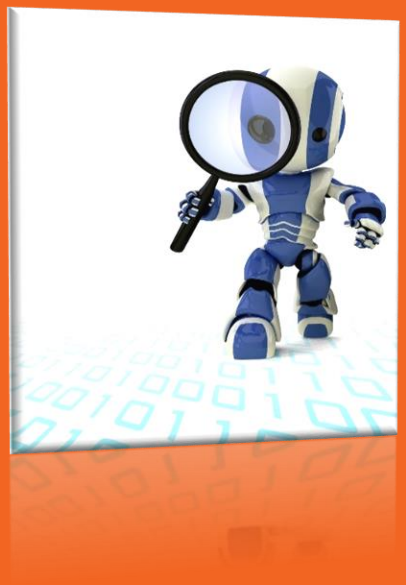
FPT Education

FPT POLYTECHNIC



LẬP TRÌNH C# 2

BÀI 4: LAMBDA EXPRESSIONS AND LINQ

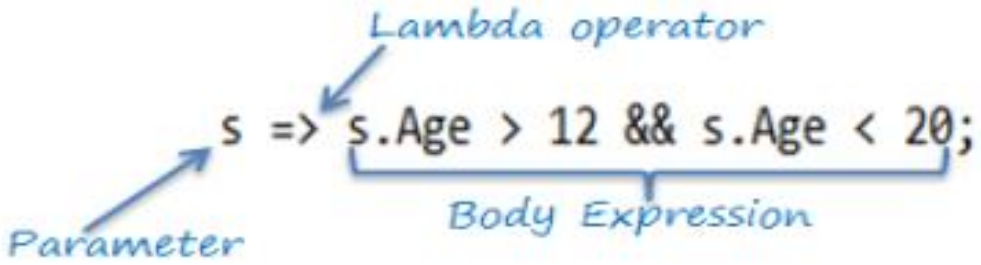
www.poly.edu.vn

Mục tiêu

- Lambda Expressions
- LINQ Queries

LAMBDA EXPRESSIONS

- Lambda expressions là một Anonymous function.
- Ứng dụng tạo các kiểu delegates hay cây biểu thức (expression tree).



Lambda Expression Structure in C#

- Là một cải tiến từ Anonymous function và delegate

LAMBDA EXPRESSIONS

Cú pháp

```
(Input Parameter) => Method Expression
```

- Input parameter: Các tham số đầu vào (như các tham số đầu vào của một hàm)
- Method Expression: Biểu thức hoặc khối lệnh xử lý (như khối lệnh xử lý thân hàm)

- Ví dụ xuất các giá trị trong mảng
  - Dùng Anonymous function

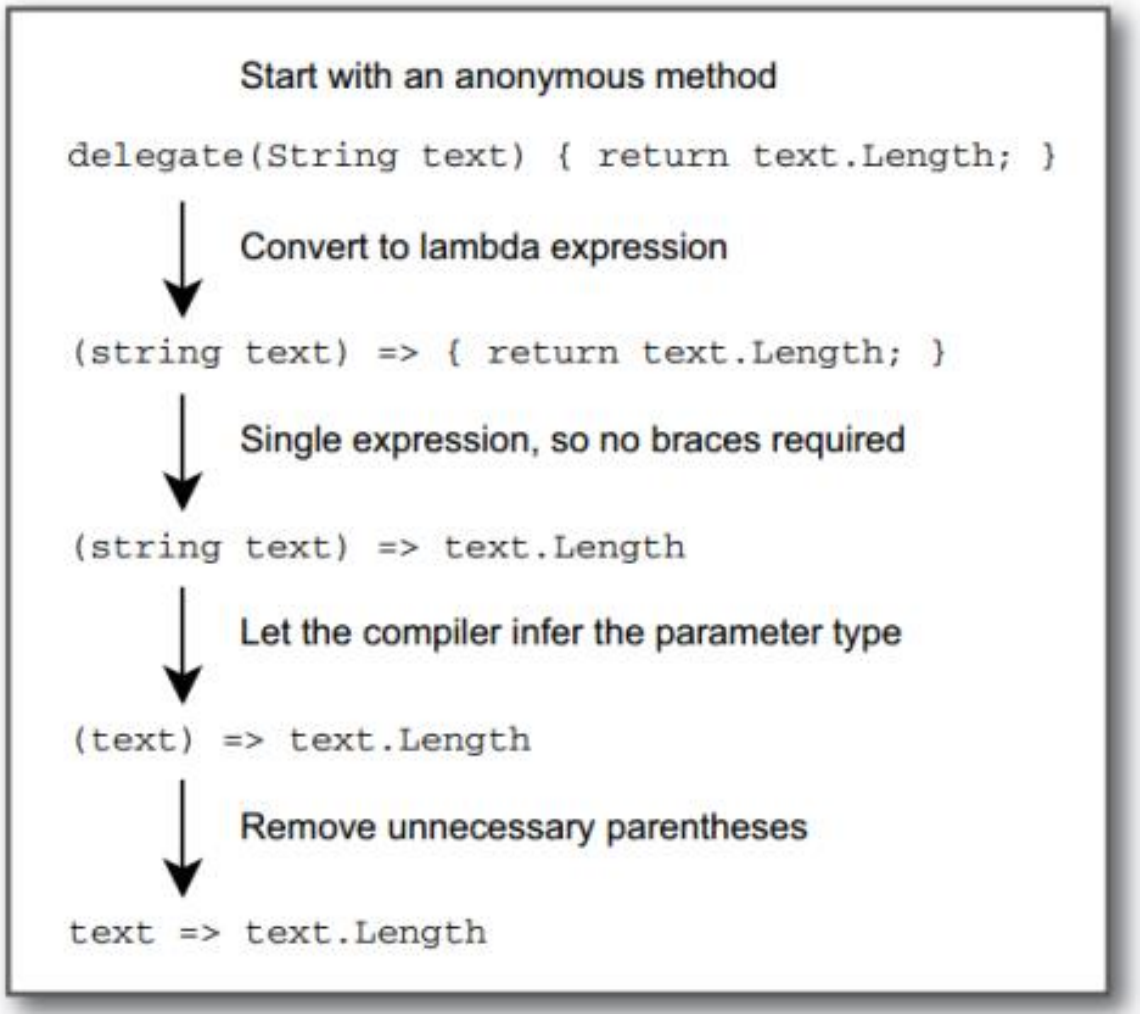
```
static void Main(string[] args)
{
    int[] numbers = { 10, 4, 3, 2, 8, 6, 5, 7, 9, 1 };
    Array.ForEach(numbers, delegate (int x) {
        Console.WriteLine(x);
    });
}
```

