HUTECH
Đại học Công nghệ Tp.HCM

KHOA CAO ĐẲNG THỰC HÀNH

THIẾT KẾ & LẬP TRÌNH WEBSITE
(Chuyên ngành: Đồ Họa Đa Truyền Thông)

Chương 6
LẬP TRÌNH WEB FORM VỚI ADO.NET

NỘI DUNG

1 Tổng quan về ADO.Net

2 Các đối tượng trong ADO.Net

3 Xây dựng lớp xử lý dữ liệu

4 Xử lý giỏ hàng cho website thương mại điện tử

1. TỔNG QUAN VỀ ADO.NET

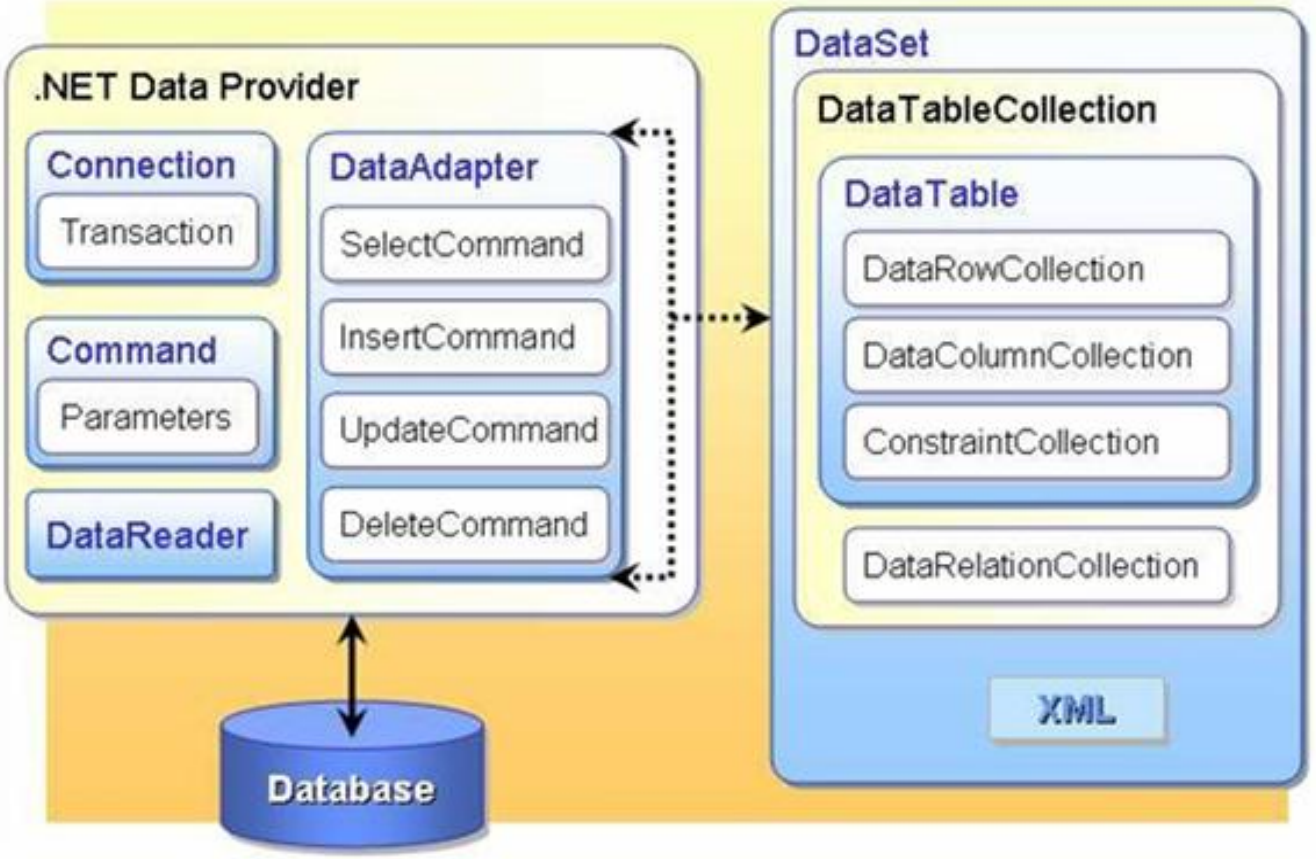
1.1. ADO.NET là gì ?

- ✓ ActiveX Data Object .NET (ADO.NET)- Mô hình truy xuất CSDL trên nền .NET
- ✓ Do Microsoft Soft phát triển từ nền tảng ADO
- ✓ Cung cấp các lớp đối tượng và hàm thư viện phục vụ cho việc kết nối và xử lý dữ liệu
- ✓ Tăng tốc truy xuất dữ liệu theo mô hình đa lớp: tách biệt truy cập dữ liệu với thao tác dữ liệu.
- ✓ Cho phép truy xuất dữ liệu ở chế độ connected và disconnected.
- ✓ Hỗ trợ thao tác với XML.

1. TỔNG QUAN VỀ ADO.NET

1.2. Kiến trúc của ADO.NET:

ADO.Net Gồm 2 thành phần chính: **.Net Data Provider** và **DataSet**.

