C# Sharp Development

Bài 8

LINQ, Entity Framework, Lambda expressions

Mục tiêu

- Giới thiệu về LINQ
- LINQ Provider
- Một số thao tác cơ bản trên LINQ
- Một số phương thức mở rộng trên LINQ

Giới thiệu về LINQ

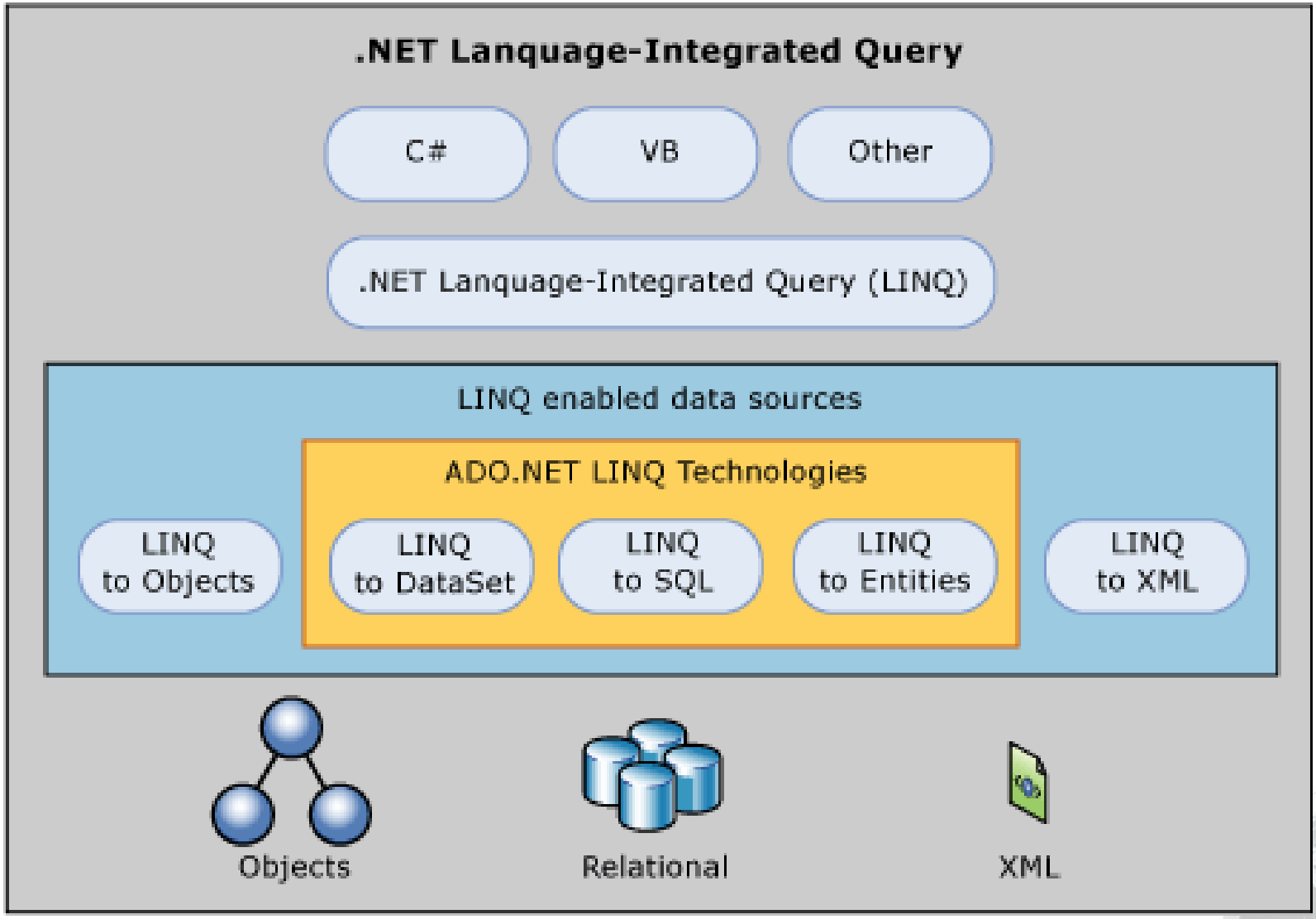
- Để giảm gánh nặng thao tác trên nhiều ngôn ngữ khác nhau và cải thiện năng suất lập trình, Microsoft đã phát triển giải pháp tích hợp dữ liệu cho .NET Framework có tên gọi là LINQ (Language Integrated Query)
- Đây là thư viện mở rộng cho các ngôn ngữ lập trình C# và Visual Basic.NET (có thể mở rộng cho các ngôn ngữ khác) cung cấp khả năng truy vấn trực tiếp dữ liệu Object, cơ sở dữ liệu và XML.
- Điểm mạnh của LINQ là "viết truy vấn cho rất nhiều các đối tượng dữ liệu". Từ cơ sở dữ liệu, XML, Data Object ... thậm chí là viết truy vấn cho một biến mảng đã tạo ra trước đó.
- Vì thế ta có các phương pháp truy vấn như là LinQ to SQL, LinQ to XML,....

