


LẬP TRÌNH CSDL VỚI C#

PHAN TRỌNG TIẾN
BM Công nghệ phần mềm
Khoa Công nghệ thông tin, VNUA
Email: phantien84@gmail.com
Website: <http://timoday.edu.vn>

7/5/16 Lập trình CSDL với C# 1



Giới thiệu chung

- ❑ ADO.NET là một cải tiến ADO (Active Data Objects), được sử dụng để tương tác với CSDL hoặc các nguồn dữ liệu hỗ trợ bởi .NET
- ❑ ADO.NET cung cấp rất nhiều component cho tạo các ứng dụng phân tán, chia sẻ dữ liệu
- ❑ ADO.NET truy xuất đồng nhất tới các nguồn dữ liệu khác nhau như SQL Server và XML, và các nguồn dữ liệu qua OLE DB và ODBC

7/5/16 Lập trình CSDL với C# 2



Nội dung chính

1. Tổng quan
2. Tổng quan ADO .Net
3. .Net Data Provider
4. Demo: Lấy dữ liệu dùng ADO .Net
5. Đối tượng DataSet
6. Thiết kế và gắn kết dữ liệu
7. Tích hợp XML
8. Lab: Tạo ứng dụng ADO.Net



Mục đích của chương

- Chương này cung cấp sinh viên kiến thức cần thiết để tạo các ứng dụng mức cao truy cập dữ liệu dùng C#
- Sau bài này sinh viên có thể:
 - Liệt kê các lợi ích ADO .Net
 - Tạo các ứng dụng dùng ADO .Net
 - Liệt kê các thành phần chính của đối tượng ADO.Net và các chức năng của nó.
 - Dùng VS.Net để thiết kế và gắn kết dữ liệu
 - Giải thích cách tích hợp XML cùng ADO.Net



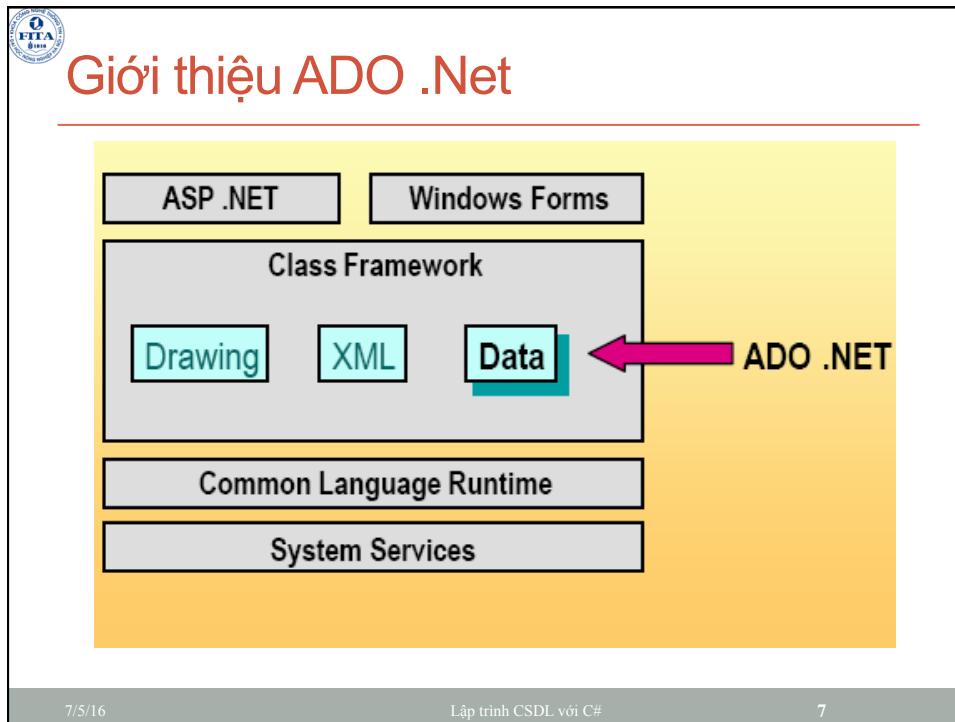
1. Tổng quan

- Bạn sẽ học cách sử dụng ADO .Net
- Cách sử dụng đối tượng **DataSet**
- Bạn sẽ học cách thiết kế dữ liệu trong VS.Net và cách gắn dữ liệu lên WinForm và WebForm.
- Cuối cùng bạn sẽ học cách tích hợp XML vào ADO .Net



2. Tổng quan ADO .Net

- Giới thiệu ADO .Net
- Các lợi ích ADO .Net



7/5/16

Lập trình CSDL với C#

7

Giới thiệu ADO.Net

- ❑ ADO.Net là một tập các lớp cho phép các ứng dụng .Net đọc và cập nhật thông tin DB và các nơi lưu trữ dữ liệu khác. Namespace: **System.Data**
- ❑ ADO.Net cung cấp đồng nhất cách truy cập các nguồn dữ liệu khác nhau như SQL Server, OLE DB, các nguồn dữ liệu không quan hệ như MS Exchange, và các tài liệu XML
- ❑ ADO.Net cải tiến khả năng disconnect tới dữ liệu.

7/5/16

Lập trình CSDL với C#

8



Giới thiệu ADO.Net

- ❑ ADO.Net cung cấp bốn .Net Data Providers:
 - ❑ .Net Data Provider cho SQL Server
 - ❑ .Net Data Provider cho OLE DB
 - ❑ .Net Data Provider cho ODBC
 - ❑ .Net Data Provider cho Oracle
- ❑ ADO.Net cung cấp nhiều công cụ cho việc đọc, cập nhật, thêm mới và xóa dữ liệu. Nhiều đối tượng trong thư viện là tương tự nhau và được nhận diện qua tên tiền tố của chúng ví dụ như **SqlDataReader** và **OleDbDataReader** cả hai đều cung cấp đọc các bản ghi từ nguồn dữ liệu.