  - Dùng Lambda expressions

```
static void Main(string[] args)
{
    int[] numbers = { 10, 4, 3, 2, 8, 6, 5, 7, 9, 1 };
    Array.ForEach(numbers, (int x) => { Console.WriteLine(x); });
}
```

LAMBDA EXPRESSIONS

- Một số qui tắc viết lambda expression:



LAMBDA EXPRESSIONS

- Một số qui tắc viết lambda expression:

```
//1.Có thể bỏ qua kiểu dữ liệu của parameter truyền vào
(string qua) => {Console.WriteLine("Tặng quà: " + qua);}
(qua) => {Console.WriteLine("Tặng quà: " + qua);}

//2. Nếu không có parameter, bỏ dấu () trống
() => {Console.WriteLine("Hello");}

//3. Nếu chỉ có 1 parameter, có thể bỏ luôn dấu ()
(x) => {Console.WriteLine("Hello " + x);}
x => {Console.WriteLine("Hello " + x);}

//4. Nếu có nhiều parameter, ngăn cách bằng dấu phẩy
(x, y) => {Console.WriteLine("Hello " + x + y);}

//5.Nếu anonymous function chỉ có 1 câu lệnh,có thể bỏ dấu {}
x => { Console.WriteLine("Hello " + x); }
x => Console.WriteLine("Hello " + x)

//6. Nếu chỉ return 1 giá trị, có thể bỏ chữ return.
//4 lambda expression sau tương đương nhau
(x) => { return x > 4; }
x => { return x > 4; }
x => return x > 4
x => x > 4
```

LAMBDA EXPRESSIONS

- Quá trình từ Anonymous function đến lambda expression:

```
delegate(Student s) { return s.Age > 12 && s.Age < 20; };
```

1 - Remove Delegate and Parameter Type and add lamda operator =>

```
delegate(Student s)=> { return s.Age > 12 && s.Age < 20; };
```

```
(s) => { return s.Age > 12 && s.Age < 20; };
```

```
(s) => { return s.Age > 12 && s.Age < 20; };
```

Lambda operator

Parameter

Body Expression

2 - Remove curly bracket, return and semicolon

```
(s) => s.Age > 12 && s.Age < 20;
```

3 - Remove Parenthesis around parameter if there is only one parameter

```
s => s.Age > 12 && s.Age < 20;
```

LAMBDA EXPRESSIONS

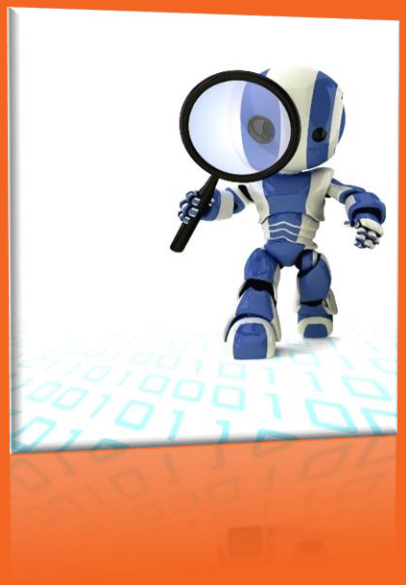
- Cho biết kết quả:
  - x=>x+3 if x=1 ? 4
  - x=> x==5: if x=5 True
  - (x,y)=> x+y if x=2, y=4 6
  - (x,y)=>x==y if x=2, y=4 False