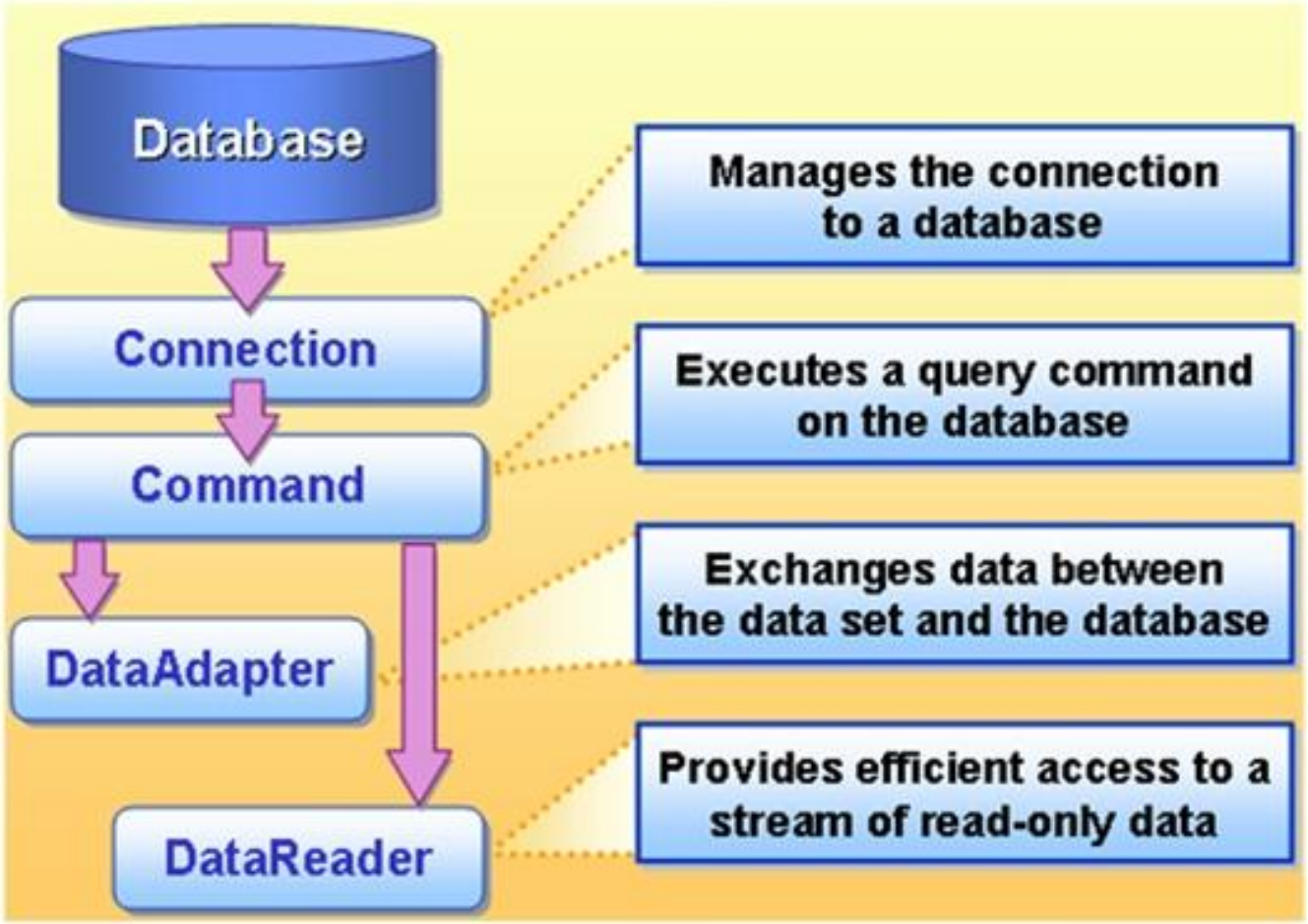


1. TỔNG QUAN VỀ ADO.NET

.Net Data Provider: gồm 4 thành phần:

- ✓ Connection: Thực hiện thiết lập và duy trì kết nối đến CSDL.
- ✓ Command: Lưu trữ các lệnh truy vấn hay stored procedure.
- ✓ DataReader: Lưu trữ kết quả thực thi lệnh truy vấn từ CSDL.
- ✓ DataAdapter: Là cầu nối giúp trao đổi dữ liệu giữa DataSet và CSDL.