Các lợi ích của ADO.Net

- ❑ Tương tự như ADO
- ❑ Được thiết kế cho dữ liệu không kết nối.
- ❑ Nằm trong nội tại .Net Framework nên dễ dàng trong việc sử dụng ngôn ngữ để phát triển.
- ❑ Hỗ trợ XML
 - ❑ ADO và XML có trước nhưng không tương thích
 - ❑ ADO dựa trên cơ sở dữ liệu quan hệ.
 - ❑ XML dựa trên cơ sở dữ liệu phân cấp.
 - ❑ ADO.Net kết hợp hai công nghệ này lại với nhau.



3. .Net Data Provider

- ❑ .Net Data Provider cho phép truy cập các nguồn dữ liệu xác định:
 - ❑ **System.Data.SqlClient** dùng truy cập SQL Server 7.0 trở lên
 - ❑ **System.Data.OleDb** dùng truy cập bất kỳ nguồn dữ liệu nào hỗ trợ OLE DB
 - ❑ **System.Data.Odbc**
 - ❑ **System.Data.OracleClient**
- ❑ Dùng đối tượng **Connection**
 - ❑ Kết nối tới một Database
- ❑ Dùng đối tượng **Command**
 - ❑ Thực thi các câu lệnh và các tùy chọn và trả về dữ liệu từ
 - ❑ Dùng đối tượng **Command** với các Stored Procedure
- ❑ Dùng đối tượng **DataReader**
 - ❑ đối tượng để tạo một luồng dữ liệu chỉ đọc
- ❑ Dùng đối tượng **DataAdapter**
 - ❑ đối tượng để trao đổi dữ liệu giữa nguồn và một **DataSet**

7/5/16

Lập trình CSDL với C#

11



Đối tượng **Connection**

- ❑ Dùng để kết nguồn dữ liệu, chuỗi kết nối được thiết lập qua thuộc tính **ConnectionString**
- ❑ **SqlConnection**

```
SqlConnection conSQL = new SqlConnection();
conSQL.ConnectionString = "Server = localhost; DataSource = Northwind; Uid = sa;
Pwd = admin123;";
conSQL.Open();
```
- ❑ **OleDbConnection**

```
OleDbConnection conAccess = new OleDb.OleDbConnection();
conAccess.ConnectionString = "Provider=
Microsoft.Jet.OLEDB.4.0;Data Source=c:\NWind.MDB";
conAccess.Open();
```

7/5/16

Lập trình CSDL với C#

12



Đối tượng *Command*

- Được dùng để thực hiện các câu truy vấn (query) và tùy chọn có thể trả về kết quả
- Có thể dùng với các stored query và procedure chấp nhận các tham số truyền vào

Tên	Mô tả
CommandText	Thuộc tính chỉ định câu lệnh SQL hoặc tên Stored procedure
CommandType	Thuộc tính xác định kiểu câu lệnh SQL
Connection	Thuộc tính cung cấp vào đối tượng Connection
CreateParameter	Phương thức để cung cấp các Parameter của câu lệnh SQL

7/5/16

Lập trình CSDL với C#

13



Đối tượng *Command*

- Có hai cách để tạo đối tượng *Command*:
 - Sử dụng Constructor *Command*
 - Sử dụng phương thức *CreateCommand*
- Có ba cách để thực thi một *Command*:
 - ExecuteReader
 - ExecuteScalar
 - ExecuteNonQuery
 - ExecuteXMLReader

```
SqlCommand commSQL = new SqlCommand();
commSQL.Connection = conSQL;
commSQL.CommandText = "Select Count(*) from Products";
MessageBox.Show(commSQL.ExecuteScalar().ToString);
```

7/5/16

Lập trình CSDL với C#

14



Đối tượng **Command** với các **Stored Procedure**

- ❑ Tạo một đối tượng **Command**
- ❑ Thiết lập **CommandType** là **StoredProcedure**
- ❑ Dùng phương thức **Add** để tạo và thiết lập các biến (Parameter)
- ❑ Dùng thuộc tính **ParameterDirection** để thiết lập kiểu biến
- ❑ Gọi phương thức **ExecuteReader**
- ❑ Dùng đối tượng **DataReader** để hiển thị hoặc duyệt qua các bản ghi và đóng khi kết thúc
- ❑ Truy cập đầu ra và trả về các biến

7/5/16

Lập trình CSDL với C#

15



Demo: Đối tượng **Command**

- ❑ Bạn thảo Stored Procedures: [CustOrdersDetail] trong CSDL NorthWind


```
ALTER PROCEDURE [dbo].[CustOrdersDetail] @OrderID int
AS
SELECT ProductName,
       UnitPrice=ROUND(Od.UnitPrice, 2),
       Quantity,
       Discount=CONVERT(int, Discount * 100),
       ExtendedPrice=ROUND(CONVERT(money, Quantity * (1 - Discount) * Od.UnitPrice), 2)
FROM Products P, [Order Details] Od
WHERE Od.ProductID = P.ProductID and Od.OrderID = @OrderID
```
- ❑ Code C#:


```
private void btnFind_Click(object sender, EventArgs e)
{
    SqlConnection conn = new SqlConnection("Server=TIENPT\
    \SQL;Database=NorthWind;Uid=sa;Pwd=admin123");
    if (conn.State != ConnectionState.Open)
    {
        conn.Open();
    }
}
```

7/5/16

Lập trình CSDL với C#

16



Demo: Đối tượng Command

```

SqlCommand cmd = new SqlCommand();
cmd.Connection = conn;
cmd.CommandType = CommandType.StoredProcedure;
cmd.CommandText = "CustOrdersDetail";
SqlParameter orderID = new SqlParameter("@OrderID",
SqlDbType.Int);
orderID.Value = 10250;
cmd.Parameters.Add(orderID);
SqlDataReader dr = cmd.ExecuteReader();
while (dr.Read())
{
    MessageBox.Show(dr[0].ToString());
}
dr.Close();
}