LINQ is supported by .NET 3.5 and C# 3.0

DEVMASTER.VN

2002/2003	2005	2006	2008
Microsoft Visual Studio® 2002/2003	Visual Studio 2005	Visual Studio 2005 + Extensions	**Visual Studio 2008**
C# 1.0	C# 2.0	C# 2.0	**C# 3.0**
.NET 1.0/1.1	.NET 2.0	.NET 3.0	**.NET 3.5**
CLR 1.0	CLR 2.0	CLR 2.0	**CLR 2.0 SP1**

LINQ providers



LINQ to Objects

- Thuật ngữ "LINQ to Objects" đề cập đến việc sử dụng các truy vấn LINQ trực tiếp với bất kỳ dữ liệu dạng tập hợp IEnumerable or IEnumerable<T> , mà không cần sử dụng một LINQ provider hay API trung gian nào như LINQ to SQL hay LINQ to XML.
- Bạn có thể sử dụng LINQ truy vấn bất kỳ bộ tập hợp, liệt kê nào như List<T>, Array, hoặc Dictionary<TKey, TValue>.
- Tập hợp có thể do người dùng định nghĩa hoặc có thể được trả về từ API của .NET Framework.

Ví dụ về LINQ to Object

DEVMASTER.VN

Ví dụ

```
string[] a = { "Sunday", "Monday", "Tuesday", "Wednesday",
                 "Thursday", "Friday", "Saturday" };
IEnumerable<string> days = from day in a select day;
foreach (var day in days)
{
    Console.WriteLine(day);
}
```

Linq to Entities

- LINQ to Entities là một Provider nằm trong LINQ Framework
  - Cho phép bạn truy vấn dữ liệu trên mô hình ADO.NET Entity Framework.
  - LINQ to Entities có tính mềm dẻo và khá mạnh mẽ

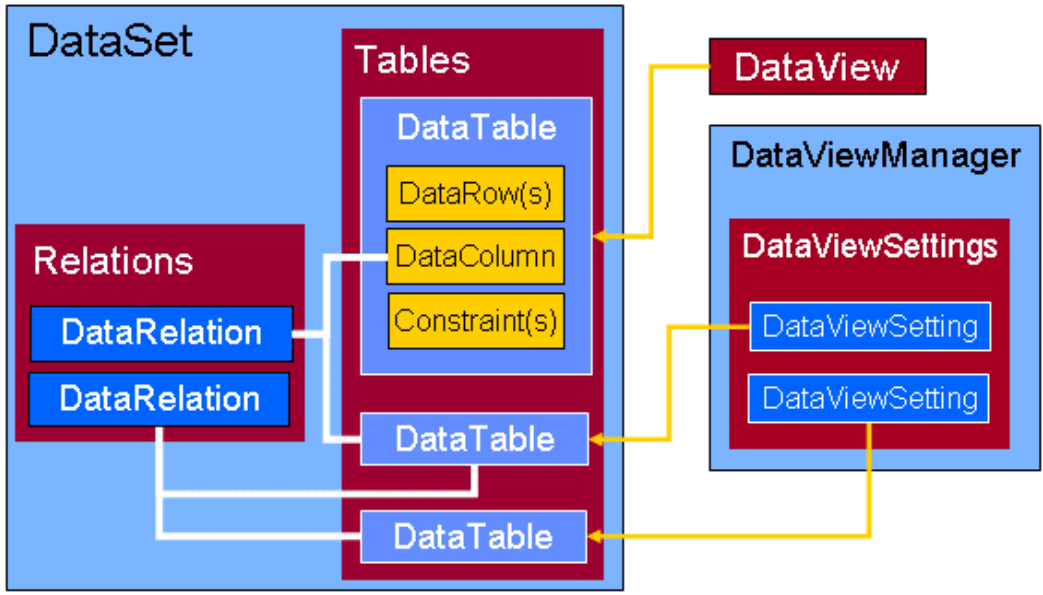
Ví dụ

```
using (var context = new NORTHWNDEntities())
{
    var listCustomers = from c in context.Customers
        where c.Address.StartsWith("a")
        select c;
}
```