DEMO

- Hiện thực các ví dụ

FPT Education

FPT POLYTECHNIC



LẬP TRÌNH C# 2

BÀI 4: LAMBDA EXPRESSIONS AND LINQ (P2)

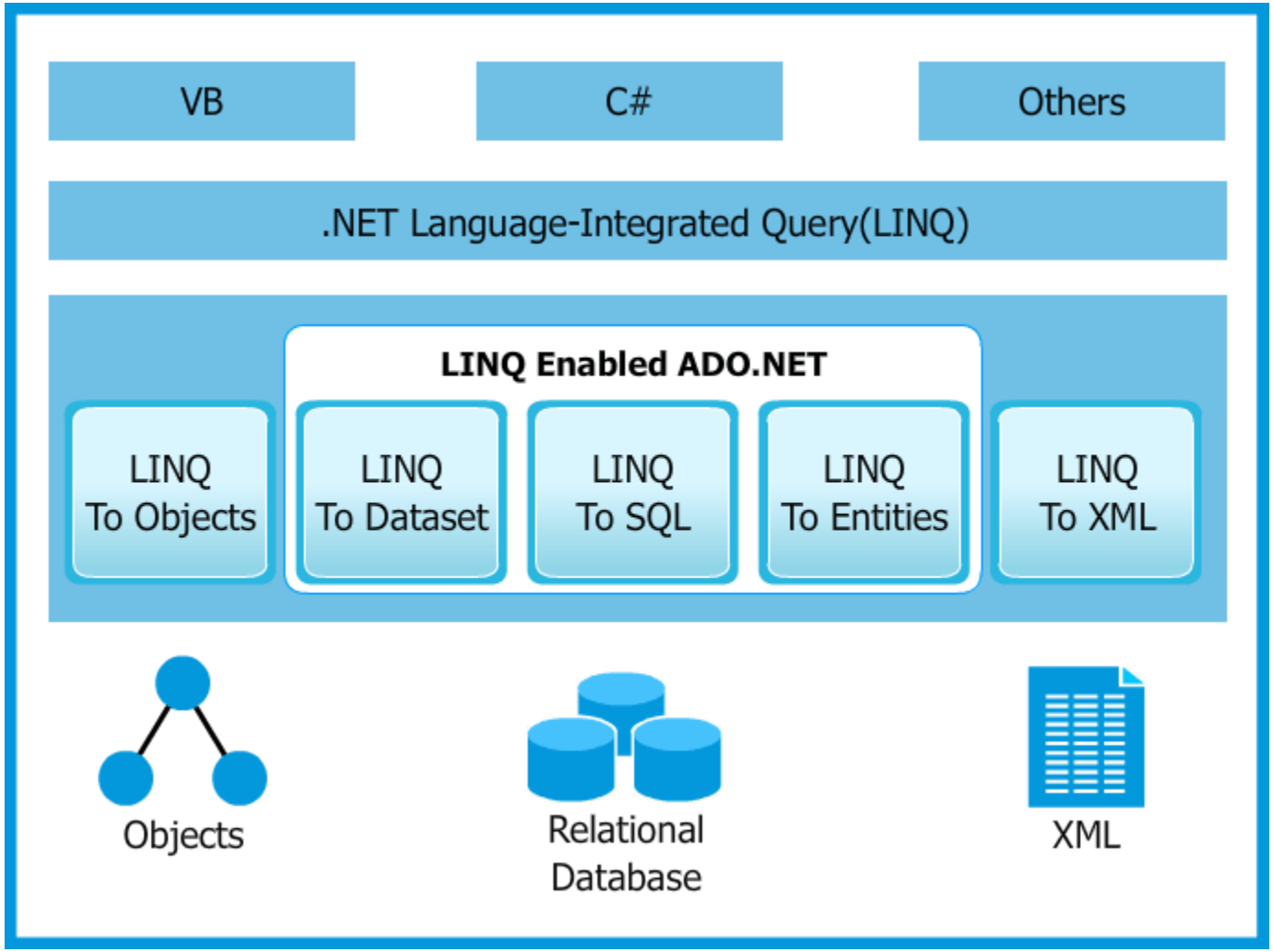
www.poly.edu.vn

LINQ Queries

- LINQ
  - Language Integrated Query
  - Ngôn ngữ truy vấn tích hợp - nó tích hợp cú pháp truy vấn (gần giống các câu lệnh SQL) vào bên trong ngôn ngữ lập trình C#, cho nó khả năng truy cập các nguồn dữ liệu khác nhau.
  - Được giới thiệu từ .Net 3.5 và hỗ trợ nhiều datasource khác nhau SQL Database, XML Documents, ADO.Net Dataset and .NET Collections….