```

7/5/16

Lập trình CSDL với C#

17



Đối tượng DataReader

- Đọc dữ liệu

```

SqlCommand cmd = new SqlCommand();
cmd.Connection = conn;
cmd.CommandText = "SELECT [CustomerID],[CompanyName],
[ContactName],[ContactTitle],[Address] FROM [Customers]";
SqlDataReader dr = cmd.ExecuteReader();
while (dr.Read())
{
    MessageBox.Show(dr.GetString(0) + " - " + dr.GetString(1) + " -
"+ dr.GetString(2));
}
dr.Close();

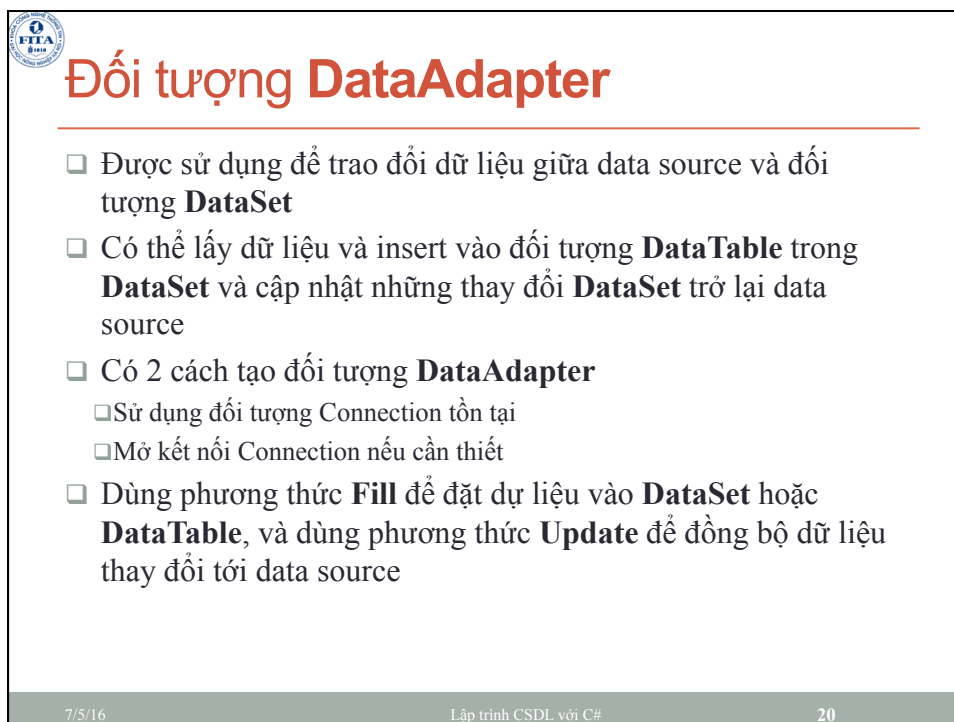
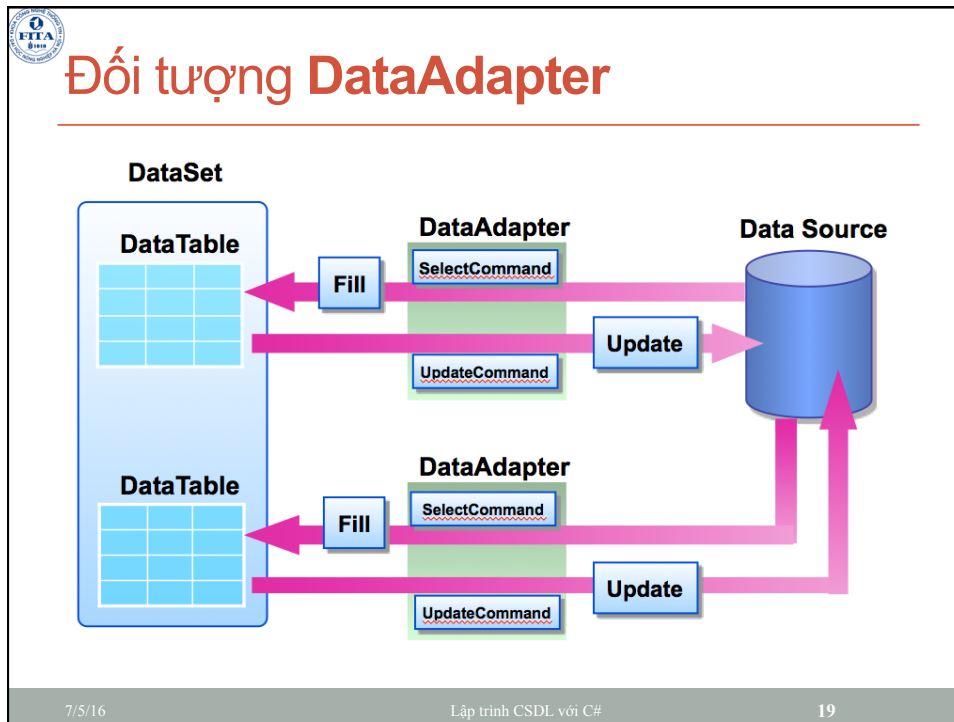
```

- Lấy thông tin
- Trả về nhiều tập hợp kết quả

7/5/16

Lập trình CSDL với C#

18





Sử dụng đối tượng **Connection** tồn tại

- ❑ Tạo đối tượng **Command** cùng với **Connection**, và khởi gán thuộc tính **SelectCommand** của đối tượng **DataAdapter**

```
private void frmProducts_Load(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection();
    con.ConnectionString = "Server=TIENPT\\SQL;
    Database=Northwind;Uid = sa; Pwd = admin123";
    con.Open();
    SqlCommand cmd = new SqlCommand("Select * from
    Products", con);
    SqlDataAdapter adp = new SqlDataAdapter();
    adp.SelectCommand = cmd;
    DataSet ds = new DataSet();
    adp.Fill(ds);
    dgvProducts.DataSource = ds.Tables[0]; //DataGridView
    dgvProducts.Refresh();
}
```