Linq to DataSet

- DataSet là một cơ sở dữ liệu thu nhỏ trên bộ nhớ và từ lâu nó đã là một thành phần truy xuất dữ liệu trong .NET.

- LINQ to DataSet cho phép bạn sử dụng các toán tử truy vấn để truy vấn dữ liệu bên trong một DataSet hoặc DataTable.



Linq to DataSet

DEVMASTER.VN

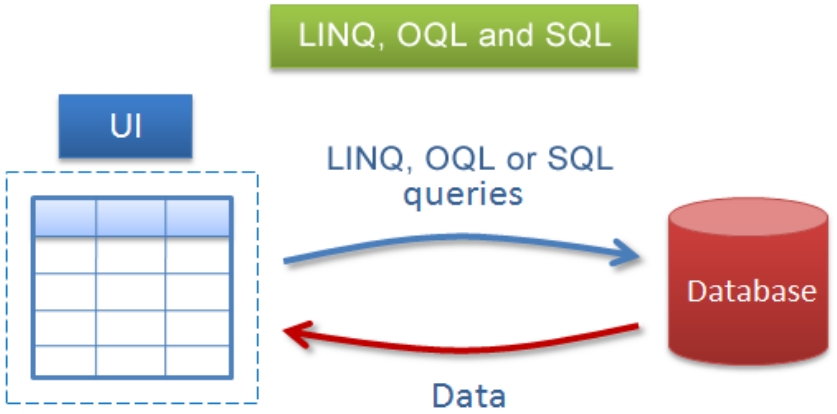
Ví dụ

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

DataTable products = ds.Tables[0];
IEnumerable<DataRow> query = from row in products.AsEnumerable()
                             select row;
Console.WriteLine("Name:");
foreach (DataRow item in query)
{
    Console.WriteLine(item.Field<string>("Name"));
}
```

Linq to SQL

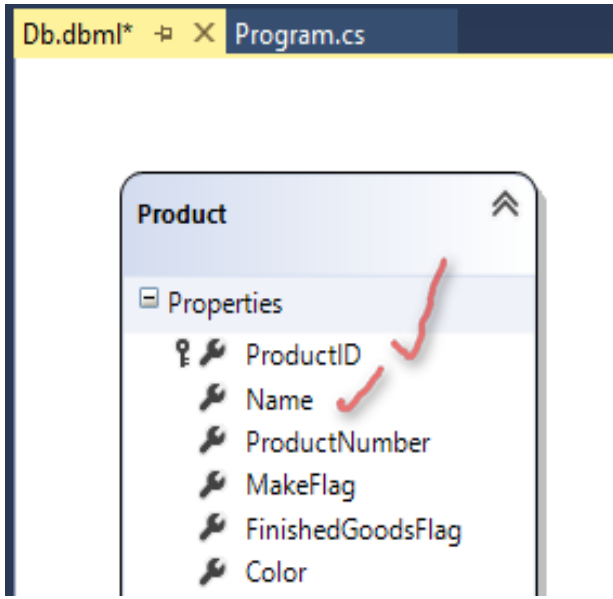
- LINQ to SQL là một công cụ ORM kết nối với cơ sở dữ liệu SQL Server.
- Nó tạo ra các đối tượng nghiệp vụ phù hợp với cấu trúc bảng của cơ sở dữ liệu, mà tất cả đều có sẵn thông qua các lớp DataContext.
- Các nhà thiết kế LINQ to SQL tạo ra một lớp DataContext tùy chỉnh có chứa một loạt các đối tượng Table <Entity> đại diện cho mỗi bảng trong cơ sở dữ liệu của bạn.

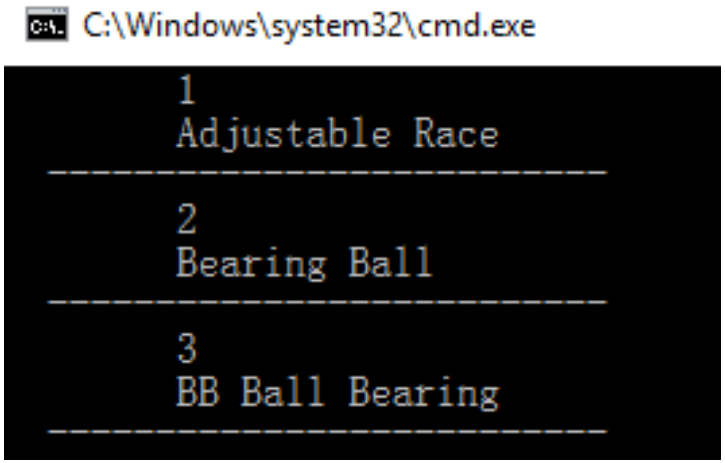


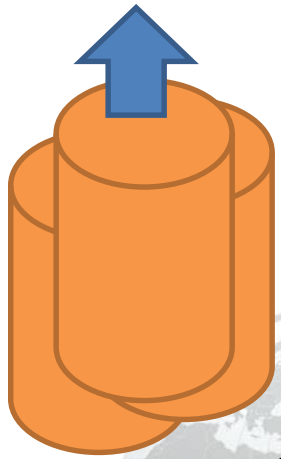
Linq to SQL

Ví dụ

