



Mục tiêu

- Giúp sinh viên hiểu và nắm được
 - Các lưu trữ dữ liệu dùng DataSet
 - Cách sử dụng DataAdapter để đọc các dòng từ cơ sở dữ liệu
 - Trích lọc, sắp xếp dữ liệu từ DataTable thông qua DataView
 - Gắn kết dữ liệu vào một điều khiển Windows Form

Nội dung

- Lớp SqlDataAdapter
- Tạo và sử dụng đối tượng DataAdapter
- Lớp DataSet
- Tạo đối tượng DataSet
- Đưa dữ liệu vào DataSet
- Demo

Nội dung

- Lớp DataView
- Tạo và sử dụng DataView
- Demo

Lớp SqlDataAdapter

- Đóng vai trò cầu nối giữa nhóm lớp kết nối và không kết nối
- Dùng để lấy dữ liệu từ một hay nhiều bảng trong cơ sở dữ liệu, lưu vào DataSet / DataTable
- Dùng để đồng bộ hóa dữ liệu được lưu trong một DataSet với cơ sở dữ liệu SQL Server
- Sử dụng các phương thức
 - **Fill** Lấy dữ liệu → DataSet / DataTable
 - **Update** Cập nhật thay đổi lên cơ sở dữ liệu

Lớp SqlDataAdapter

- Các thuộc tính
 - Bool **AcceptChangesDuringFill** { get; set; }
 - Bool **ContinueUpdateOnError** { get; set; }
 - SqlCommand **DeleteCommand** { get; set; }
 - SqlCommand **InsertCommand** { get; set; }
 - SqlCommand **SelectCommand** { get; set; }
 - SqlCommand **UpdateCommand** { get; set; }
 - MissingMappingAction **MissingMappingAction** { get; set; }
 - MissingSchemaAction **MissingSchemaAction** { get; set; }
 - DataTableMappingCollection **TableMappings** { get; }

Lớp SqlDataAdapter

- Các phương thức
 - Int **Fill** ()
 - Int **Update** ()
 - DataTable **FillSchema** ()
 - IDataParameter[] **GetFillParameters** ()
- Sự kiện
 - FillErrorEventHandler **FillError**
 - RowUpdatingEventHandler **RowUpdating**
 - RowUpdatedEventHandler **RowUpdated**

Tạo đối tượng DataAdapter

- Sử dụng một trong các phương thức khởi tạo của lớp SqlDataAdapter như sau
 - **SqlDataAdapter** ()
 - **SqlDataAdapter** (SqlCommand mySqlCommand)
 - **SqlDataAdapter** (string selectCommandString, SqlConnection mySqlConnection)
 - **SqlDataAdapter** (string selectCommandString, string connectionString)
- Trong đó
 - **mySqlCommand** là đối tượng thực thi lệnh
 - **selectCommandString** là lệnh SQL SELECT hoặc tên một thủ tục.
 - **mySqlConnection** là đối tượng kết nối cơ sở dữ liệu
 - **connectionString** là chuỗi chứa thông tin kết nối cơ sở dữ liệu.

Tạo đối tượng DataAdapter

```
SqlCommand mySqlCommand = mySqlConnection.CreateCommand();
```

```
mySqlCommand.CommandText =  
    "SELECT TOP 5 ProductID, ProductName, UnitPrice " +  
    "FROM Products ORDER BY ProductID";
```

- Ví dụ 1

```
SqlDataAdapter mySqlDataAdapter = new SqlDataAdapter();  
mySqlDataAdapter.SelectCommand = mySqlCommand;
```

- Ví dụ 2

```
SqlDataAdapter mySqlDataAdapter = new SqlDataAdapter(mySqlCommand);
```

Tạo đối tượng DataAdapter

```
string selectCommandString = "SELECT TOP 10 ProductID, ProductName, UnitPrice  
    FROM Products ORDER BY ProductID";
```

- Ví dụ 3

```
SqlConnection mySqlConnection = new SqlConnection(  
    "server=localhost;database=Northwind;uid=sa;pwd=sa" );
```

```
SqlDataAdapter mySqlDataAdapter = new SqlDataAdapter(  
    selectCommandString, mySqlConnection);
```