7/5/16

Lập trình CSDL với C#

21



Mở kết nối **Connection** nếu cần thiết

- ❑ Khởi tạo đối tượng **DataAdapter** đặt vào các tham số là chuỗi query và đối tượng **Connection**
- ❑ Nó sẽ tự kiểm tra **Connection**, sẽ tự động mở và đóng khi hoàn thành

```
private void frmProducts_Load(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection();
    con.ConnectionString = "Server=TIENPT\
    \SQL;Database=Northwind;Uid = sa; Pwd = admin123";
    SqlDataAdapter adp = new SqlDataAdapter("Select * from
    Products", con);
    DataSet ds = new DataSet();
    adp.Fill(ds);
    dgvProducts.DataSource = ds.Tables[0]; //DataGridView
    dgvProducts.Refresh();
}
```

7/5/16

Lập trình CSDL với C#

22



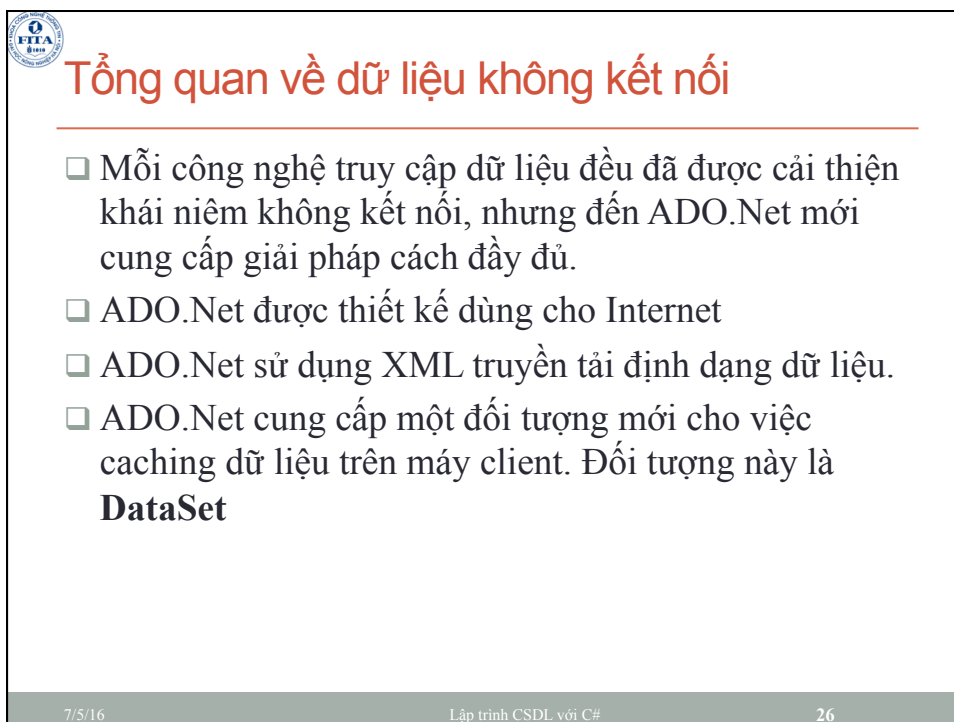
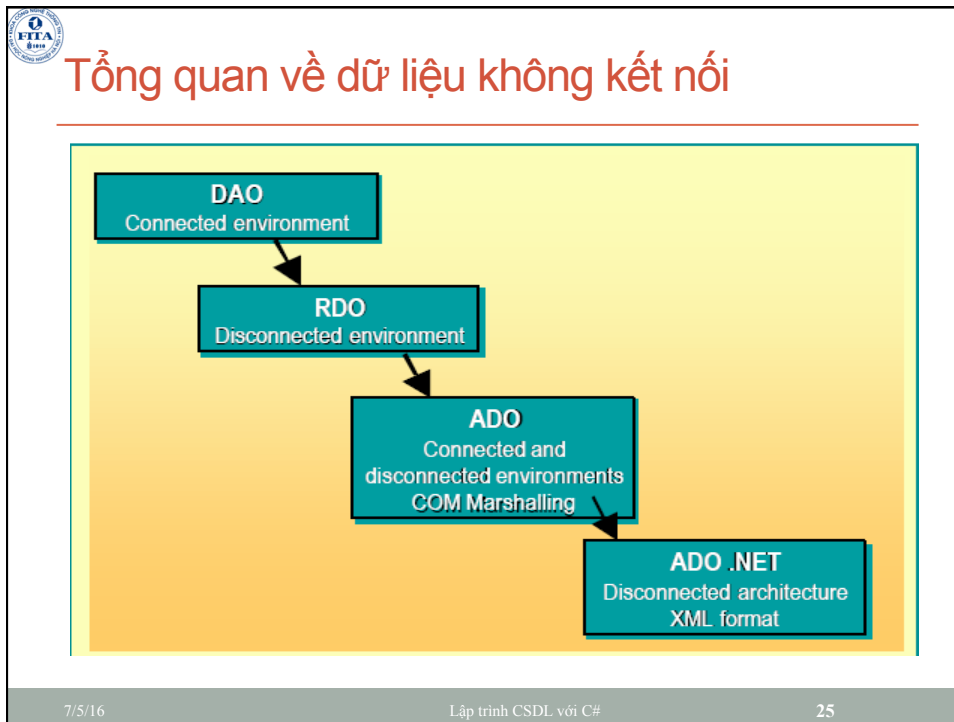
4. Demo: Lấy dữ liệu dùng ADO .Net