```
//Sử dụng csdl AdventureWorks
DbDataContext db = new DbDataContext();
var products = from product in db.Products
               select product;
foreach (var p in products)
{
    Console.WriteLine(p.ProductID);
    Console.WriteLine(p.Name);
    Console.WriteLine("---------------------------");
}
```



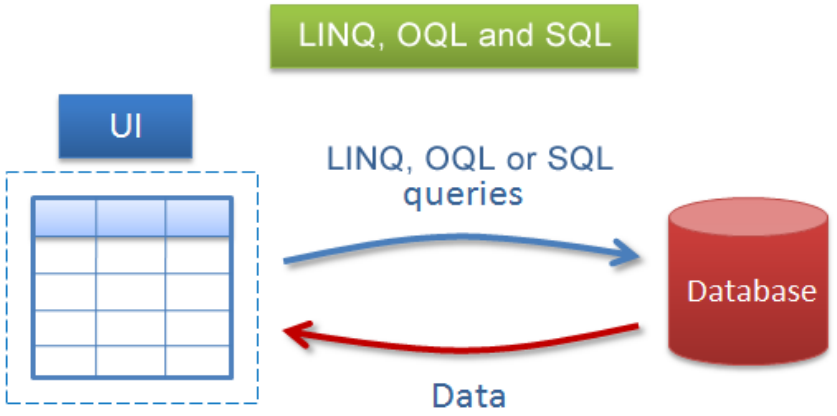




SQL Server Database

Linq to Entity Framework

- LINQ to SQL là một công cụ ORM kết nối với cơ sở dữ liệu SQL Server.
- Nó tạo ra các đối tượng nghiệp vụ phù hợp với cấu trúc bảng của cơ sở dữ liệu, mà tất cả đều có sẵn thông qua các lớp DataContext.
- Các nhà thiết kế LINQ to SQL tạo ra một lớp DataContext tùy chỉnh có chứa một loạt các đối tượng Table <Entity> đại diện cho mỗi bảng trong cơ sở dữ liệu của bạn.



Một số thao tác cơ bản trong Linq 1-4

DEVMASTER.VN

Tạo 2 lớp chứa dữ liệu

```
class Customer
{
    public Customer(int id, string name, string city, int age)
    {
        Id = id;
        Name = name;
        City = city;
        Age = age;
    }
    public int Id { get; set; }
    public string Name { get; set; }
    public string City { get; set; }
    public int Age { get; set; }
    public override string ToString()
    {
        return Id + " | " + Name + " | " + City + " | " + Age + "\n";
    }
}
class Province
{
    public Province(string city, int capacity)
    {
        City = city;
        Capacity = capacity;
    }
    public string City { get; set; }
    public int Capacity { get; set; }
}
```

Một số thao tác cơ bản trong Linq 2-4

DEVMASTER.VN

Khởi tạo dữ liệu