1. TỔNG QUAN VỀ ADO.NET

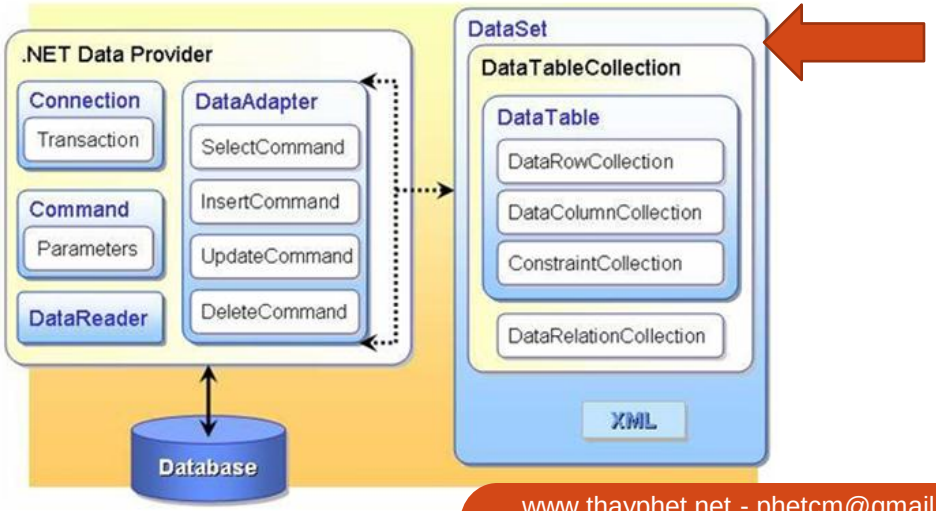


Kiến trúc của .Net Data Provider

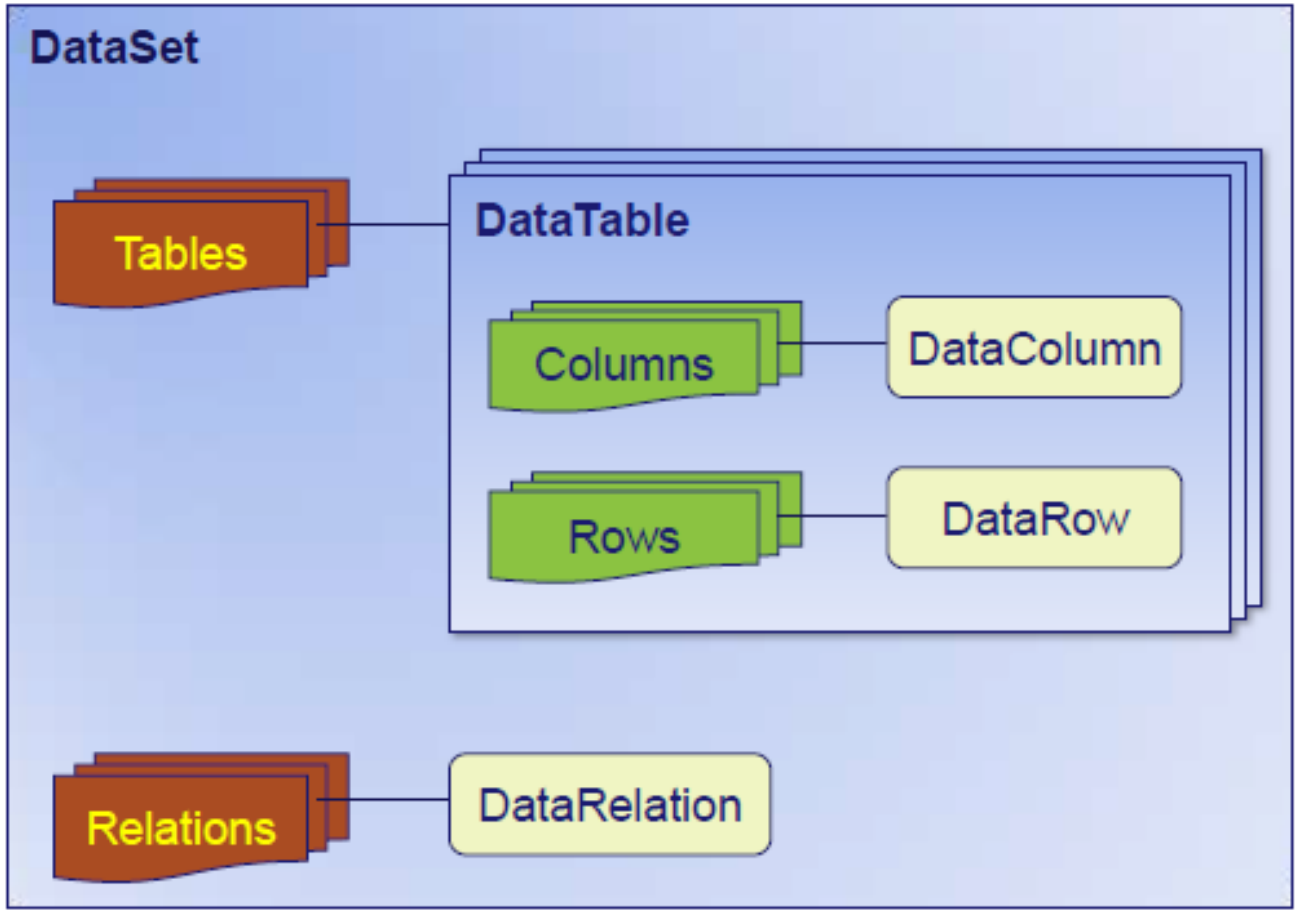
1. TỔNG QUAN VỀ ADO.NET

DataSet: Lưu trữ các bảng dữ liệu, lược đồ CSDL.

- ✓ Thực thi cơ chế ngắt kết nối (disconnected) nhằm tăng hiệu năng truy xuất CSDL.
- ✓ Mọi thao tác thay đổi dữ liệu được thực hiện trên DataSet, không ảnh hưởng đến CSDL.
- ✓ Thay đổi dữ liệu, cập nhật CSDL thông qua đối tượng DataAdapter.



1. TỔNG QUAN VỀ ADO.NET



Kiến trúc của Dataset

1.3. Đặc điểm của ADO.NET

- ✓ Cho phép lấy cả một cấu trúc phức tạp của dữ liệu từ CSDL, sau đó ngắt kết nối rồi mới thực hiện thao tác xử lý.
- ✓ ADO.NET mạnh mẽ: Kế thừa các ưu điểm của ADO. Kết hợp với ý tưởng thiết kế hoàn toàn mới
- ✓ Thiết kế hoàn toàn hướng đối tượng: Đặc trưng của thư viện .NET Framework

2. CÁC ĐỐI TƯỢNG TRONG ADO.NET

2.1 Đối tượng Connection

- ✓ Để tương tác với database thì phải có một kết nối.
- ✓ Kết nối cần xác định:
  - Server name
  - Database name
  - User name
  - Password

2.2. Đối tượng Command

- ✓ Dùng đối tượng command gửi một câu lệnh SQL tới database để thực hiện hành động tương tác với Database
- ✓ Một đối tượng command dùng một đối tượng connection để xác định database.
- ✓ Có thể dùng một đối tượng command riêng lẻ để thực thi lệnh trực tiếp, hoặc gắn cho một SqlDataAdapter

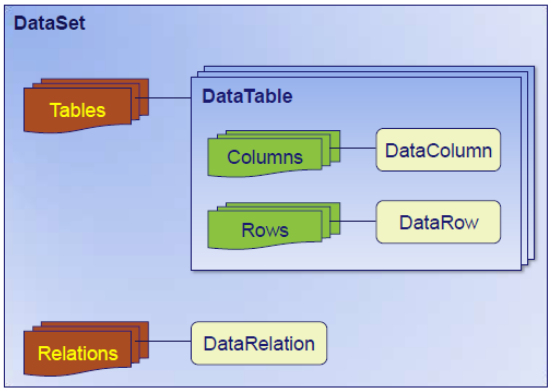
2.3. Đối tượng Datareader

- ✓ Nhiều thao tác dữ liệu chỉ lấy một luồng dữ liệu để đọc. Đối tượng Data Reader cho phép lấy được kết quả của câu lệnh SELECT từ đối tượng command.
- ✓ Để tăng hiệu suất, dữ liệu trả về từ một Data Reader là một luồng dữ liệu fast forward-only có lợi về tốc độ.
- ✓ Tuy nhiên nếu phải thao tác dữ liệu, thì một DataSet sẽ là một đối tượng tốt hơn để làm việc

2. CÁC ĐỐI TƯỢNG TRONG ADO.NET

2.4. Dataset

- Là một thể hiện của dữ liệu trong bộ nhớ, chứa nhiều DataTable, như các database thông thường.
- Có thể định nghĩa dữ liệu giữa các table để tạo các quan hệ.
- Được thiết kế đặc biệt để giúp quản lý dữ liệu không cần kết nối (disconnected) trên dữ liệu.
- Nhờ đối tượng DataAdapter làm trung gian



2. CÁC ĐỐI TƯỢNG TRONG ADO.NET

2.5. Data adapter

- Data Adapter cho phép quản lý dữ liệu trong chế độ ngắt kết nối. Khi cần làm việc ở chế độ read-only, cần lưu trữ tạm dữ liệu trong bộ nhớ để hạn chế truy xuất đến Database.
- Data Adapter sẽ đổ vào DataSet khi đọc dữ liệu và thực hiện thay đổi dữ liệu một lượt vào database.
- Data Adapter chứa một tham chiếu đến đối tượng connection và mở/đóng kết nối tự động khi đọc và ghi dữ liệu vào database.
- Data adapter chứa đối tượng command cho những thao tác Select, Insert, Update và Delete trên dữ liệu.