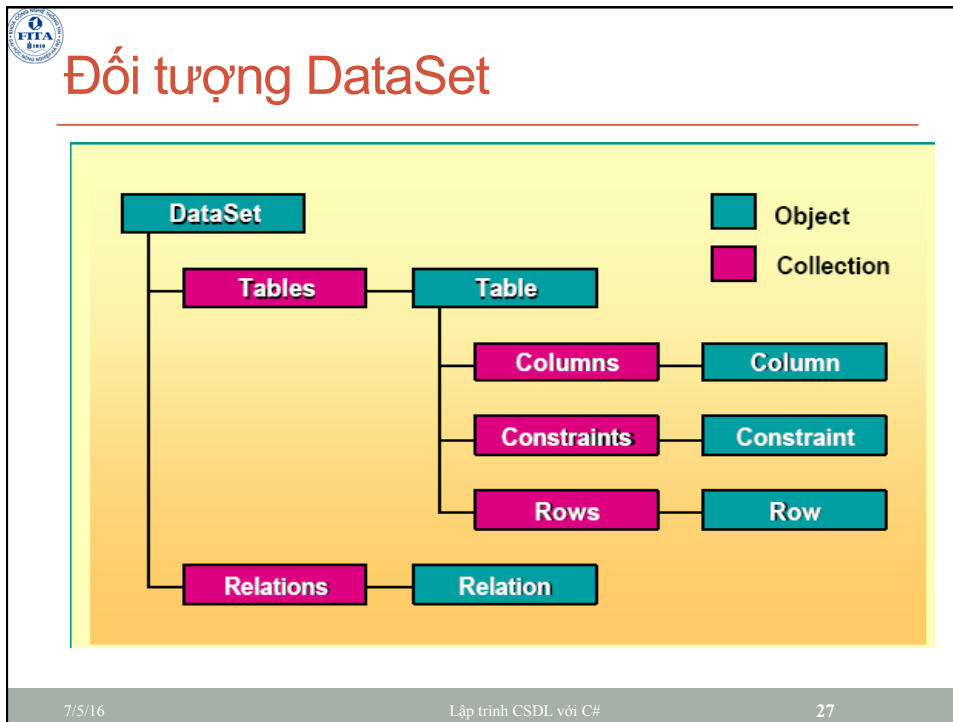
- Bạn sẽ học cách lấy dữ liệu từ CSDL SQL Server bằng việc dùng đối tượng **SQLDataAdapter** trong .NET.




5. Đối tượng DataSet

- Tổng quan về dữ liệu không kết nối
- Đối tượng DataSet
- Cư trú dữ liệu trong DataSet
- Sử dụng Relationship trong DataSet
- Sử dụng các Constraint
- Cập nhật dữ liệu trong DataSet
- Cập nhật dữ liệu tại nguồn





 **Cự trữ dữ liệu trong DataSet**

- ❑ Cự trữ dữ liệu trong DataSet từ một RDBMS


```

SqlDataAdapter adp = new SqlDataAdapter("Select * from Products",
con);
DataSet ds = new DataSet();
adp.Fill(ds, "Products");
      
```
- ❑ Lập trình tạo Dataset


```

DataSet dsPubs = new DataSet();
DataTable dtAuthor = new DataTable("Author");
dtAuthor.Columns.Add("AuthorID", Type.GetType("System.Int32"));
dsPubs.Tables.Add(dtAuthor);
dgvProducts.DataSource = dsPubs.Tables[0]; //DataGridView
dgvProducts.Refresh();
      
```

7/5/16 Lập trình CSDL với C# 28



Sử dụng Relationship trong DataSet

- ❑ Tạo Relationship

```
DataRelation relPubsTitle = new
DataRelation("PubsTitles",dsPubs.Tables["Publishers"].Columns[
"pub_id"],dsPubs.Tables["Titles"].Columns["pub_id"]);
dsPubs.Relations.Add(relPubsTitle);
```

- ❑ Truy cập dữ liệu quan hệ

```
//Lay ban ghi dau tien trong bang Publishers
DataRow pubRow = dsPubs.Tables["Publishers"].Rows[0];
//Lay cac ban ghi con cua quan he bang Publishers va Titles
DataRow[] titleRows =
pubRow.GetChildRows("PubsTitles");
```

7/5/16

Lập trình CSDL với C#

29



Sử dụng các Constraint

- ❑ Bạn có thể tạo các ràng buộc riêng cho **DataSet** hoặc copy các ràng buộc tồn tại từ data source.
- ❑ Tạo mới các ràng buộc: có thể áp dụng 2 kiểu ràng buộc
 - ❑ **ForeignKeyConstraints**: đây là ràng buộc xảy ra khi một hàng con cũng được cập nhật khi hàng cha được update hoặc delete. Sau đây là bảng giá trị cho các thuộc tính **DeleteRule** và **UpdateRule**

Cascade	delete hoặc update bất kỳ bản ghi nào theo bản ghi cha
SetNull	Thiết lập giá trị liên quan thành DBNull
SetDefault	Thiết lập các giá trị liên quan theo giá trị mặc định
None	Không tác động tới các hàng liên quan

```
DataColumn colParent = dsPubs.Tables["Publishers"].Columns["pub_id"];
DataColumn colChild = dsPubs.Tables["Titles"].Columns["pub_id"];
ForeignKeyConstraint fkcPubsTitles = new ForeignKeyConstraint("PubsTitlesFKC",colParent,colChild);
fkcPubsTitles.DeleteRule = Rule.SetNull;
fkcPubsTitles.UpdateRule = Rule.Cascade;
dsPubs.Tables["Titles"].Constraints.Add(fkcPubsTitles);
dsPubs.EnforceConstraints = true;
```

7/5/16

Lập trình CSDL với C#

30



Sử dụng các Constraint

- ❑ **UniqueConstraints:** đây là ràng buộc đảm bảo các giá trị trong một cột hoặc nhiều cột là duy nhất.

```
UniqueConstraint ucTitles = new UniqueConstraint("UniqueTitles",
dsPubs.Tables["Titles"].Columns["title"]);
dsPubs.EnforceConstraints = true;
```