```
Customer[] customers ={
                        new Customer(1,"Dung","Ha Noi",20),
                        new Customer(2,"Long","Ha Tay",18),
                        new Customer(3,"Anh","Ha Noi",30),
                        new Customer(4,"Trang","Nam Dinh",26)
                      };
Province[] provinces ={
                         new Province("Ha Noi",100000),
                         new Province("Nam Dinh",80000)
                      };
```

Truy vấn dữ liệu

```
//lấy tất cả các khách hàng
IEnumerable<Customer> listCustom = from c in customers select c;
//duyệt và in ra
foreach (var c in listCustom)
{
    Console.WriteLine(c.ToString());
}
```

Một số thao tác cơ bản trong Linq 3-4

DEVMASTER.VN

Lọc dữ liệu

```
//lọc tất cả khách hàng có tuổi nằm trong khoảng 20-30
var listcust = from c in customers where c.Age > 20 && c.Age < 30
               select c;
foreach (var cust in listcust)
{
    Console.WriteLine(cust.ToString());
}
```

Một số thao tác cơ bản trong Linq 4-4

DEVMASTER.VN

Join 2 tập dữ liệu

```
//join hai khách hàng và thành phố và lấy ra thông tin chung
var listcustcity = from cust in customers
                   join pro in provinces on cust.City equals pro.City
                   select new { Id = cust.Id, Name = cust.Name,
                       City = cust.City, Capacity = pro.Capacity };
foreach (var item in listcustcity)
{
    Console.WriteLine(item.Id + " | " + item.Name + " | " +
        item.City + " | " + item.Capacity + "\n");
}
```

Một số phương thức mở rộng trong Linq 1-3

Khởi tạo 3 tập dữ liệu sau

```
//khai báo chung
int[] Numbers = { 7, 9, 3, 5, 2, 1, 0, 6, 4, 3, 1 };

string[] Words = {
                  "Chỉ","trích","phê", "phán", "người","khác",
                  "giống","như","con","chim","bồ","câu","đưa","thư",
                  "bao", "giờ","cũng", "quay","về","nơi","xuất", "phát"
                 };

List<Film> ListFilm = new List<Film>()
{
   new Film{FilmId="F01",FilmName="Điệp viên 007",Price=120000},
   new Film{FilmId="F02",FilmName="Tam quốc diễn nghĩa",Price=130000},
   new Film{FilmId="F03",FilmName="Thiếu lâm truyền kỳ",Price=16000},
   new Film{FilmId="F04",FilmName="Người nhện 2",Price=100000},
   new Film{FilmId="F05",FilmName="Ngân hàng tình yêu",Price=340000},
   new Film{FilmId="F06",FilmName="Người đẹp và quái thú",Price=230000},
   new Film{FilmId="F07",FilmName="Biệt động Sài Gòn",Price=190000},
};
```

Một số phương thức mở rộng trong Linq 2-3

DEVMASTER.VN

Phương thức Distinct

```
//Lọc các số duy nhất trong tập các số
var querynumber = Numbers.Distinct();
MessageBox.Show("Số phần tử tìm được:" + querynumber.Count());
//Đếm xem có bao nhiêu từ không trùng nhau
var count = Words.Distinct().Count();
MessageBox.Show("Đếm được:" + count);
```

Phương thức Take

```
//lấy 4 số đầu tiên trong dãy
var querynumber = Numbers.Take(4);
//lấy 2 từ đầu tiên trong câu
var queryword = Words.Take(2);
```

Một số phương thức mở rộng trong Linq 3-3

DEVMASTER.VN

Phương thức Skip

```
//bỏ qua 3 phần từ đầu tiên, lấy tất cả các phần tử còn lại
var querynumber = Numbers.Skip(3);
MessageBox.Show("Số phần tử:" + querynumber.Count());
//bỏ qua 4 phần từ đầu tiên lấy 3 phần tử kế tiếp
var querynumber1 = Numbers.Skip(4).Take(3);
//bỏ qua 3 phim đầu tiên lấy 3 phim kết tiếp
//(có thể áp dụng để phân trang)
var queryfilm = ListFilm.Skip(3).Take(3);
```

Phương thức SkipWhile

```
//Sắp xếp giảm dần, sau đó lấy các phần tử <5
var querynumber =
    Numbers.OrderByDescending(x => x).SkipWhile(x=>x > 5);
MessageBox.Show("số phẩn tử:" + querynumber.Count());
```

Hỏi Đáp