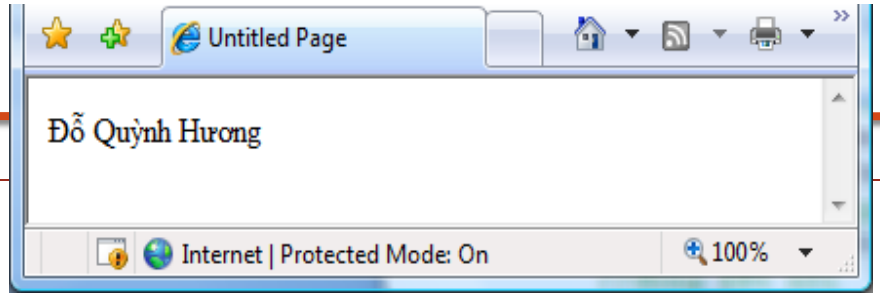
2. CÁC ĐỐI TƯỢNG TRONG ADO.NET

2.6. Minh họa tạo kết nối CSDL

Cơ bản các bước thực hiện với database

- ✓Bước 1: Tạo kết nối
- ✓Bước 2: Mở kết nối dữ liệu
- ✓Bước 3: Tạo lệnh điều khiển truy vấn SQL
- ✓Bước 4: Thực thi lệnh
- ✓Bước 5: Đóng kết nối
- ✓Bước 6: in kết quả

Ví dụ:




```
using System;
using System.Data;
using System.Data.SqlClient;
public partial class vd1 : System.Web.UI.Page{
    protected void Page_Load(object sender, EventArgs e)  {
     //Khai báo và khởi tạo biến Connection
       SqlConnection cnn = new SqlConnection("Data Source=(local);
                                Initial Catalog=QLbansach;User ID=sa;Password=");
      cnn.Open(); //Mở kết nối
     //Command điều khiển truy vấn sql
       SqlCommand cmd = cnn.CreateCommand();
       cmd.CommandText="select TenKH from Khachhang where MaKH=5";
     //lấy về chuỗi giá trị trong cơ sở dữ liệu
       string result = (string)cmd.ExecuteScalar();
     cnn.Close(); //đóng kết nối
     Response.Write(result); //in giá trị ra màn hình
    }
}
```

3. ĐỐI TƯỢNG CONNECTION

Microsoft ADO .Net

Vai trò của Connection trong ADO.net là tạo kết nối giữa ứng dụng với CSDL

Data Provider

- ✓System.Data.Oledb : Sử dụng với Access
- ✓System.Data.SqlCient : Sử dụng với SQLServer

Ứng với mỗi tên miền:

- ✓System.Data.Oledb.OledbConnection
- ✓System.Data.SqlClient.SqlConnection

Và các Data Provider khác:

- ✓System.data.OcracleClient(Ocracle)
- ✓MicroSoft.data.Odbc(Thông qua ODBC của HĐH)
- ✓Microsoft.Data.Sqlxml (XML trên Sqlserver)

3. ĐỐI TƯỢNG CONNECTION

Microsoft ADO .Net

Nếu kết nối với CSDL SQLServer

Provider: Khai báo Data Provider của SQLServer

Data Source/Server: Tên Server

Initial Catalog/DataBase: Tên CSDL

User ID/UID: Tên người dùng

Password/ PWD: Mật khẩu

Integrated Security: Cơ chế chứng thực đăng nhập

true: tài khoản Windows;

false: Tài khoản SqlServer (ví dụ: sa)

3. ĐỐI TƯỢNG CONNECTION

Microsoft ADO .Net

Ví dụ: Tạo kết nối với CSDL SQLServer

```
using System;
using System.Data;
using System.Data.sqlClient;
public partial class KetnoiCSDL : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        String    StrCnn="Data Source=.; Initial Catalog=QLbansach;
                                                    User ID=sa;Password=“;
        SqlConnection cnn = new SqlConnection(StrCnn);
         cnn.Open();
        //Truy xuất, xử lý dữ liệu
         cnn.Close();
    }
}
```

3. ĐỐI TƯỢNG CONNECTION

Microsoft ADO .Net

Các thuộc tính Của Connection

DataBase: Tên CSDL

DataSource: Tên Server

Provider:Tương ứng với Provider của HQTCSDL

State: Tình trạng kết nối của Connection:

- Broken: Kết nối đã bị ngắt khi đã kết nối
- Closed: Kết nối đã đóng
- Connecting: Đang kết nối
- Executing: Kết nối đang thực hiện một lệnh
- Fetching: Kết nối đang truy xuất dữ liệu
- Open: Kết nối đang mở

3. ĐỐI TƯỢNG CONNECTION

Microsoft ADO .Net

Các phương thức

Change Databse: Thay đổi DataBase làm việc

Close : Đóng kết nối

Dispose: Giải phóng bộ nhớ

Open: Thực hiện kết nối

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

Sau khi tạo kết nối CSDL, mọi thao tác với nguồn dữ liệu có thể được thực hiện thông qua Command.

