

Bài 14

Tính năng nâng cao trong C#

Mục tiêu

- System-Defined Generic Delegate
- Phương thức Anonymous
- Biểu thức Lambda
- Ngôn ngữ Linq
- Kết hợp Lambda với Linq methods
- Phương thức mở rộng
- Lớp thành phần
- Kiểu Nullable
- Xử lý bất đồng bộ

System-Defined Generic Delegates

- Là các delegate được định nghĩa sẵn trong hệ thống với 16 dạng khác nhau cho mọi kiểu dữ liệu, do đó gần như chúng ta không phải định nghĩa delegate riêng.

`Func<T1, T2 ,..., T16, TResult>(T1 arg1, T2 arg2,..., T16 arg16)`

T1 ...T2 là kiểu của các tham số và **TResult** là kiểu trả về của delegate.

Nếu kiểu trả về là void hãy dùng delegate `Action<T1, T2,..>(...)`

```
class Program
{
    static void Main(string[] args)
    {
        //sử dụng delegate có sẵn tham chiếu tới phương thức CountWord
        Func<string, int> cw = CountWord;
        Console.WriteLine(cw("I love VietNam")); //gọi phương thức
    }
    public static int CountWord(string str)
    {
        return str.Split(new char[] { ' ' }).Length;
    }
}
```

Phương thức nặc danh (Anonymous method)

- Đôi khi việc tạo 1 phương thức riêng biệt chỉ để gọi thông qua delegate sẽ trở nên cồng kềnh chính vì vậy Anonymous method được đưa vào sử dụng.
- Anonymous method à một khối code dạng inline không có tên và được tạo với từ khóa delegate, do đó nó được tham chiếu bởi delegate.

```
class Program
{
    static void Main(string[] args)
    {
        Thread th1 = new Thread(
            delegate ()
            {
                Console.WriteLine("Thread 1 running!");
            }
        );
        th1.Start();
    }
}
```

```
class Program
{
    static void Main(string[] args)
    {
        Func<string, int> cw = delegate (string str)
        {
            return str.Split(new char[] { ' ' }).Length;
        };
        Console.WriteLine(cw("I love csharp"));
    }
}
```

Lambda Expression

- Lambda expression là dạng biểu thức nặc danh (rút gọn của phương thức nặc danh), nó chứa các biểu thức hoặc câu lệnh đơn giản dạng inline.

CÚ PHÁP:

```
parameter_list=>expression or statement
```

Trong đó:

`parameter_list`: là danh sách tham số

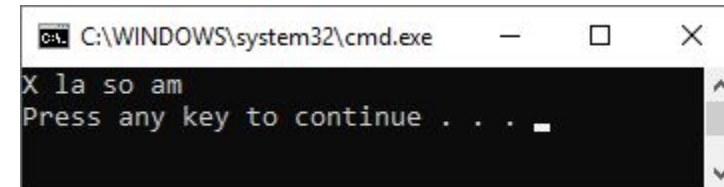
`expression or statement`: là biểu thức hoặc câu lệnh

VÍ DỤ:

```
word=>word.Length
```

Lambda Expression

```
class Program
{
    static void Main(string[] args)
    {
        //trở delegate tới 1 phương thức nặc danh kiểm tra số dương hay âm
        Func<int, bool> fnormal = delegate (int n) { return n >= 0; };
        //rút gọn hơn (biểu thức lambda)
        Func<int, bool> fshort = (int n) => { return n >= 0; };
        //rất gọn (biểu thức lambda)
        Func<int, bool> fveryshort = n => n >= 0;
        //gọi hàm
        int x = -5;
        if(fveryshort(x))
        {
            Console.WriteLine("X là số dương");
        }
        else
        {
            Console.WriteLine("X là số âm");
        }
    }
}
```



