء من عملية الحفظ أي أثناء سير العملية.

يمكن أن نقوم بكتابة كود معين ضمن هذه الأحداث لكي يتم تنفيذها لأداء مهمة معينة.

```
adapter.RowUpdating += new
SqlRowUpdatingEventHandler( OnRowUpdating );
```

حيث سيتم تنفيذ التابع OnRowUpdating والذي يحتوي على مانريد تنفيذه كما يلي مثلا:

```
private static void OnRowUpdating(object sender,
SqlRowUpdatingEventArgs e)
{
    PrintEventArgs(e);
}
```

دورة في ADO.NET - الدرس العاشر (تعمق بالمحتوى)

الصف SqlDataReader !!!!

يقوم هذا الصف بجلب البيانات من قاعدة البيانات ويقوم بتخزينها لديه على هيئة جداول يمكن لنا الوصول إليها والتحكم بها كما نرغب.

وهو باتجاه واحد أي للقراءة فقط لا يمكن له العودة بتغيرات معينة تم اجراءها عليه أو تثبيت هذه التغيرات على قاعدة البيانات.

أولا – التابع البناء constractor !!!!

يمكن انشاء مثيل من هذا الصف من خلال الطريقة التقليدية حيث أنه ل يأخذ أي وسيط دخل له كما يلي:

```
SqlDataReader dr = new SqlDataReader();
```

وكما يمكن لنا أن نقوم بإنساد sqlcommand مربوطا بتابع ExecuteReader كما يلي:

```
SqlDataReader dr = cmd.ExecuteReader();
```

ثانيا – أهم خصائص الصف !!!

- Depth : يعطينا قيمة العقدة الحالية للحقل الحالي الذي نعمل به ضمن الداتاريدر

```
int x = dr.Depth;
```

- FieldCount : يعيد عدد الأعمدة في السطر الحالي :

```
int x = dr.FieldCount;
```

- HasRows : يعيد قيمة فيما إذا كان الريدر يحوي على أسطر أم أنه فارغ كقيم بوليانية صح أو خطأ:

```
bool x = dr.HasRows;
```

- IsClosed : يعيد قيمة بوليانية فيما إذا كان الريدر الحالي مغلق أم لا :

```
bool x = dr.IsClosed;
```

- Item[Int32] : حيث نضع رقم العنصر ضمن الريدي ليتم طباعته كما يلي:

```
object val = dr[1];
```

- Item[String] : مثل سابقتها ولكن مع ذكر اسم الحقل كما هو موجود في الاستعلام وقاعدة البيانات:

```
object val = dr["studID"];
```

ثالثا – أهم الطرق المستخدمة بهذا الصف !!!!

- Close : يقوم بإغلاق الريدر ويصبح غير قابل للقراءة.
- NextResult : يقوم بتحريك المؤشر للنتيجة التالية ضمن الريدر لقراءته أثناء تنفيذ استعلام T-SQL.
- Read : يقوم بالانتقال للسطر التالي لقراءة البيانات منه ضمن الريدر الحالي.

مثال عملي :

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
SqlConnection con = new SqlConnection(connection);
string SqlStatment = "select studID,studName,registerDate from students";
SqlCommand cmd = new SqlCommand(SqlStatment, con);
con.Open();
SqlDataReader dr = cmd.ExecuteReader();
```

دورة في ADO.NET - الدرس الحادي عشر (تعمق بالمحتوى)

الصف DataSet !!!!
تعتبر DataSet كمخزن للبيانات في الذاكرة وتعمل مثل عمل DataReader ولكن بفارق أنها تعمل باتجاهين أي قراءة وكتابة من قاعدة البيانات وليس قراءة فقط حيث يمكن لـ DataSet أن تقوم بتطبيق التغيرات على قاعدة البيانات بعد أن تم اعدادها وكما يمكن لها أن تكون مخزن مؤقت في الذاكرة بشكل منفصل عن قاعدة البيانات.

يمكن بناء متحول من الصف DataSet بالريقة التقليدية وهي :

```
DataSet dataSet = new DataSet();
```

من أهم ما يميز هذا الكائن هو وجود الجداول Tables فيه وهي احدى الخصائص الموجودة في الكائن فكما سبق وذكرنا هو قاعدة بيانات في الذاكرة.

لفهم dataset سنقوم بإعداد كائن في الذاكرة من خلال خلق DataSet وإضافة جداول Tables عليها ونتعلم كيفية استدعاء كل جدول منها :
أولا نقوم بتعريف كائن dataTable في الذاكرة كما يلي:

```
DataTable table = new DataTable("studentsInfo");
```

من ثم سنقوم بتعريف عودين في هذا الجدول هما على الشكل التالي:

```
DataColumn col1 = new DataColumn("id", typeof(int));
DataColumn col2 = new DataColumn("name", typeof(string));
```

الأول هو id وهو من النوع int والثاني هو name من النوع string وسنقوم بإلحاق هذان العمودان بالجدول table السابق كما يلي:

```
table.Columns.Add(col1);
table.Columns.Add(col2);
```

الخطوة التالية بعد أن قمنا بتجهيز الأعمدة سنقوم بإضافة أسطر فيها بياناتنا لهذه الأعمدة من خلال إضافة DataRow وتزويدها بالبيانات الخاصة بها كما يلي:

```
DataRow dr1 = table.NewRow();
dr1["id"] = 1;
dr1["name"] = "tammam";
table.Rows.Add(dr1);

DataRow dr2 = table.NewRow();
dr2["id"] = 2;
dr2["name"] = "yasser";
table.Rows.Add(dr2);
```

```
DataRow dr3 = table.NewRow();
dr3["id"] = 3;
dr3["name"] = "lamis";
table.Rows.Add(dr3);
```

الأن أصبح الجدول table فيه عمودان اثنان وفيهما ثلاث أسطر بقي أن نقوم بإلحاق هذا الجدول ضمن DataSet التي سنقوم بإنشاءها وسنقوم بإلحاق الجدول بها كما يلي:

```
DataSet ds = new DataSet();
ds.Tables.Add(table);
```

وللتحقق من صحة بياناتنا سنقوم بوضع GridView ضمن صفحة ونجعل الـ DataSource خاصتها هي الـ DataSet التي لدينا وسنرى النتيجة ان شاء الله كما أردنا كما يلي:

```
GridView2.DataSource = ds;
GridView2.DataBind();
```

id	name
1	tammam
2	yasser
3	lamis

ولجلب البيانات من قاعدة البيانات إلى DataSet وفق استعلام معين إليكم المثال التالي:

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
SqlConnection con = new SqlConnection(connection);
string SqlStatment = "select studID,studName,registerDate from students";
SqlDataAdapter da = new SqlDataAdapter(SqlStatment, con);
```

```
DataSet ds = new DataSet();
da.Fill(ds);
GridView2.DataSource = ds;
GridView2.DataBind();
```

كل الجديد فيما سبق هو استخدام DataAdapter لملء DataSet بالبيانات من خلال التابع Fill.

وإليكم الكود بلغة VB.Net :

```
Dim connection As String = ConfigurationManager.Connection
Strings("ConnectionString").ConnectionString
Dim con As SqlConnection = New SqlConnection(connection)
Dim SqlStatment As String = "select studID,studName,regist
erDate from students"
Dim da As SqlDataAdapter = New SqlDataAdapter(SqlStatment,
con)
Dim ds As DataSet = New DataSet
da.Fill(ds)
GridView2.DataSource = ds
GridView2.DataBind
```

دورة في ADO.NET - الدرس الثاني عشر (الأخير)

أخوتي اليوم في الدرس الأخير سأقوم بكتابة مثال يقوم بعملية الإضافة (بطريقتين: استعلام TSQL و Stored Procedure)

عملية الإضافة لقاعدة البيانات !!!!
حتى نقوم بالإضافة لقاعدة البيانات علينا أولا تحديد استعلام يقوم بعملية الإضافة وتجهيز اتصال مع قاعدة البيانات.

الطريقة الأولى T-SQL :
أولا نقوم بتعريف سلسلة الاتصال لقاعدة البيانات ثم الاتصال مع قاعدة البيانات كما يلي:

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
SqlConnection con = new SqlConnection(connection);
```

ثم نقوم بتحديد الاستعلام المراد أن يتم تنفيذه و هو استعلام الإدخال لقاعدة البيانات ويأخذ القيم Values من مربع نصي وتاريخ التسجيل يكون هو لحظة الضغط على الزر ومعرف الطالب تلقائي في قاعدة البيانات كما يلي:

```
string SqlStatment = "insert into students (studName,registerDate) values ('"+txtName.Text+"',GETDATE())";
```

نقوم بتعريف أداة تنفيذ الاستعلام ونمرر لها الاستعلام والاتصال ونقوم بعملية فتح الاتصال كما يلي:

```
SqlCommand cmd = new SqlCommand(SqlStatment, con);
con.Open();
```

نقوم بتنفيذ الاستعلام من خلال التابع ExecuteNonQuery الذي لايعيد قيمة انما يعيد عدد الأسطر التي تم اجراء تغيرات عليها وهي بحالتنا ادخال سطر واحد فقط ونضع شرط للتحقق فيما اذا كان الاستعلام نجح بإدخال السطر أم لا كما يلي:

```
int x = cmd.ExecuteNonQuery();
if (x > 0)
{
    lblMsg.Text = "Student Added!!!!";
}
else
{
    lblMsg.Text = "Faild to add student!!!!";
}
con.Close();
```

وإليكم الكود بلغة VB.NET:

```
Dim connection As String = ConfigurationManager.ConnectionStrings("ConnectionString").ConnectionString
Dim con As SqlConnection = New SqlConnection(connection)
Dim SqlStatment As String = ("insert into students(studName,registerDate) values('" + (txtName.Text + "',GETDATE())")
Dim cmd As SqlCommand = New SqlCommand(SqlStatment, con)
con.Open
Dim x As Integer = cmd.ExecuteNonQuery
If (x > 0) Then
    lblMsg.Text = "Student Added!!!!"
Else
    lblMsg.Text = "Faild to add student!!!!"
End If
con.Close
```

الطريقة الثانية من خلال إجرائية Stored Procedure!!!!
ليكن لدينا الإجرائية المخزنة التالية:

```
ALTER PROCEDURE dbo.AddNewStudent
    @Name nvarchar(50)
AS
    insert into students(studName,registerDate)
values(@Name,GETDATE())
    RETURN
```

كيف سنقوم باستدعائها من خلال كود السي شارب؟؟؟؟
الطريقة هي ذاتها السابقة مع تغيير سلسلة الاستعلام وتحويلها إلى إجرائية مخزنة كما يلي:

```
string connection = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
SqlConnection con = new SqlConnection(connection);
```

هنا بدأ أول تعديل وهي حذف الاستعلام النص واستدالة باسم الإجرائية المخزنة مررة للمتحول cmd كما يلي:

```
SqlCommand cmd = new SqlCommand("AddNewStudent", con);
```

نقوم للمتحول cmd إن نوع الاستعلام الذي ستقوم بتنفيذه هو إجرائية مخزنة وهذا الكلام ضروري جدا فالحالة الافتراضية هي استعلام نص بالطريقة الأولى كما يلي:

```
cmd.CommandType = CommandType.StoredProcedure;
```

ضمن الإجرائية المخزنة باراميتر @Name وهو الاسم المراد إدخاله سنقوم بتمريره بطريقتين الأولى من خلال تعريف متحول من نوع SqlParameter ومن ثم نقوم بتمرير اسم الباراميتر وقيمته كما يلي:

```
SqlParameter Para = new SqlParameter("@Name", txtName.Text);
```

ونقوم بإلحاق البارامتر بالمتحول cmd ليقوم بإرساله للإجرائية المخزنة كما يلي:

```
cmd.Parameters.Add(Para);
```

أو الطريقة الثانية وهي الإضافة المباشرة للبارامتر مع تحديد اسم البارامتر ونوع بياناته وقيمته أخيرا كما يلي:

```
cmd.Parameters.Add("@Name",      SqlDbType.NVarChar).Value      =
txtName.Text;
```

ولك حرية اختيار احدى الطريقتين وأفضل الثانية لأنها مختصرة أكثر عند وجود عدة براميترات وثم نتابع سير البرنامج بشكل طبيعي كما يلي:

```
con.Open();
int x = cmd.ExecuteNonQuery();
if (x > 0)
{
    lblMsg.Text = "Student Added!!!!";
}
else
{
    lblMsg.Text = "Faild to add student!!!!";
}
```

وإليكم الكود مكتوبا بلغة VB.NET :

```
Dim connection As String = ConfigurationManager.Connection
Strings("ConnectionString").ConnectionString
Dim con As SqlConnection = New SqlConnection(connection)
Dim cmd As SqlCommand = New SqlCommand("AddNewStudent", co
n)
cmd.CommandType        =        CommandType.StoredProcedure
Dim Para As SqlParameter = New SqlParameter("@Name", txtNa
me.Text)
cmd.Parameters.Add(Para)
con.Open
Dim     x     As     Integer     =     cmd.ExecuteNonQuery
If           (x           >           0)           Then
               lblMsg.Text    =    "Student    Added!!!!"
Else
          lblMsg.Text   =   "Faild   to   add   student!!!!"
End                                                      If
con.Close
```

والحمد لله رب العالمين