Tùy theo loại Connection đối tượng Command thuộc tên miền:

System.Data.OleDb.OleDbCommand

System.Data.SqlClient.SqlCommand

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

Tạo Command

Cú pháp:

<Loai Command> <Biến Command> As New <Loai Command>;

<Biến Command>.Connection=<Biến Connection >;

<Biến Command>.CommandText=Lệnh SQL>;

Hoặc

<Loai Command> <Biến Command>

As New <Loại Command>(<Lệnh SQL>);

<Biến Command>.Connection=<Biến Connection >;

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

Các thuộc tính Của Command

CommandText: Lệnh SQL hay tên Procedure

CommandType: Loại Command

Text: (Mặc định): Là câu lệnh SQL

StoredProcedure: Tên thủ tục

TableDirect: Tên Connectionủa table

VD:

```
SqlCommand cmd As SqlCommand = New SqlCommand();
cmd.Connection = cnn;
cmd.CommandType = CommandType.Text;
cmd.CommandText = "Select * From Khachhang"
```

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

Parameters

- Lệnh SQL trong commandText có thể sử dụng
  - ? (khi sử dụng Access)
  - @Tênbiến (khi sử dụng SQLServer)

  thay cho trị chưa xác định và khi thực hiện sẽ dùng đối tượng Parameters để truyền giá trị vào dấu ?/ @Tênbiến.
- Tùy theo Command Parameter sẽ khai báo khác nhau

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

SQLServer

SqlParameter <tên Parameter>
As New SqlParameter();

SqlParameter <Ten Parameter>
As New SqlParameter(<Tên>);

SqlParameter <Tên parameter>
As New SqlParamter(<tên>,<giá>);

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

Các thuộc tính cần chú ý:

Direction: Giá trị cho biết lọai tham số

- Input: (mặc định) Loại tham số đầu vào
- InputOutput: Loại tham số đầu vào và ra
- Output: Loại tham số đầu ra
- ReturnValue: Loại tham số nhận trị trả về

OleDbType / SqlDbType: Kiểu dữ liệu của tham số.

ParameterName: Tên tham số

Value: Giá trị tham số

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

VD: Khi sử dụng SqlDbCommand

```
cmd.CommandText=”Select * From KhachHang
                                    Where MaKH=@MaKH”;
SqlParameter Par As
                          SqlParameter = cmd.CreateParameter();
Par.ParameterName=”@MaKH”;
Par.Value=”KH01”;
cmd.Parameters.Add(Par);
```

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

VD: Khi sử dụng SqlDbCommand

```
cmd.CommandText=”Select * From BangDiem
                Where Masv=@MaSV and MaMH = @MaMH ”;
SqlDbParameter Par1 As SqlDbParameter=
    cmd.CreateParameters.Add(“@MaSV”,SqlType.Char,4);
Par1.Value=”SV01”
SqlDbParameter Par2 As SqlDbParameter=
cmd.CreateParameters.Add(“@MaMH”,SqlType.Char,4);
Par2.Value=”MH01”;
```

4. ĐỐI TƯỢNG COMMAND

Tạo tham số và đưa vào tập hợp Parameters

VD: Procedure SpKetQuaThi cần 2 tham số đầu vào: @MaSV , @MaMH và trả về Điểm thi của Môn học của sinh viên đó.

4. ĐỐI TƯỢNG COMMAND

Microsoft ADO .Net

Thực hiện Command:

❖ Phương thức ExecuteReader: Trả về đối tượng DataReader để đọc dữ liệu mỗi lần một dòng với phương thức Read.(DataReader đọc dữ liệu trực tiếp từ nguồn nên phải duy trì kết nối đến khi đọc xong)

SqlDataReader <Tên DataReader> As SqlDataReader;
<Tên DataReader> = <tên Command>.ExecuteReader;

VD: SqlDataReader rd As SqlDataReader;
rd = cmd.ExecuteReader;

❖ Phương thức ExcuteNoneQuery: Dùng thực thi các phát biểu T-Sql như: Insert, Update, Delete, Create,…

❖ Phương thức ExcuteScalar: Trả về từ phát biểu SQL dạng Select chỉ có một cột một hàng.

Ví dụ 1: Sử dụng Command với câu lệnh Select

Microsoft ADO .Net

```
try
    {
        SqlConnection cnn = new SqlConnection("Data Source=(local);
                            Initial Catalog=QLbansach;User ID=sa;Password=");
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = cnn;
        //Loại command là câu lệnh SQL
        cmd.CommandText = "Select Count(*) From Chude";
        cmd.CommandType = CommandType.Text;
        //Mở kết nối và lấy dữ liệu
        cnn.Open();
        int count = (int)cmd.ExecuteScalar();
        response.write(count.ToString());
        cnn.Close();
    }
    catch (Exception)
    {
        response.write("Không thành công!“);
    }
```

Ví dụ 2: Sử dụng Command với lệnh Insert,Update,Delete

```
try
    {
        SqlConnection cnn = new SqlConnection("Data Source=(local);
                                Initial Catalog=QLbansach;User ID=sa;Password=");
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = cnn;
        //Biến Commnad thao tác Insert, Update, Delete
        cmd.CommandText = "Insert Into Chude(tencd) Values(n'văn hóa')";
        cmd.CommandType = CommandType.Text;
        cnn.Open();
        cmd.ExecuteNonQuery();
        response.write("Thành công!“);
        cnn.Close();
    }
    catch (Exception)
    {
        response.write(“Thất bại!“);
    }
```

Ví dụ 3: Command với lệnh Insert,Update,Delete + Tham số

```
try
        {
             SqlConnection cnn = new SqlConnection("Data Source=(local);
                                        Initial Catalog=QLbansach;User ID=sa;Password=");
            SqlCommand cmd = new SqlCommand();
             cmd.Connection = cnn;
             SqlParameter parTenLinhVuc =
                        new SqlParameter("@TENCHUDE", SqlDbType.NVarChar, 50);
             cmd.CommandText =
                                        "INSERT INTO CHUDE VALUES(@TENCHUDE)";
             cmd.CommandType = CommandType.Text;
             cmd.Parameters.Add(parTenchude);
             parTenLinhVuc.Value = TextBox1.Text;
            cnn.Open();
            cmd.ExecuteNonQuery();
            cnn.Close();
            response.write("Thành công!“);
        }
        catch (Exception)
        {
            response.write(“Thất bại!“);
        }
```

5. ĐỐI TƯỢNG DATAREADER

Microsoft ADO .Net

❖Là đối tượng truy cập dữ liệu trực tiếp, sử dụng con trỏ phía Server và duy trì kết nối với Server trong suốt quá trình đọc dữ liệu,

❖Tùy theo loại Connection mà DataReader thuộc tên miền:

System.Data.OleDb.OleDbDataReader

System.Data.SqlClient.SqlDataReader

5. ĐỐI TƯỢNG DATAREADER

Microsoft ADO .Net

Các thuộc tính

- FieldCout: Số cột trên dòng hiện hành của DataReader
- IsClosed : Cho biết dataReader đã đóng
- Item:Trị của cột truyền vào. Tham số truyền vào là tên cột hoặc số thứ tự tính từ 0.