- ❑ Sử dụng Constraint tồn tại
 - ❑ Nếu đã tồn tại các ràng buộc trong RDBMS, có thể copy trực tiếp tới **DataSet**
 - ❑ Sử dụng phương thức **FillSchema** để copy

```
SqlConnection con = new SqlConnection("Server=TIENPT\\SQL;Database=Northwind;Uid =
sa; Pwd = admin123");
SqlDataAdapter adp = new SqlDataAdapter("Select title_id,title,type,price from Titles",con);
adp.FillSchema(dsPubs, SchemaType.Source, "Titles");
adp.Fill(dsPubs,"Titles");
//Thực hiện cập nhật dữ liệu trong dsPubs và cập nhật lại nguồn
adp.Fill(dsPubs,"Titles");
```



Cập nhật dữ liệu trong DataSet

- ❑ Thêm một Row mới

```
DataRow drNewRow = dsPubs.Tables["Titles"].NewRow();
drNewRow["title"] = "New Book";
drNewRow["type"] = "business";
dsPubs.Tables["Titles"].Rows.Add(drNewRow);
```

- ❑ Thay đổi Rows

```
DataRow drChangeRow = dsPubs.Tables["Titles"].Rows[0];
drChangeRow.BeginEdit();
drChangeRow["title"] = drChangeRow["title"].ToString() + " 1";
drChangeRow.EndEdit();
```

- ❑ Xóa dữ liệu

```
DataRow drDelRow = dsPubs.Tables["Titles"].Rows[0];
dsPubs.Tables["Titles"].Rows.Remove(drDelRow);
```

Phương thức này chỉ đánh dấu hàng được xóa, gọi **RejectChanges** sẽ undo việc xóa



Xác nhận thay đổi dữ liệu

- ❑ Để cập nhật **DataSet**, dùng các phương thức thích hợp để chỉnh sửa bảng, rồi gọi **AcceptChanges** hoặc **RejectChanges** cho mỗi dòng hoặc cho toàn bộ bảng
- ❑ Bạn có thể kiểm tra trạng thái của hàng qua thuộc tính **RowState**, các trạng thái:

Unchanged	Không có bất cứ thay đổi nào được làm
Added	Hàng dữ liệu đã bị thêm tới bảng
Modified	Một số thứ trong hàng đã bị thay đổi
Deleted	Hàng dữ liệu đã bị xoá bởi phương thức Delete
Detached	Hàng dữ liệu đã bị xoá hoặc hàng dữ liệu đã được tạo nhưng phương thức Add không được gọi



Cập nhật dữ liệu tại nguồn

- ❑ Sau khi update các bảng trong **DataSet**, chúng ta muốn cập nhật lên data source thì bạn dùng phương thức **Update** của đối tượng **DataAdapter**, nó liên kết giữa DataSet và data source
- ❑ Nó xác định các thay đổi của dữ liệu và thực thi câu lệnh SQL tương ứng (Insert, Update hoặc Delete)



Cập nhật dữ liệu tại nguồn

- Chỉ rõ các tham số cập nhật

```
SqlCommand cmd = new SqlCommand();
cmd.CommandText = "Insert into titles(title_id, title, type)
values(@t_id,@title,@type)";
cmd.Parameters.Add("@t_id", SqlDbType.VarChar,6,"title_id");
cmd.Parameters.Add("@title", SqlDbType.VarChar,80,"title");
cmd.Parameters.Add("@type", SqlDbType.Char,12,"type");
adp.InsertCommand = cmd;
adp.Update(dsPubs, "titles");
```

- Tự động phát sinh update

```
SqlCommandBuilder sqlCommBuild = new SqlCommandBuilder(adp);
adp.Update(dsPubs,"titles");
MessageBox.Show(sqlCommBuild.GetInsertCommand().ToString());
```

7/5/16

Lập trình CSDL với C#

35




Demo: Sử dụng DataSet



7/5/16


Lập trình CSDL với C#

36

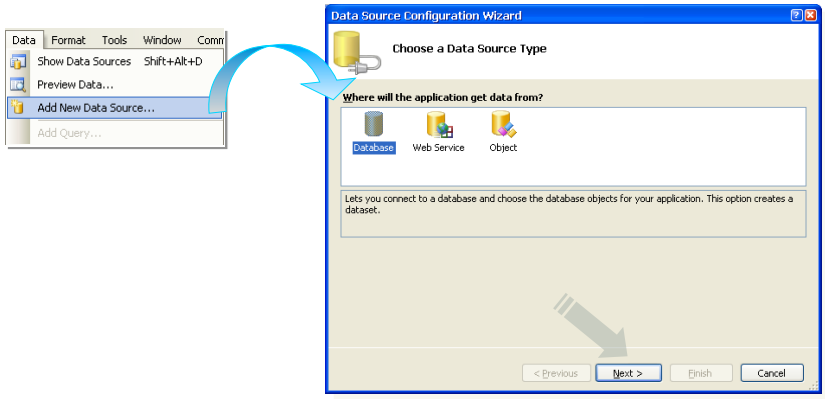
 **6. Sử dụng chức năng Wizard**

- Thiết lập nhanh Data Source cho project
- Xây dựng form hiển thị và thao tác dữ liệu
- Thông qua thao tác kéo thả từ Data Source