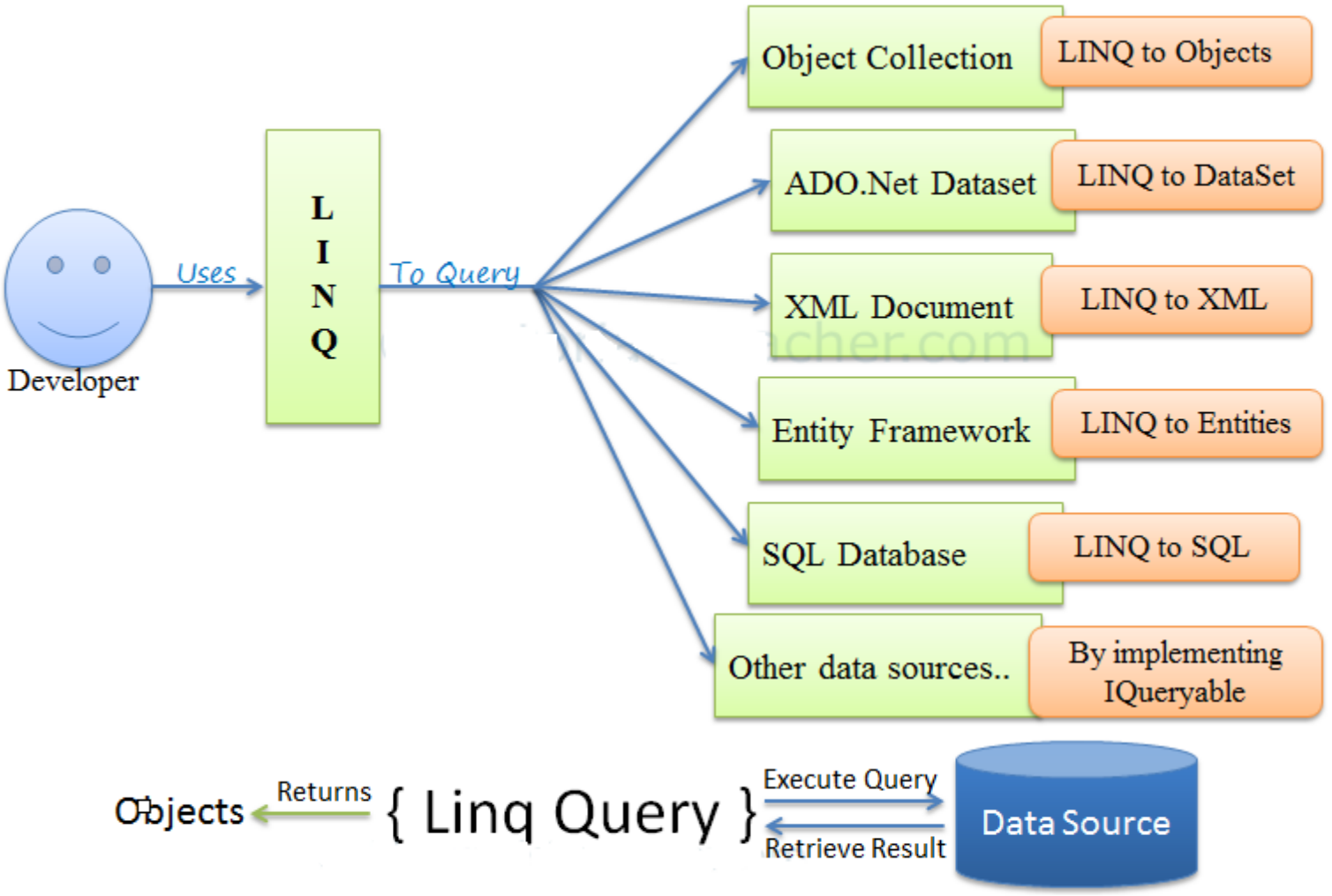
LINQ Queries

LINQ



LINQ Queries

LINQ



Ví dụ LINQ Query to Array

```
// Data source
string[] names = {"Bill", "Steve", "James", "Mohan" };

// LINQ Query
var myLinqQuery = from name in names
                  where name.Contains('a')
                  select name;

// Query execution
foreach(var name in myLinqQuery)
    Console.Write(name + " ");
```