5. ĐỐI TƯỢNG DATAREADER

Microsoft ADO .Net

Các phương thức

- Close: Đóng DataReader
- GetFieldType: Trả về kiểu dữ liệu của tham số truyền vào.
- GetName: Trả về tên của cột truyền vào
- GetValue: Trả về trị của cột truyền vào
- Read: Di chuyển đến dòng kế tiếp và trả về true nếu còn dòng để di chuyển, ngược lại trả về False.

Trong khi dataReader đang mở các thao tác dữ liệu trên nguồn dữ liệu đều không thể cho đến khi dataReader đóng lại bằng lệnh Close

Ví dụ 1:DataReader với lệnh Insert,Update,Delete +Tham số

```
try
   {        SqlConnection cnn = new SqlConnection("Data Source=(local);
                                Initial Catalog=QLbansach;User ID=sa;Password=");
            SqlCommand cmd = new SqlCommand();
            cmd.Connection = cnn;
            cmd.CommandText = "SELECT * FROM Nhaxuatban";
            cmd.CommandType = CommandType.Text;
            cnn.Open();
            IDataReader dr = cmd.ExecuteReader();
            String list = "";
            while (dr.Read())
            {
                        list = list + dr["TenNXB"].ToString().Trim() + " ";
            }
            dr.Close();
            response.write(list.ToString());
            cnn.Close();
     }
catch (Exception)
       {
            response.write(“Thất bại!“);
       }
```

Ví dụ 2: DataReader + gọi procedure (VD: Getnhaxuatban)

```
try
{       SqlConnection cnn = new SqlConnection("Data Source=(local);
                        Initial Catalog=QLbansach;User ID=sa;Password=");
        SqlCommand cmd = new SqlCommand("Getnhaxuatban", cnn);
        cmd.Connection = cnn;
        cmd.CommandType = CommandType.StoredProcedure;
        cnn.Open();
        IDataReader dr = cmd.ExecuteReader();
        String list = "";
        while (dr.Read())
        {
                list = list + dr["TenNXB"].ToString();
        }
        dr.Close();
        response.write(list.ToString());
        cnn.Close();
}
catch (Exception)
    {
        response.write("Thất bại!");
    }
```

Ví dụ 2: DataReader + gọi procedure có tham số

```
Create Procedure GetchudeByMaCD
@Machude char(15)
AS
Begin
        Select * From Chude Where MaCD=@Machude
End
```

```
try
{           SqlConnection cnn = new SqlConnection("Data Source=(local);
                                    Initial Ctalog=QLbansach;User ID=sa;Password=");
            SqlCommand cmd = new SqlCommand("GetchudeByMaCD", cnn);
            cmd.Connection = cnn;
            cmd.CommandType = CommandType.StoredProcedure;
            SqlParameter parMALINHVUC =
                              new SqlParameter("@Machude", SqlDbType.NChar, 10);
            parMAVHUDE.Value = TextBox1.Text;
            cmd.Parameters.Add(parMACHUDE);
            cnn.Open();
            IDataReader dr = cmd.ExecuteReader();
            String list = "";
            while (dr.Read())
            {
                        list = list + dr["Tenchude"].ToString();
            }
            dr.Close();
            response.write(list.ToString());
            cnn.Close();
}
catch (Exception)
      {
           response.write(“Thất bại!“);
      }
```

6. ĐỐI TƯỢNG DATAADAPTER

Microsoft ADO .Net

❖ Để lấy dữ liệu từ nguồn dữ liệu về cho ứng dụng, chúng ta sử dụng đối tượng DataAdapter. Đối tượng này cho phép ta lấy cấu trúc và dữ liệu của các bảng.

❖ DataAdapter là một bộ gồm 4 đối tượng:

- SelectCommand: Cho phép lấy thông tin từ nguồn.
- InsertCommand: Cho phép thêm dữ liệu vào bảng trong nguồn.
- UpdateCommand: Cho phép điều chỉnh dữ liệu của bảng trong nguồn.
- DeleteCommand: Cho phép xóa dữ liệu của bảng trong nguồn.

6. ĐỐI TƯỢNG DATAADAPTER

Microsoft ADO .Net

❖ Tạo DataAdapter

Cú pháp:

<Loai>DataAdapter <Biến DataAdapter> =
    New <Loai>DataAdapter(<Lệnh>,<Biến Connection>)

DataAdapter chỉ thao tác với nguồn dữ liệu qua đối tượng connection đang kết nối, khi Connection chưa mở thì DataAdapter sẽ tự động mở kết nối khi cần và đóng lại

6. ĐỐI TƯỢNG DATAADAPTER

Microsoft ADO .Net

❖ Các thuộc tính của DataAdapter

- DeleteCommand: Đối tượng Command chứa nội dung lệnh hủy các mẫu tin trên nguồn dữ liệu.
- InsertCommand: Đối tượng Command chứa nội dung lệnh thêm các mẫu tin trên nguồn dữ liệu.
- SelectCommand: Đối tượng Command chứa nội dung lệnh truy xuất các mẫu tin trên nguồn dữ liệu.
- UpdateCommand: Đối tượng Command chứa nội dung lệnh sửa các mẫu tin trên nguồn dữ liệu.

6. ĐỐI TƯỢNG DATAADAPTER

Microsoft ADO .Net

❖ Các chức năng của DataAdapter

- Lấy dữ liệu từ nguồn:

  - DataTable: Fill(<DataTable>)

  - DataSet: Fill(<DataSet>)

  Dữ liệu lấy về DataSet dưới dạng các dataTable với tên là: Table0,Table1, Table2. . .