7/5/16 Lập trình CSDL với C# 37

 **Sử dụng chức năng Wizard**

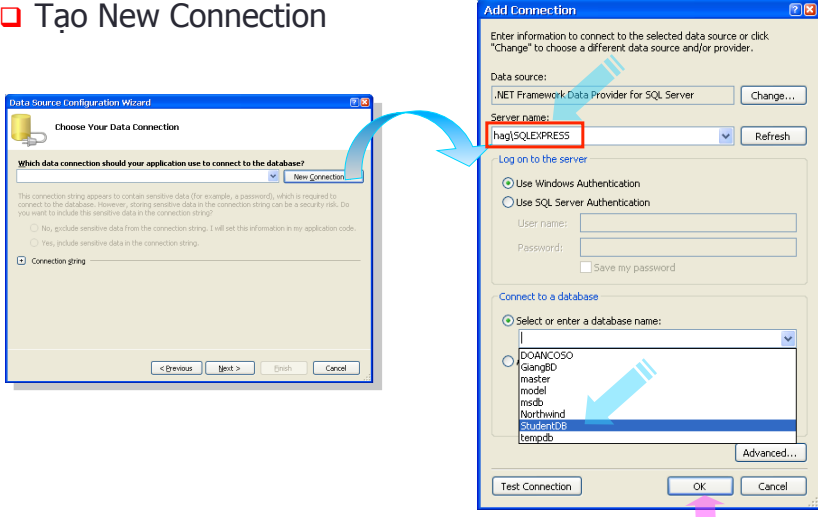
- ❑ Tạo một ứng dụng Windows Application.
- ❑ Trong menu Data | Add New Data Source...



7/5/16 Lập trình CSDL với C# 38

Sử dụng chức năng Wizard

❑ Tạo New Connection

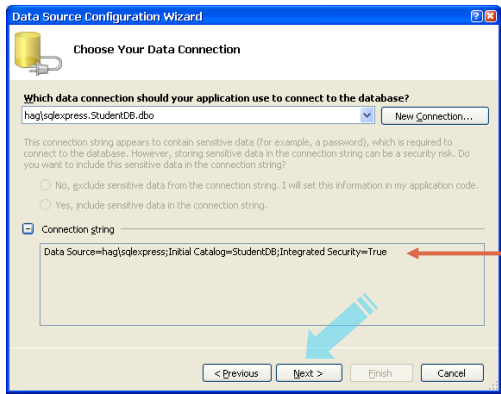


The image shows two windows from a .NET application. On the left is the 'Data Source Configuration Wizard' with the 'Choose Your Data Connection' step. On the right is the 'Add Connection' dialog box. In the 'Add Connection' dialog, the 'Server name' is set to 'hag\SQLEXPRESS'. Under 'Log on to the server', 'Use Windows Authentication' is selected. Under 'Connect to a database', 'Select or enter a database name' is selected, and a list of databases is shown, with 'StudentDB' highlighted. A pink arrow points to the 'OK' button.

7/5/16 Lập trình CSDL với C# 39

Sử dụng chức năng Wizard

❑ Hoàn tất khai báo Data Source



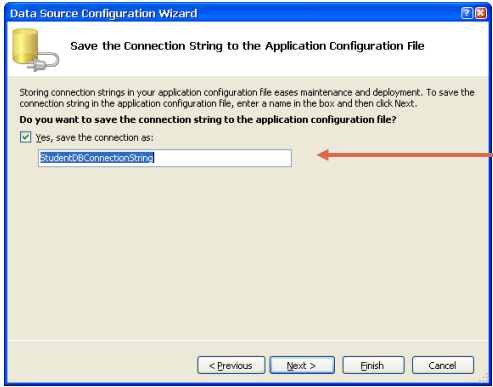
The image shows the 'Data Source Configuration Wizard' at the 'Choose Your Data Connection' step. The 'Which data connection should your application use to connect to the database?' dropdown is set to 'hag\sqlserver.StudentDB.dbo'. The 'Connection string' field contains 'Data Source=hag\sqlserver;Initial Catalog=StudentDB;Integrated Security=True'. A red arrow points to this field with the label 'Chuỗi kết nối'.

Chuỗi kết nối

7/5/16 Lập trình CSDL với C# 40

Sử dụng chức năng Wizard

❑ Lưu chuỗi kết nối trong file cấu hình

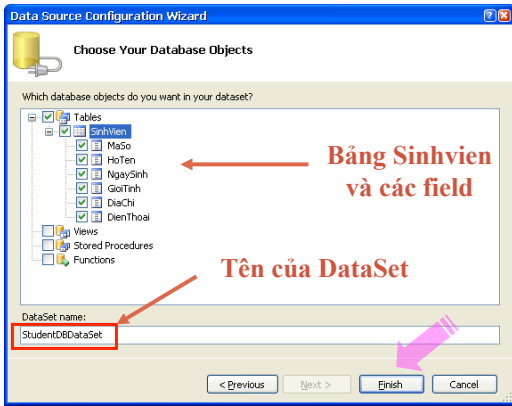


Tên của chuỗi kết nối

7/5/16 Lập trình CSDL với C# 41

Sử dụng chức năng Wizard

❑ Chọn bảng dữ liệu



Bảng Sinhvien và các field

Tên của DataSet

7/5/16 Lập trình CSDL với C# 42

Sử dụng chức năng Wizard

- Wizard sẽ tạo ứng dụng với Data Source

Data Sources Windows

Form in Design View

7/5/16 Lập trình CSDL với C# 43

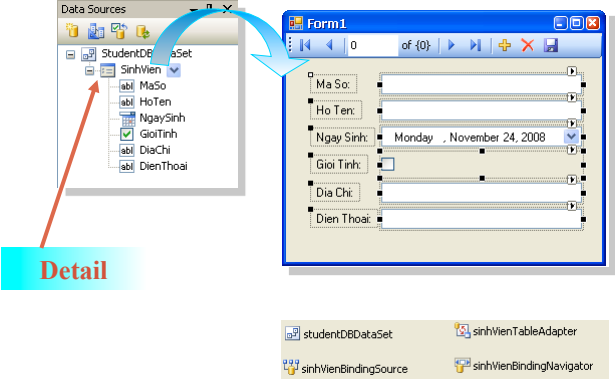
Sử dụng chức năng Wizard

- Kéo thả binding control vào Form
 - Trong cửa sổ Data Source
 - Chọn bảng cần sử dụng
 - Thiết lập view là DataGridView hay Details
 - Kích chọn vào đầu mũi tên xuống sau tên bảng