- Ví dụ 4

```
string connectionString = "server=localhost;database=Northwind;  
    uid=sa;pwd=sa";
```

```
SqlDataAdapter mySqlDataAdapter = new SqlDataAdapter(  
    selectCommandString, connectionString);
```

Lớp DataSet

- Biểu diễn một bản sao thông tin được lưu trong cơ sở dữ liệu
 - Bảng (**DataTable** ⇔ Table)
 - Dòng (**DataRow** ⇔ Record / Row)
 - Cột (**DataColumn** ⇔ Column / Field)
 - Ràng buộc (Constraint)
 - Quan hệ (**DataRelation** ⇔ Relationship)
- Có thể tạo ra các thay đổi dữ liệu trên DataSet mà không làm ảnh hưởng đến cơ sở dữ liệu và sau đó đồng bộ hóa các thay đổi với cơ sở dữ liệu qua đối tượng DataAdapter.

Lớp DataSet

- Một số thuộc tính
 - Bool **CaseSensitive** { get; set; }
 - String **DataSetName** { get; set; }
 - Bool **EnforceConstraints** { get; set; }
 - Bool **HasErrors** { get; }
 - DataRelation **Relations** { get; }
 - DataTableCollection **Tables** { get; }
 - PropertyCollection **ExtendedProperties** { get; }
 - DataViewManager **DefaultViewManager** { get; }

Lớp DataSet

- Một số phương thức
 - Bool `HasChanges ()`
 - Void `AcceptChanges ()`
 - Void `RejectChanges ()`
 - Void `Clear ()`
 - Void `Merge ()`
 - Void `Reset ()`
 - DataSet `Clone ()`
 - DataSet `Copy ()`
 - DataSet `GetChanges ()`
- Sự kiện
 - `MergeFailedEventHandler` `MergeFailed`

Tạo đối tượng DataSet

- Dùng một trong hai phương thức khởi tạo sau
 - `DataSet ()`
 - `DataSet (string dataSetNameString)`
- Ví dụ
 - `//Tạo một đối tượng DataSet mới`
`DataSet myDataSet = new DataSet();`
 - `//Tạo một đối tượng DataSet mới có tên là myDataSet`
`DataSet myDataSet = new DataSet("myDataSet");`

Đưa dữ liệu vào DataSet / DataTable

- Sử dụng phương thức **Fill** của DataAdapter
 - Trả về số nguyên chỉ ra số dòng được đưa vào *DataSet*.
 - Các dạng của phương thức Fill

```
int Fill (DataSet myDataSet)
int Fill (DataTable myDataTable)
int Fill (DataSet myDataSet, string dataTableName)
int Fill (DataSet myDataSet, int startRow, int numOfWeeks, string dataTableName)
```
- Trường hợp 1: Lấy toàn bộ tập kết quả
- Trường hợp 2: Lấy một phần của tập kết quả
- Trường hợp 3: Sử dụng Stored Procedure

Demo: Lấy tất cả các dòng

```
mySqlCommand.CommandText = "SELECT TOP 5 ProductID, ProductName, UnitPrice " +
    "FROM Products ORDER BY ProductID";

SqlDataAdapter mySqlDataAdapter = new SqlDataAdapter();
mySqlDataAdapter.SelectCommand = mySqlCommand;

DataSet myDataSet = new DataSet();
mySqlConnection.Open();

int numberOfRows = mySqlDataAdapter. Fill (myDataSet, "Products");
mySqlConnection.Close();

DataTable myDataTable = myDataSet.Tables["Products"];
// DataTable myDataTable = myDataSet.Tables[0];

foreach (DataRow myDataRow in myDataTable.Rows)
{
    Console.WriteLine("ProductID = " + myDataRow["ProductID"]);
    Console.WriteLine("ProductName = " + myDataRow["ProductName"]);
    Console.WriteLine("UnitPrice = " + myDataRow["UnitPrice"]);
}
```