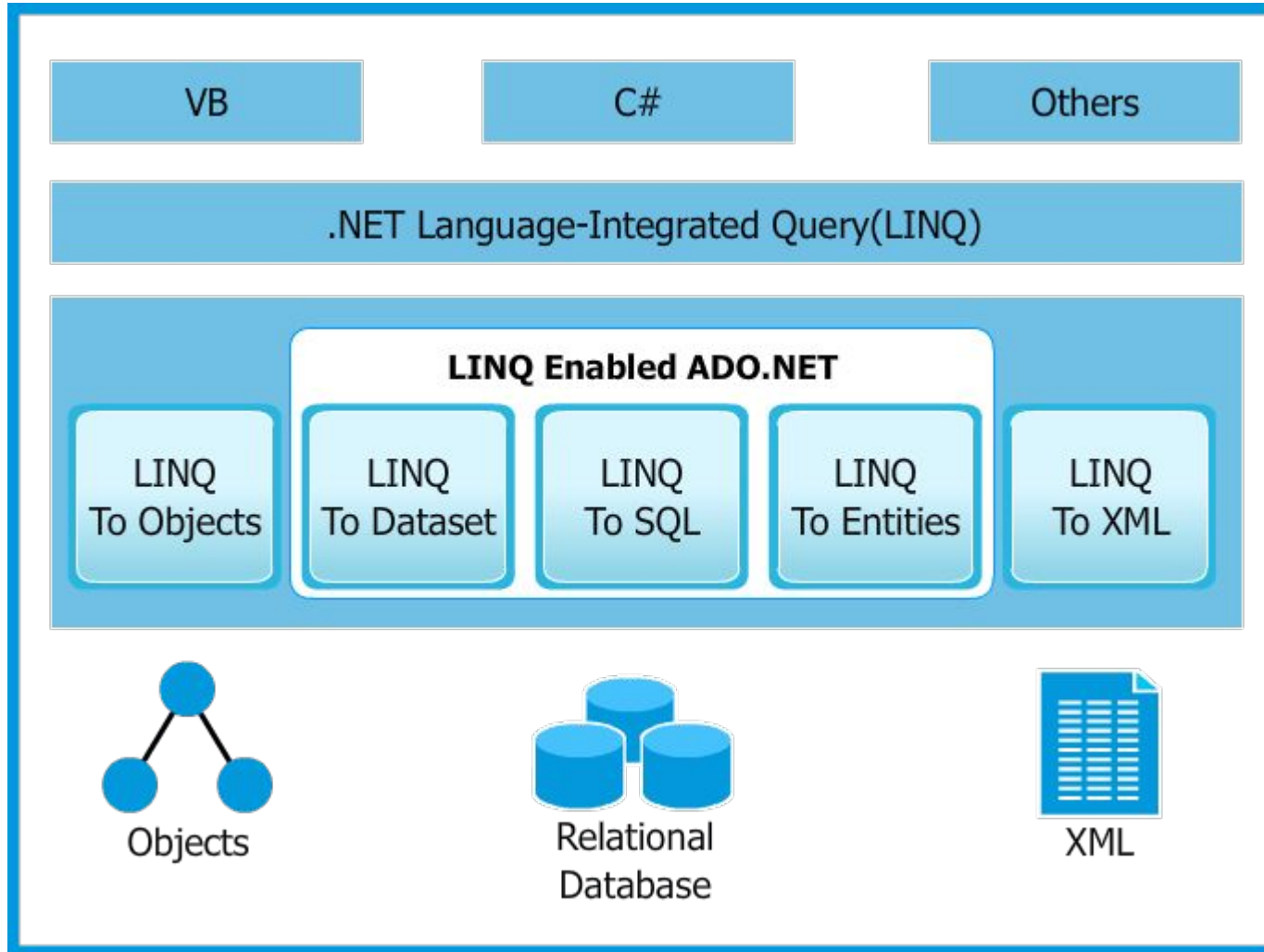
A screenshot of a Windows command prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.exe'. The window contains two lines of text: 'X là số âm' and 'Press any key to continue . . .', with a cursor at the end of the second line.

Ngôn ngữ LINQ

- LINQ (Language Integrated Query) là thư viện mở rộng cho một số các ngôn ngữ lập trình trong .NET Framework như C#, Visual Basic.NET. Nó cung cấp khả năng truy vấn trực tiếp dữ liệu Object, Database, XML...