  - Đổ dữ liệu vào Datset cho bảng DataTable nếu chưa có sẽ tạo mới:

    Fill(<DataSet>,<Tên dataTable>)

6. ĐỐI TƯỢNG DATAADAPTER

- Phương thức trả về mẫu tin lấy được

  Dataset DS as New Dataset()

  Integer so;

  so= DA.Fill(DS,”Sinhvien”)

- Để cập nhật dữ liệu về nguồn

  Update(<mảng dòng>): Cập nhật các dòng (Các đối tượng DataRow) vào nguồn dữ liệu.

  Update(<Dataset>): Cập nhật các thay đổi trên tất cả các bảng của Dataset vào nguồn dữ liệu.

  Update(<DataTable>): Cập nhật tất cả các thay đổi trên DataTable vào nguồn dữ liệu.

  Update(<Dataset>,<Tên bảng>) Cập nhật các thay đổi trên bảng trong Dataset vào nguồn.

7. ĐỐI TƯỢNG DATASET

Microsoft ADO .Net

❖ Dataset là một mô hình CSDL quan hệ thu nhỏ đáp ứng nhu cầu của ứng dụng.

❖ Dataset chứa các bảng (DataTable), các quanhệ (DataRelation) và các ràng buộc (constraint)

❖ Dataset thuộc tên miền: System.Data.Dataset.

❖ Khai báo

New System.Data.Dataset()

Hoặc

New System.Data.Dataset(<tên Dataset>)

7. ĐỐI TƯỢNG DATASET

Microsoft ADO .Net

❖ Các phương thức

Thêm một bảng vào Dataset

Tables.Add()

Một bảng mới tự động được tạo ra với tên mặc định Table1, Table2 . . .

Tables.Addd(<Tên bảng>)

Một bảng mới tạo ra theo đúng <tên bảng>

Ghi chú: Tên bảng có phân biệt chữ in, thường

Xóa bảng ra khỏi Dataset

Tables.Remove(<Tên bảng>)

Xóa bảng ra khỏi tập hợp Table.

7. ĐỐI TƯỢNG DATASET

Microsoft ADO .Net

Kiểm tra bảng có thuộc về Dataset

Tables.Contains(<Tên bảng>)

Lấy chỉ số của bảng

Tables.IndexOf(<tên bảng>)

Lấy số bảng trong Dataset

Tables.Count

Lấy ra một bảng trong Dataset

Tables(<Chỉ số>)

Để cập nhật các thay đổi trên Dataset

AcceptChanges()

7. ĐỐI TƯỢNG DATASET

Microsoft ADO .Net

Để hủy các thay đổi trên Dataset

RejectChanges()

Để xóa bỏ mọi dữ liệu trên dataSet

Clear()

Để tạo một bản sau của Dataset

Clone()

Để xóa bỏ Dataset

Dispone()

Giải phóng mọi tài nguyên trên vùng nhớ Dataset đang sử dụng.

Tạo quan hệ giữa hai bảng trong Dataset.

Relations.Add(<DataColumn trên bảng cha>, <Data Column trên bảng con>)

Xóa quan hệ giữa hai bảng trong Dataset.

Relations.Remove(<quan hệ>)

Dữ liệu các bảng trong nguồn dữ liệu được lấy về và đưa vào các DataTable. DataTable thuộc tên miền : System.Data.dataTable.

Cú pháp:

New DataTable();

New DataTable(<Tên bảng>);

DataTable được hình thành từ DataColumn và DataRow.

Ví dụ 1: DataAdapter + update dữ liệu

```
try
 {          SqlConnection cnn = new SqlConnection("Data Source=(local);
                                    Initial Catalog=QLbansach;User ID=sa;Password=");
            SqlDataAdapter da=new SqlDataAdapter("select * from CHUDE", cnn);
            SqlCommandBuilder commandBuilder = new SqlCommandBuilder(da);
            DataSet ds = new DataSet();
            da.Fill(ds);
            foreach (DataRow row in ds.Tables[0].Rows)
            if (row["MaCD"]=="1")
            {
                        row["TENCHUDE"] = "BBB";
            }
            response.write(ds.Tables[0].Rows[2].ItemArray[1].ToString());
            GridView1.DataSource = ds.Tables[0];
            GridView1.DataBind();
            //Không sử dụng SqlCommandBuilder thì không thể update dữ liệu.
            da.Update(ds);
}
 catch (Exception)
       {
            response.write("Thất bại!");
       }
```

Ví dụ 2: DataAdapter + Procedure(GetNXB)

```
try
{
        SqlConnection cnn = new SqlConnection("Data Source=(local);
                        Initial Catalog=QLbansach;User ID=sa;Password=");
        SqlDataAdapter da = new SqlDataAdapter("GETNXB", cnn);
        DataSet ds = new DataSet();
        da.Fill(ds);
        response.write(ds.Tables[0].Rows[2].ItemArray[1].ToString());
        GridView2.DataSource = ds.Tables[0];
        GridView2.DataBind();}
catch (Exception)
    {
        response.write("Thất bại!");
    }
```

Ví dụ 3: DataAdapter + Procedure tham số (Getchude)

```
try
{           SqlConnection cnn = new SqlConnection("Data Source=(local);
                                Initial Catalog=QLbansach;User ID=sa;Password=");
            SqlCommand cmd = new SqlCommand("Getchude", cnn);
            cmd.Connection = cnn;
            cmd.CommandType = CommandType.StoredProcedure;
            SqlParameter parMACD =
                              new SqlParameter("@MACD", SqlDbType.NChar, 10);
            parMACD.Value = "1";
            cmd.Parameters.Add(parMACD);
            cnn.Open();
            SqlDataAdapter da = new SqlDataAdapter();
            da.SelectCommand = cmd;
            DataSet ds = new DataSet();
            da.Fill(ds);
            GridView1.DataSource = ds.Tables[0];
            GridView1.DataBind();
            cnn.Close();
}
 catch (Exception)
       {
            response.write(“Thất bại!“);
       }
```

Ví dụ 4: DataAdapter + Đối số là command

```
try
{  SqlConnection cnn = new SqlConnection("Data Source=(local);
    Initial Catalog=QLbansach;User ID=sa;Password=");
    SqlCommand cmd = new SqlCommand("GETNXB", cnn);
    cmd.Connection = cnn;
    cmd.CommandType = CommandType.StoredProcedure;
    cnn.Open();
    SqlDataAdapter da = new SqlDataAdapter();
    da.SelectCommand = cmd;
    //da.InsertCommand = cmd;
    //da.DeleteCommand = cmd;
    //da.UpdateCommand = cmd;
    DataSet ds = new DataSet();
    da.Fill(ds);
    GridView1.DataSource = ds.Tables[0];
    GridView1.DataBind();
    cnn.Close();
}
catch (Exception)
        {
            response.write(“Thất bại!“);
        }
```

8. XÂY DỰNG LỚP XỬ LÝ DỮ LIỆU

Microsoft ADO .Net

❖ Để các thao tác với CSDL thuận lợi. Ta nên xây dựng lớp xử lý dữ liệu đảm nhận việc kết nối CSDL và các thủ tục xử lý.