Ưu nhược điểm của LinQ

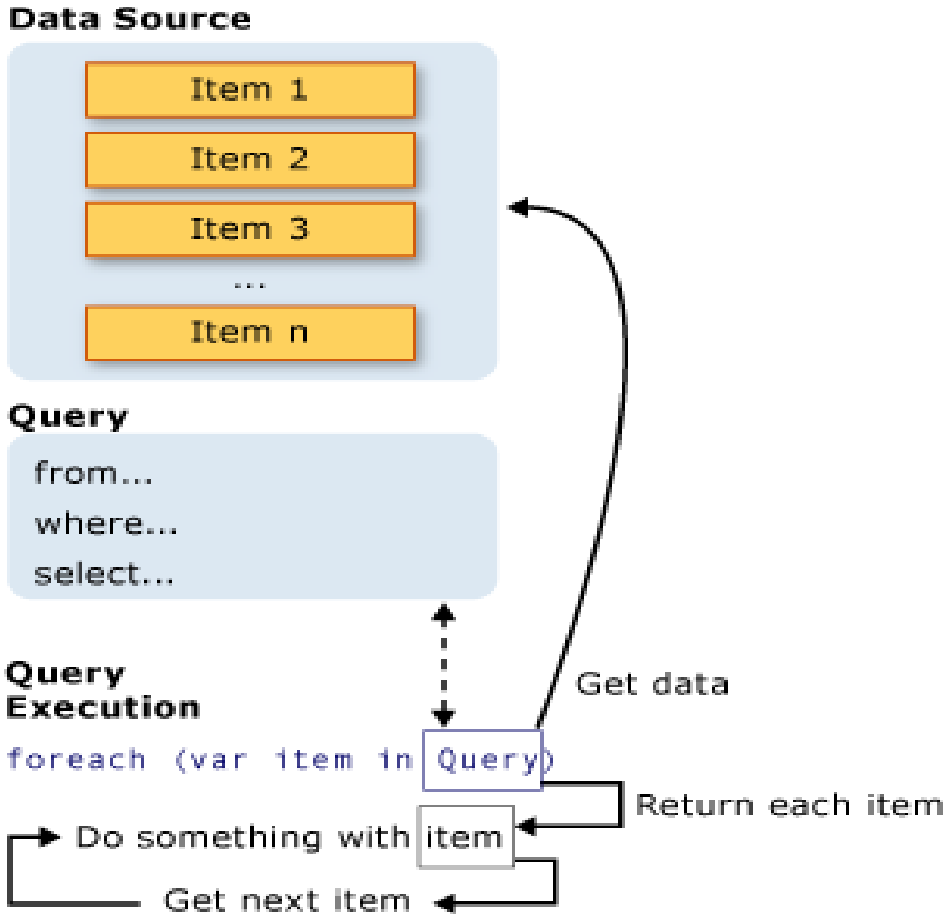
Ưu điểm

- Ưu điểm lớn nhất của Linq đó chính là sự mạch lạc trong code, xây dựng nhanh, ít gây lỗi
- Linq cung cấp nhiều phương thức trong truy vấn dữ liệu, nếu cùng một chức năng, khi sử dụng truy vấn thuần có thể phải code nhiều gấp 2, 3 lần khi sử dụng linq (tùy ứng dụng)
- Cách tiếp cận khai báo giúp truy vấn dễ hiểu và gọn hơn