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 Provider



Common Query

Working with Objects

Less Coding

Easy to Understand

Compile Time Safety

IntelliSense Support

Query Expression

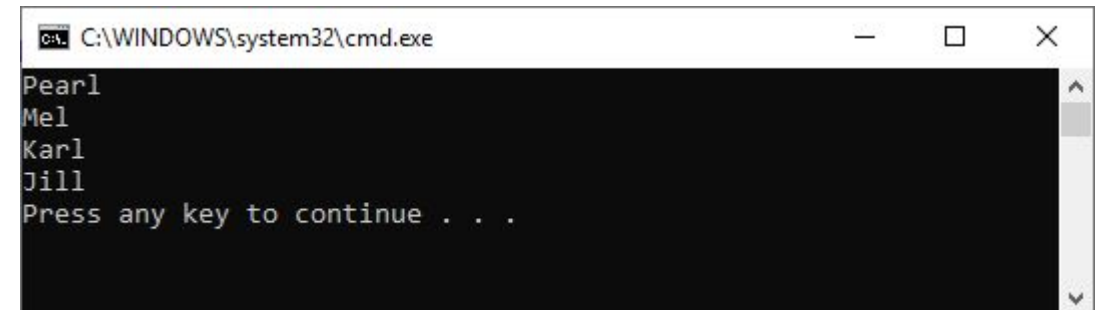
- Query Expression là một truy vấn được viết theo cú pháp truy vấn bằng các mệnh đề như from, select, where, v.v. Các mệnh đề này là một phần ngôn ngữ LINQ.

Mệnh đề	Mô tả
from	Xác định nguồn dữ liệu và biến duyệt dữ liệu
where	Chỉ là biểu thức lọc dữ liệu với các toán tử && or
select	Chọn ra dữ liệu cần lấy
group	Nhóm tập kết quả theo khóa
orderby	Sắp xếp
ascending	Tăng dần
descending	Giảm dần

Query Expression

```
class Program
{
    static void Main(string[] args)
    {
        //Khai báo tập dữ liệu dạng mảng
        string[] names = { "Hanna", "Jim", "Pearl", "Mel", "Jill", "Peter", "Karl", "Abby", "Benjamin" };
        //sử dụng câu truy vấn lấy ra tất cả các tên kết thúc là ký tự l
        IEnumerable<string> words = from word in names
                                    where word.EndsWith("l")
                                    orderby word descending
                                    select word;

        //in kết quả
        foreach (string s in words)
            Console.WriteLine(s);
    }
}
```



```
C:\WINDOWS\system32\cmd.exe
Pearl
Mel
Karl
Jill
Press any key to continue . . .
```

Lambda kết hợp với LINQ Methods

```
class Program
{
    static void Main(string[] args)
    {
        //khởi tạo tập số
        int[] numbers = { 4, 5, 8, 3, 1, 67, 89, 45, 34, 24, 56, 92 };
        //lấy các số lẻ và sắp xếp giảm dần
        var odds = numbers.Where(x => x % 2 != 0).OrderByDescending(n=>n);
        //in kết quả
        foreach (var o in odds)
        {
            Console.WriteLine(o);
        }
    }
}
```



```
C:\WINDOWS\system32\cmd.exe
67
45
5
3
1
Press any key to continue . . .
```

Phương thức mở rộng (Extension method)

Bổ sung thêm tính năng mới vào các lớp đã tồn tại mà không cần chỉnh sửa code gốc

Extension method là phương thức tĩnh được tạo trong một lớp tĩnh

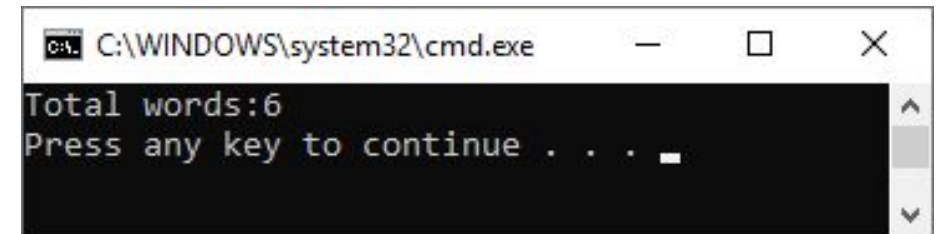
Extension method phải có 1 tham số đầu tiên khai báo với từ khóa this