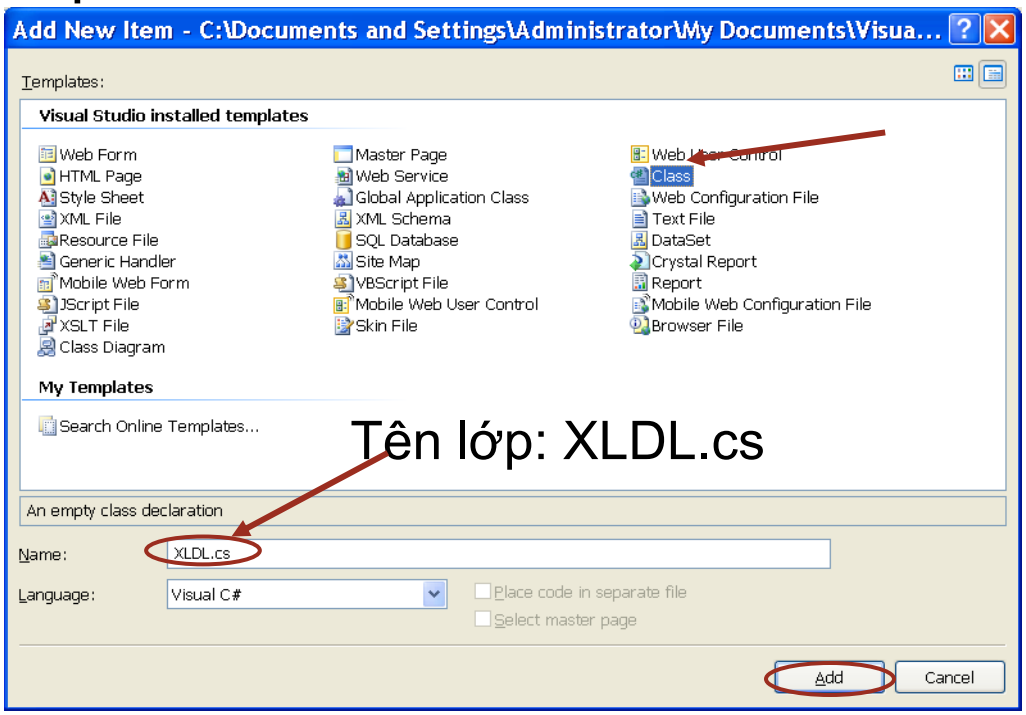
- Docbang(string LenhSQL): Nhằm thực hiện câu lệnh truy vấn SQL để trả về dữ liệu là 1 DataTable
- Thuchienlenh(string LenhSQL): Nhằm thực hiện câu lệnh Insert, Update, Delete để cập nhật dữ liệu cho CSDL.

❖ Thực hiện:

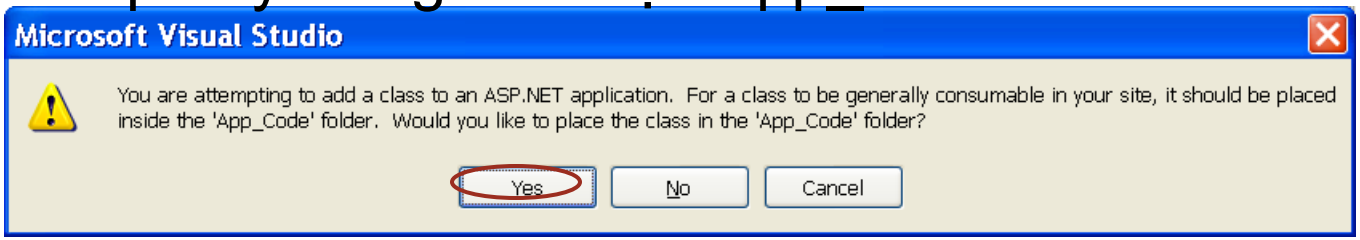
- Tạo cấu hình chuỗi kết nối CSDL trong tập tin Webconfig. (Có thể dùng SQLDatasource để sinh mã)

```
<add name="Ketnoi" connectionString="Data Source =(local);
    Initial Catalog=BooksOnline; UID=sa, PWD=Pass123; Integrated
                                                Security=False“
                                providerName="System.Data.SqlClient" />
```

- Tạo mới lớp XLDL.cs: Thêm mới 1 Item



Sẽ lưu lớp này trong thư mục App_Code



Thực hiện mã code cho lớp XLDL.cs

```
. . .
using System.Data.SqlClient;
public class clsXLDL
{
        static string StrCnn = ConfigurationManager.ConnectionStrings["Ketnoi"].
                                                        ConnectionString.ToString();
    public static DataTable getData(string LenhSQL)
    {
        using (SqlConnection cnn = new SqlConnection(StrCnn))
        {
            SqlDataAdapter bodocghi = new SqlDataAdapter(LenhSQL, cnn);
            DataTable bang = new DataTable();
            bodocghi.Fill(bang);
            return bang;
        }
    }
. . .
```

```
. . .
public static void Execute(string LenhSQL)
    {
        using (SqlConnection cnn = new SqlConnection(StrCnn))
        {
                cnn.Open();
                SqlCommand bolenh = new SqlCommand(LenhSQL, cnn);
                bolenh.ExecuteNonQuery();
                cnn.Close();
         }
    }
}
```

HUTECH
Đại học Công nghệ Tp.HCM

KHOA CAO ĐẲNG THỰC HÀNH

Chương 6
LẬP TRÌNH WEB FORM VỚI ADO.NET

THE END.

www.thayphet.net - phetcm@gmail.com