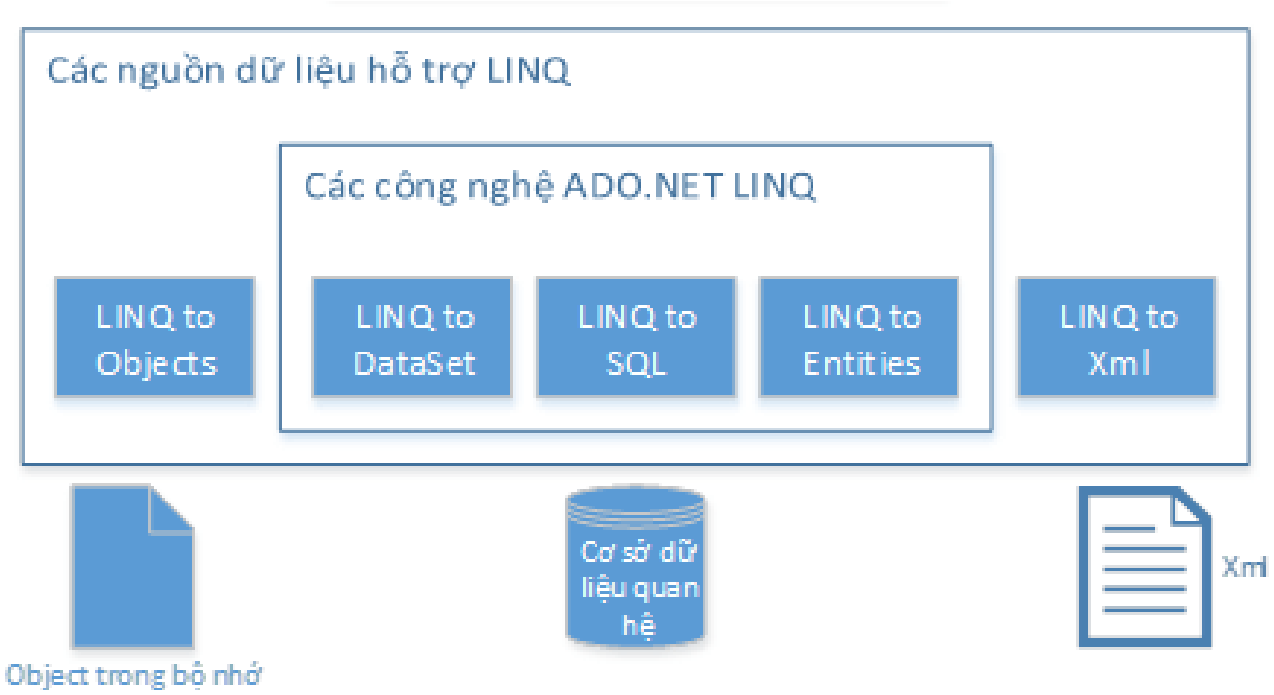
Nhược điểm

- Kém linh hoạt hơn so với truy vấn thuần
- Tốc độ chậm nếu viết linq không khéo

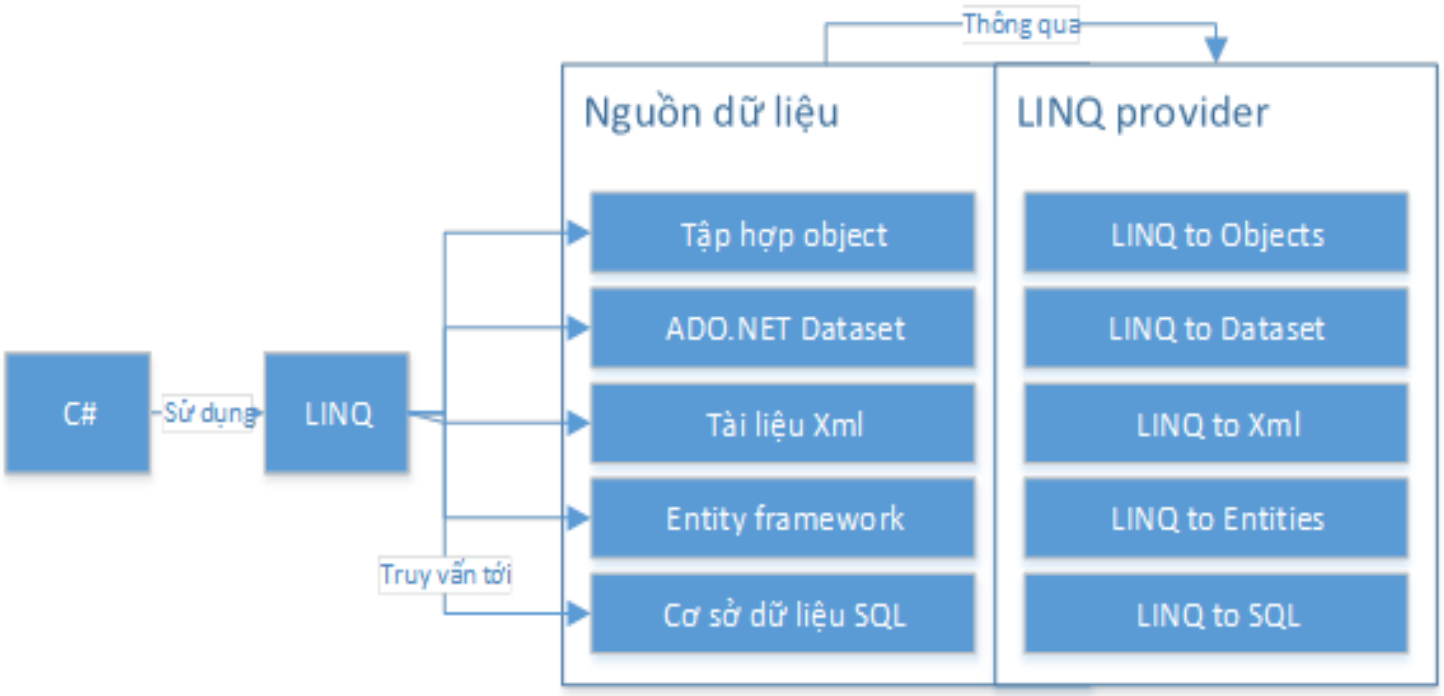
- Để sử dụng LINQ cần có ba thành phần: nguồn dữ liệu (data source), truy vấn (query), lời gọi thực hiện truy vấn (query execution)



- Nguồn dữ liệu
  - Các truy vấn LINQ trong C# đều tác động trên nguồn dữ liệu



- Nguồn dữ liệu
  - LINQ provider giúp chuyển đổi dữ liệu về dạng object và ngược lại



LINQ QUERIES

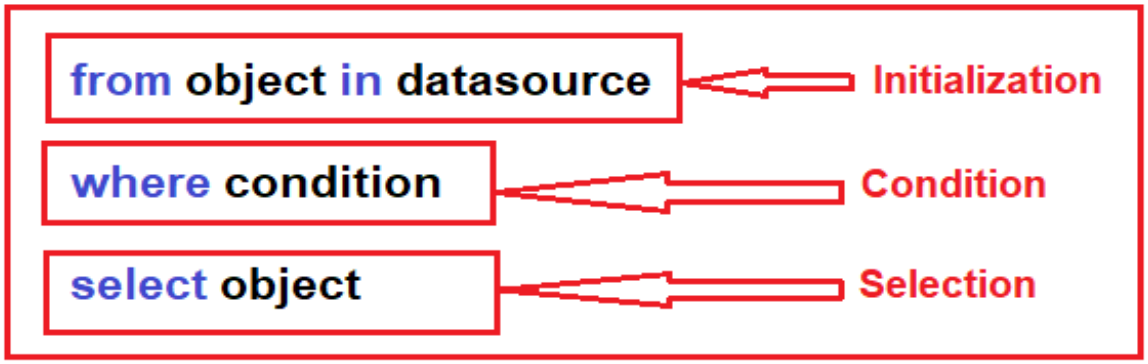
- Nguồn dữ liệu IEnumerable
  - IEnumerable nằm trong namespace System.Collections
  - IEnumerable có thể duyệt các phần tử chỉ 1 chiều tiến lên, nó không thể duyệt ngược lại giữa các phần tử
  - IEnumerable phù hợp với Linq to Object và Linq to XML
  - IEnumerable không hỗ trợ lazy loading vì thế không phù hợp với trường hợp phân trang