Tham số đầu tiên của phương thức xác định kiểu mà phương thức sẽ được gọi

Đối tượng gọi phương thức sẽ tự động truyền cho tham số đầu tiên của phương thức

Phương thức mở rộng (Extension method)

```
class Program
{
    static void Main(string[] args)
    {
        //khai báo biến chuỗi
        string st = "Everything you can imagine is real";
        //gọi phương thức mở rộng
        Console.WriteLine("Total words:" + st.CountWord());
    }
}
//tạo lớp tĩnh
static class ExtendString
{
    //định nghĩa phương thức mở rộng đếm số từ cho lớp string
    public static int CountWord(this string str)
    {
        return str.Split(new char[] { ' ' }).Length;
    }
}
```

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.exe'. The window contains two lines of text: 'Total words:6' and 'Press any key to continue . . .'. The cursor is positioned at the end of the second line.

```
C:\WINDOWS\system32\cmd.exe
Total words:6
Press any key to continue . . .
```

Lớp thành phần (Partial Type)

- Lớp thành phần là lớp được tạo ra với nhiều phần, mỗi phần nằm ở 1 tệp tin khác nhau.
- Lớp thành phần được khai báo với từ khóa `partial` và các phần của lớp phải có cùng tên và `access_modifier`.

FILE1: EntityStudent1.cs

```
public partial class Student
{
    //code
}
```

FILE2: EntityStudent2.cs

```
public partial class Student
{
    //code
}
```

Lớp thành phần (Partial Type)

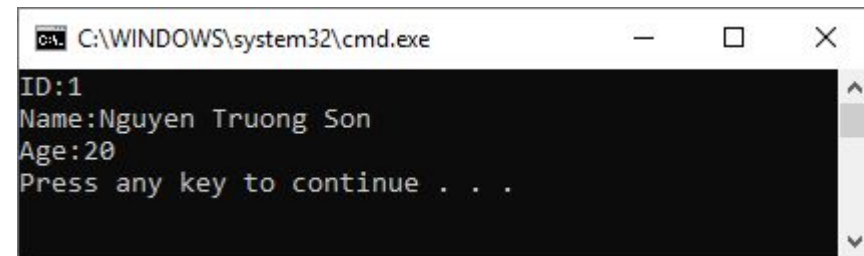
File Student1.cs

```
partial class Student
{
    public int StudentId { get; set; }
    public string StudentName { get; set; }
    public int Age { get; set; }
}

class Program
{
    static void Main(string[] args)
    {
        Student st = new Student() { StudentId = 1, StudentName = "Nguyen Truong Son", Age = 20 };
        st.Display();
    }
}
```

File Student2.cs

```
partial class Student
{
    public void Display()
    {
        Console.WriteLine("ID:" + StudentId);
        Console.WriteLine("Name:" + StudentName);
        Console.WriteLine("Age:" + Age);
    }
}
```



A screenshot of a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The window has a black background with white text. The output displayed is: "ID:1", "Name:Nguyen Truong Son", "Age:20", and "Press any key to continue . . .". The cursor is positioned at the end of the last line.

Kiểu Nullable

- Trước đây chỉ có kiểu tham chiếu mới lưu được giá trị null, từ phiên bản .NET 2.0 trở đi chúng ta có thể sử dụng kiểu Nullable để cho phép các kiểu giá trị có thể lưu trữ được giá trị null.
- Có 2 cách khai báo

```
Nullable<T> var_name;
```

```
Primitive_type? var_name;
```

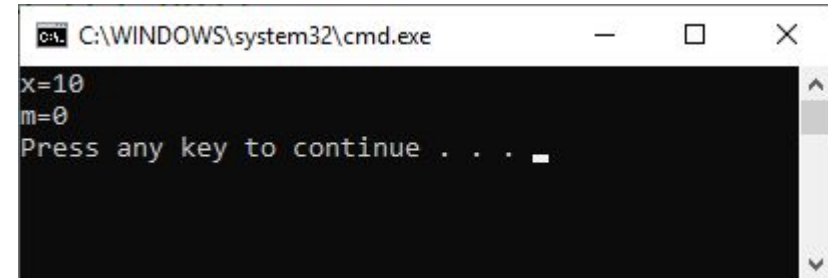
Thuộc tính của kiểu Nullable

HasValue	Kiểm tra xem biến đó có giá trị không?
Value	Lấy giá trị lưu trong biến

Toán tử **??**: xác định xem biến có null không? Nếu null thì trả về giá trị xác định.