Demo: Lấy một phần của tập kết quả

```
mySqlCommand.CommandText = "SELECT TOP 5 ProductID, ProductName, UnitPrice " +  
    "FROM Products ORDER BY ProductID";  
  
SqlDataAdapter mySqlDataAdapter = new SqlDataAdapter();  
mySqlDataAdapter.SelectCommand = mySqlCommand;  
  
DataSet myDataSet = new DataSet();  
int numberOfRows = mySqlDataAdapter.Fill(myDataSet, 1, 3, "Products");  
  
DataTable myDataTable = myDataSet.Tables[0];  
  
foreach (DataRow myDataRow in myDataTable.Rows)  
{  
    Console.WriteLine("ProductID = " + myDataRow["ProductID"]);  
    Console.WriteLine("ProductName = " + myDataRow["ProductName"]);  
    Console.WriteLine("UnitPrice = " + myDataRow["UnitPrice"]);  
}
```

Nguyễn Văn Phúc - Khoa Công nghệ Thông tin - Đại học Đà Lạt

17

Demo: sử dụng Stored Procedure

```
SqlCommand mySqlCommand = mySqlConnection.CreateCommand();  
  
mySqlCommand.CommandText = "EXECUTE CustOrderHist @CustomerID";  
mySqlCommand.Parameters.Add("@CustomerID", SqlDbType.NVarChar, 5).Value =  
    "ALFKI";  
  
SqlDataAdapter mySqlDataAdapter = new SqlDataAdapter();  
mySqlDataAdapter.SelectCommand = mySqlCommand;  
  
DataSet myDataSet = new DataSet();  
mySqlConnection.Open();  
  
int numberOfRows = mySqlDataAdapter.Fill(myDataSet, "CustOrderHist");  
mySqlConnection.Close();
```

Nguyễn Văn Phúc - Khoa Công nghệ Thông tin - Đại học Đà Lạt

18

Đưa dữ liệu vào nhiều DataTable của một DataSet

- Giải pháp
 - Dùng **nhiều lệnh SELECT** trong thuộc tính SelectCommand
 - **Thay đổi giá trị cho thuộc tính CommandText** của SelectCommand trong DataAdapter trước mỗi lần gọi hàm Fill
 - **Sử dụng nhiều đối tượng DataAdapter** trên cùng một DataSet
- Với cách thứ nhất, tên các bảng được đặt tuần tự như sau
 - Table
 - Table1
 - ...
- **Đổi tên bảng** theo cú pháp sau
 - `myDataSet.Tables["Tên_Cũ"].TableName = "Tên_Mới";`

Lớp DataView

- Được dùng để xem nội dung các dòng trong một DataTable qua một bộ lọc.
- Sắp xếp các dòng hoặc thêm dòng mới, cập nhật và xóa các dòng khỏi DataTable thông qua một DataView.
- Các thay đổi dữ liệu trên DataView cũng được cập nhật cho các DataRow trong DataTable mà nó đọc dữ liệu
- Sự kiện
 - `ListChangedEventHandler` `ListChanged`

Lớp DataView

- Một số thuộc tính
 - Bool AllowDelete { get; set; }
 - Bool AllowEdit { get; set; }
 - Bool AllowNew { get; set; }
 - Bool ApplyDefaultSort { get; set; }
 - Int Count { get; }
 - String RowFilter { get; set; }
 - String Sort { get; set; }
- DataTable Table { get; set; }
- DataViewManager DataViewManager { get; }
- DataRowState RowStateFilter { get; set; }

Nguyễn Văn Phúc - Khoa Công nghệ Thông tin - Đại học Đà Lạt

Lớp DataView

- Một số phương thức
 - Void BeginInit ()
 - Void CopyTo ()
 - Void Delete ()
 - Void EndInit ()
 - Int Find ()
 - DataRowView AddNew ()
 - DataRowView[] FindRows ()
 - IEnumerator GetEnumerator ()
 - String ToString ()

Nguyễn Văn Phúc - Khoa Công nghệ Thông tin - Đại học Đà Lạt

Tạo và sử dụng DataView

- Để tạo một đối tượng DataView, ta dùng một trong các phương thức khởi tạo sau
 - DataView ()
 - DataView (DataTable myDataTable)
 - DataView (DataTable myDataTable, string filterExpression, string sortExpression, DataViewRowState rowState)