LINQ QUERIES

- Nguồn dữ liệu IQueryable
  - IQueryable nằm trong namespace System.Linq
  - IQueryable cũng chỉ có thể di chuyển 1 chiều tiến lên trong collection, nó không thể move back lại
  - Khi truy vấn, IQueryable thực thi câu lệnh truy vấn và lọc dữ liệu trên Server luôn
  - IQueryable phù hợp cho Linq to SQL
  - IQueryable hỗ trợ custom query sử dụng phương thức CreateQuery và Execute.

- LINQ Query Syntax
  - Câu truy vấn mô tả cách thông tin được rút trích, sắp xếp,gom nhóm… từ một nguồn hay các nguồn dữ liệu.
  - Mỗi câu query là sự kết hợp của các thành phần Initialization (làm việc với nguồn dữ liệu), Condition(where, filter, sorting condition), Selection (single selection, group selection or joining)
  - Linq có 3 cách để viết query
    - Query Syntax
    - Method Syntax
    - Mixed Syntax

LINQ Query Syntax



- Mệnh đề from để xác định nguồn dữ liệu mà truy vấn sẽ thực hiện, là những phần tử trong những biến kiểu lớp triển khai giao diện IEnumerable ví dụ mảng, List…
- Một câu truy vấn phải kết thúc bằng mệnh đề select hoặc group và xác định kiểu dữ liệu kết quả của câu truy vấn
- Mệnh đề where để lọc dữ liệu, sau nó là các biểu thức logic xác định các phần tử lọc ra

LINQ Query Syntax

```
//Data Source
List<int> integerList = new List<int>()
{
    1, 2, 3, 4, 5, 6, 7, 8, 9, 10
};
//LINQ Query using Query Syntax
var QuerySyntax = from obj in integerList
                  where obj > 5
                  select obj;

//Execution
foreach (var item in QuerySyntax)
{
    Console.Write(item + " ");
}
```

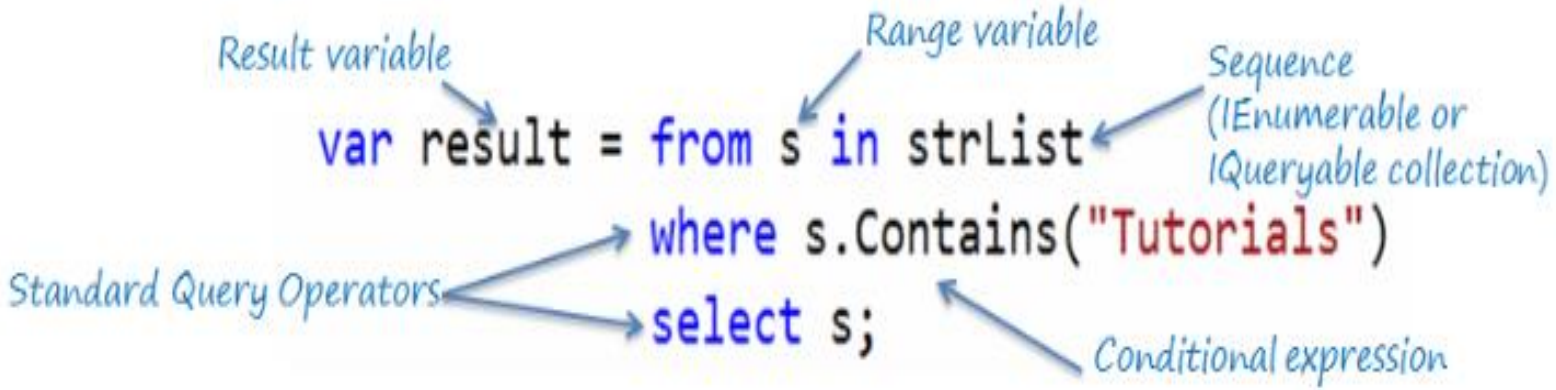
Initialization

Condition

Selection

LINQ Query Syntax

```
// string collection
IList<string> stringList = new List<string>() {
    "C# Tutorials",
    "VB.NET Tutorials",
    "Learn C++",
    "MVC Tutorials" ,
    "Java"
};
```