Kiểu Nullable

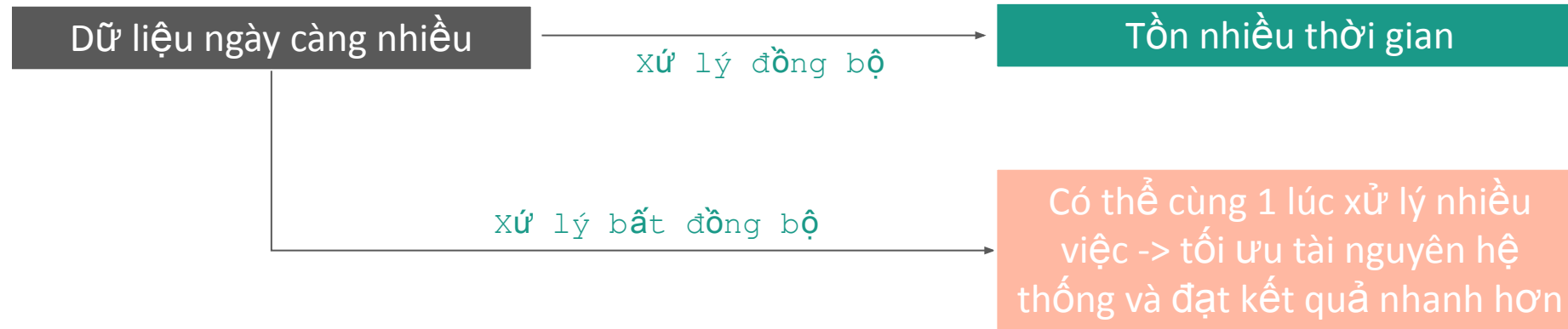
```
class Program
{
    static void Main(string[] args)
    {
        //float n = null; //dòng này báo lỗi do kiểu primitive không cho lưu giá trị null
        //khai báo kiểu primitive có thể chứa giá trị null
        Nullable<int> x = null; //c1
        double? m = null; //c2
        x = 10; //gán x=10
        //kiểm tra xem x có giá trị không?
        if(x.HasValue)
            Console.WriteLine("x=" + x);
        else
            Console.WriteLine("x is null");
        //kiểm tra nếu m is null thì gán m=0
        m = m ?? 0;
        Console.WriteLine("m=" + m);
    }
}
```



The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The output of the program is displayed as follows:

```
x=10
m=0
Press any key to continue . . .
```

Phương thức bất đồng bộ (Asynchronous method)



Kỹ thuật xử lý bất đồng bộ

Thread
Task
Delegate
Await
Async

Phương thức bất đồng bộ (Asynchronous method)

```
class ComplexTask
{
    public Task<int> AnalyzeData()
    {
        Task<int> task = new Task<int>(delegate() {
            /*Dừng thread 3 giây*/
            Thread.Sleep(3000);
            //trả về 1 số ngẫu nhiên 1-1000
            return new Random().Next(1, 1000);
        });
        task.Start();
        return task;
    }
}
```

```
class Program
{
    static async void PerformComputationAsync()
    {
        Console.WriteLine("Entering asynchronous method");
        int result = await new ComplexTask().AnalyzeData();
        Console.WriteLine(result.ToString());
    }
    static void Main(string[] args)
    {
        PerformComputationAsync();
        Console.WriteLine("Main thread executing.");
        Console.ReadLine();
    }
}
```



The screenshot shows a Windows Command Prompt window with the title bar 'C:\WINDOWS\system32\cmd.exe'. The window contains the following text: 'Entering asynchronous method', 'Main thread executing.', and '335'. A cursor is visible on the line following '335'.

HỎI ĐÁP





TRẢI NGHIỆM THỰC HÀNH