CONSTANT	DESCRIPTION
Added	Dòng mới thêm.
CurrentRows	Dòng hiện tại bao gồm luôn cả: Unchanged, Added, and ModifiedCurrent.
Deleted	Dòng đã bị xóa.
ModifiedCurrent	Dòng hiện tại đã bị thay đổi.
ModifiedOriginal	Dòng ban đầu trước khi thay đổi.
None	Không phù hợp cho bất kỳ dòng nào trong DataTable.
OriginalRows	Các dòng ban đầu bao gồm cả Unchanged và Deleted.
Unchanged	Dòng chưa bị thay đổi.

Nguyễn Văn Phúc - Khoa Công nghệ Thông tin - Đại học Đà Lạt

Tạo và sử dụng DataView

Cách 1

```
string filterExpression = "Country = 'UK'";
string sortExpression = "CustomerID ASC, CompanyName DESC";
DataViewRowState rowStateFilter = DataViewRowState.OriginalRows;
DataView customersDV = new DataView(
    customersDT, filterExpression, sortExpression, rowStateFilter);
```

Cách 2

```
DataView customersDV = new DataView();
customersDV.Table = customersDT;
customersDV.RowFilter = filterExpression;
customersDV.Sort = sortExpression;
customersDV.RowStateFilter = rowStateFilter;
```

Nguyễn Văn Phúc - Khoa Công nghệ Thông tin - Đại học Đà Lạt

Demo: Tạo và sử dụng DataView

```
// create a SqlConnection object to connect to the database
SqlConnection sqlConnection = new SqlConnection(
    "server=.; database=RestaurantManagement; Integrated
    Security=true;");

// create a SqlCommand object
SqlCommand mySqlCommand = sqlConnection.CreateCommand();

// create a SqlDataAdapter object
SqlDataAdapter myDataAdapter = new SqlDataAdapter(mySqlCommand);

// create a DataSet object
DataSet myDataSet = new DataSet();

// set the CommandText property of the SqlCommand object to
// the SELECT statement
mySqlCommand.CommandText = "Select Name, Unit, Price from Food";

// Open the connection to the database
sqlConnection.Open();

// Execute the sql command in the CommandText and save the result
// in the myDataSet dataset, table name is "Food"
int numofRows = myDataAdapter.Fill(myDataSet, "Food");

// Close the connection to the database
sqlConnection.Close();
```

25

Demo: Tạo và sử dụng DataView

```
// Get table "Food" from myDataSet
DataTable foodTable = myDataSet.Tables["Food"];

// create filter and sort expression
string filterExpression = "Name like '%ng%'";
string sortExpression = "Price DESC";
DataViewRowState rowStateFilter = DataViewRowState.OriginalRows;

// Create a data view object to view the data in foodDT data table
// filter by Name (contain 'ng') and sort descending by Price
DataView foodView = new DataView(foodTable,
    filterExpression, sortExpression, rowStateFilter);

// Assign foodDT as Data Source of data grid view
dgvFoodList.DataSource = foodView;
```

26

Demo: Tạo và sử dụng DataView

Danh sách món ăn

	Tên món ăn	Đơn vị tính	Đơn giá
▶	Gà nướng muối ớt	Con	180000
	Càng cua hấp	Đĩa	100000
	Mực nướng tỏi ớt	Đĩa	90000
	Gà nướng mật ong	Đĩa	90000
	Sò lông nướng mỡ hành	Đĩa	80000
	Cơm chiên Dương châu	Đĩa lớn	40000
	Cơm chiên Dương châu	Đĩa nhỏ	35000
	Rau muống xào tỏi	Đĩa	20000
*			

27

Thực hành

- Làm việc theo nhóm
- Tiếp tục thực hiện các yêu cầu theo đề tài nhóm dựa trên các kiến thức vừa được học
 - Viết các hàm xử lý việc sắp xếp trên các cột
 - Viết các hàm trích lọc & tìm kiếm dữ liệu

28