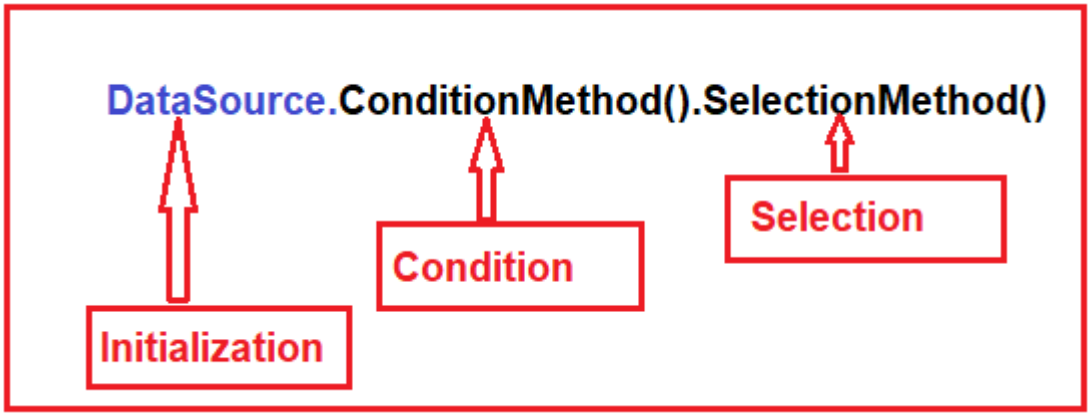
LINQ Query Syntax

- Bổ sung phần còn thiếu vào dấu chấm hỏi để hoàn thiện đoan code bên dưới:

```
static void Main(string[] args)
{
    int[] Num = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
    IEnumerable<int> result = ? numbers in Num
                              ? numbers > 3
                              ? numbers;
}
```

```
static void Main(string[] args)
{
    int[] Num = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
    IEnumerable<int> result = from numbers in Num
                              where numbers > 3
                              select numbers;
}
```

- LINQ Method Syntax



- Sử dụng lambda expression để lọc và hiển thị kết quả từ nguồn dữ liệu

```
int[] Num = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
//LINQ Method Syntax to Print Numbers Greater than 3
IEnumerable<int> result = Num.Where(n => n > 3).ToList();
```

LINQ Method Syntax

```
// string collection
IList<string> stringList = new List<string>() {
    "C# Tutorials",
    "VB.NET Tutorials",
    "Learn C++",
    "MVC Tutorials" ,
    "Java"
};
```

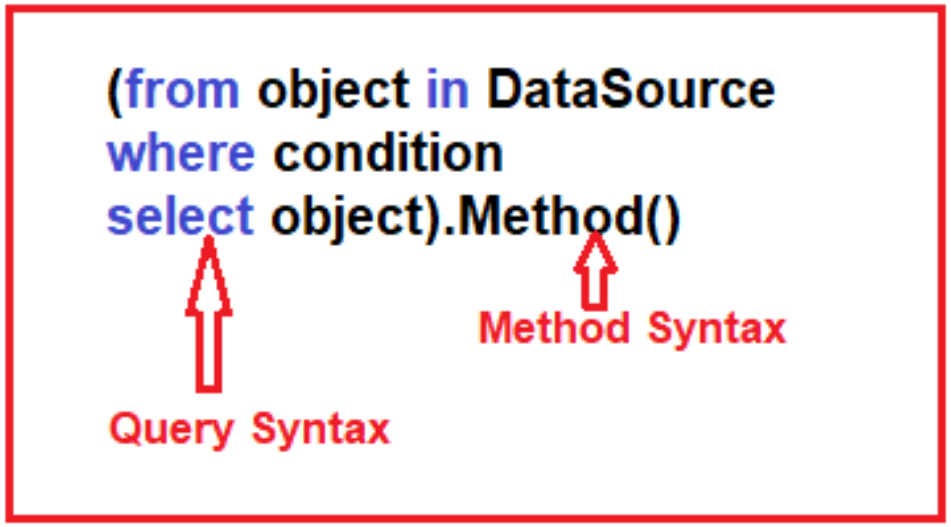
```
var result = strList.Where(s => s.Contains("Tutorials"));
```

Extension method

Lambda expression

LINQ Mixed Syntax

- LINQ Mixed Syntax là sự kết hợp sử dụng Query và MethodSyntax



LINQ Mixed Syntax

- LINQ Mixed Syntax là sự kết hợp sử dụng Query và MethodSyntax

```
static void Main(string[] args)
{
    //Data Source
    List<int> integerList = new List<int>()
    {
        1, 2, 3, 4, 5, 6, 7, 8, 9, 10
    };
    //LINQ Query using Mixed Syntax
    var MethodSyntax = (from obj in integerList where obj > 5 select obj).Sum();
    //Execution
    Console.Write("Sum Is : " + MethodSyntax);
    Console.ReadKey();
}
```

DEMO

- Hiện thực các ví dụ

Tổng kết bài học

- Lambda Expressions
- LINQ Queries

KẾT THÚC