7/5/16 Lập trình CSDL với C# 44

Sử dụng chức năng Wizard

- ❑ Kéo Table thả vào Form
 - ❑ Tự động tạo các binding control cho table

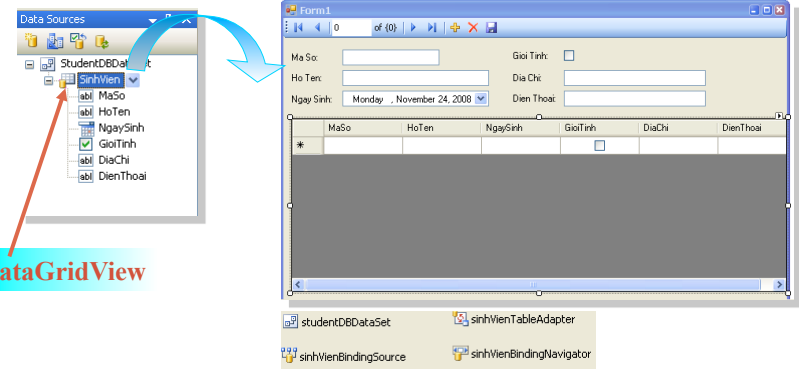


studentDBDataSet sinhVienTableAdapter
sinhVienBindingSource sinhVienBindingNavigator

7/5/16 Lập trình CSDL với C# 45


Sử dụng chức năng Wizard

- ❑ Bổ sung DataGridView cho Form
 - ❑ Thay đổi Table sang DataGridView
 - ❑ Kéo Table trong Data Source thả vào Form



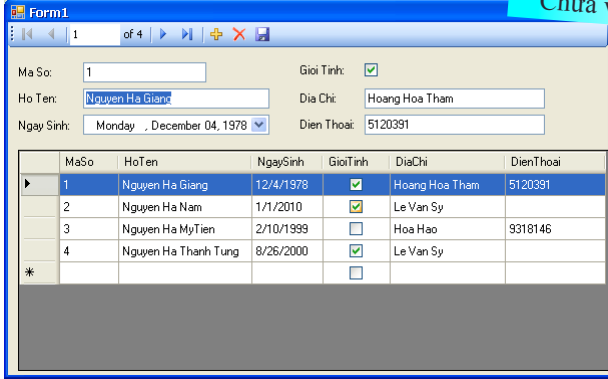
studentDBDataSet sinhVienTableAdapter
sinhVienBindingSource sinhVienBindingNavigator

7/5/16 Lập trình CSDL với C# 46

 **Sử dụng chức năng Wizard**

❑ Kết quả ứng dụng

Chưa viết code!



MaSo	HoTen	NgaySinh	GioiTinh	DiaChi	DienThoai
1	Nguyen Ha Giang	12/4/1978	<input checked="" type="checkbox"/>	Hoang Hoa Tham	5120391
2	Nguyen Ha Nam	1/1/2010	<input checked="" type="checkbox"/>	Le Van Sy	
3	Nguyen Ha MyTien	2/10/1999	<input type="checkbox"/>	Hoa Hao	9318146
4	Nguyen Ha Thanh Tung	8/26/2000	<input checked="" type="checkbox"/>	Le Van Sy	
*			<input type="checkbox"/>		

7/5/16 Lập trình CSDL với C# 47

 **Demo: Sử dụng chức năng Wizard**



7/5/16 Lập trình CSDL với C# 48



7. Thiết kế và gắn kết dữ liệu

- Thiết kế DataSet
- Tùy chỉnh form dữ liệu
- Gắn dữ liệu trong WinForms

- Gắn dữ liệu trong WebForms



Tùy chỉnh cấu hình DataAdapter

- Tên kết nối
- Kiểu truy vấn
 - Câu lệnh SQL
 - Hoặc Stored Procedure mới
 - Hoặc Stored Procedure đã tồn tại
- Các chi tiết lựa chọn câu truy vấn



Gắn dữ liệu trong WinForms

❑ Cách gắn đơn giản

```
SqlConnection con = new SqlConnection("Server=TIENPT\\SQL;Database=Northwind;Uid=sa;Pwd=admin123");
SqlDataAdapter adp = new SqlDataAdapter("Select * from Products", con);
DataSet ds = new DataSet();
adp.Fill(ds,"Products");
txtProduct.DataBindings.Add("Text", ds.Tables[0], "ProductName");
```

❑ Cách gắn phức tạp

```
SqlConnection con = new SqlConnection("Server=TIENPT\\SQL;Database=Northwind;Uid=sa;Pwd=admin123");
SqlDataAdapter adp = new SqlDataAdapter("Select * from Products", con);
DataSet ds = new DataSet();
adp.Fill(ds,"Products");
dgvProducts.DataSource = ds.Tables[0]; //DataGridView
dgvProducts.Refresh();
```

7/5/16

Lập trình CSDL với C#

51



Gắn dữ liệu trong WebForms

❑ Gắn dữ liệu chỉ đọc

```
protected void Page_Load(object sender, EventArgs e)
{
    SqlConnection conn = new SqlConnection("Server=TIENPT\\SQL;Database=NorthWind;Uid=sa;Pwd=admin123");
    conn.Open();
    SqlCommand cmd = new SqlCommand("Select * from Products", conn);
    SqlDataReader dr = cmd.ExecuteReader();
    grvProducts.DataSource = dr;
    grvProducts.DataBind();
}
```

7/5/16

Lập trình CSDL với C#

52



Tài liệu tham khảo

- ❑ Windows Forms Programming With C# - WIN0095 – Aptech Worldwide
- ❑ [https://msdn.microsoft.com/en-us/library/e80y5yhx\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/e80y5yhx(v=vs.110).aspx)
- ❑ <http://www.worldbestlearningcenter/>