LINQ Operators And Lambda Expression - Syntax & Examples

ASK A QUESTION

Manikavelu Velayutham Sep 08 2010 Article

4 9 335k

HTML clipboard

LINQ is a cool feature in C# 3.0. Most of the developers are struggling for the syntax and examples. Here I have collected various examples for each operator in LINQ and the equivalent Lambda Expressions.

Where

```
IEnumerable<Product> x = products.Where(p => p.UnitPrice >= 10);

IEnumerable<Product> x =
from p in products
where p.UnitPrice >= 10
select p;
```

Select

```
IEnumerable<string> productNames = products.Select(p => p.Name);
IEnumerable<string> productNames = from p in products select p.Name;

var namesAndPrices =
products.
Where(p => p.UnitPrice >= 10).
Select(p => new { p.Name, p.UnitPrice }).
ToList();
IEnumerable<int> indices =
products.
Select((product, index) => new { product, index }).
Where(x => x.product.UnitPrice >= 10).
Select(x => x.index);
```

SelectMany

```
IEnumerable<Order> orders =
customers.
Where(c => c.Country == "Denmark").
SelectMany(c => c.Orders);
var namesAndOrderIDs =
customers.
Where(c => c.Country == "Denmark").
SelectMany(c => c.Orders).
Where(o => o.OrderDate.Year == 2005).
Select(o => new { o.Customer.Name, o.OrderID });
var namesAndOrderIDs =
customers.
```

```
Select(co => new { co.c.Name, co.o.OrderID ``
var namesAndOrderIDs =
from c in customers
where c.Country == "Denmark"
from o in c.Orders
where o.OrderDate.Year == 2005
select new { c.Name, o.OrderID };
```

Take

```
IEnumerable<Product> MostExpensive10 =products.OrderByDescending(p => p.UnitPrice).Take(10);
```

Skip

```
IEnumerable<Product> AllButMostExpensive10 = products.OrderByDescending(p => p.UnitPrice).
```

TakeWhile SkipWhile

```
s.TakeWhile(p) s.SkipWhile(p)
```

Join

```
var custOrders = customers.
Join(orders, c => c.CustomerID, o => o.CustomerID, (c, o) => new {
    c.Name, o.OrderDate, o.Total
});
var custOrders = from c in customers
join o in orders on c.CustomerID equals o.CustomerID
select new {
    c.Name, o.OrderDate, o.Total
};
```

GroupJoin

```
var custTotalOrders = customers.
GroupJoin(orders, c => c.CustomerID, o => o.CustomerID, (c, co) => new {
    c.Name, TotalOrders = co.Sum(o => o.Total)
});
var custTotalOrders = from c in customers
join o in orders on c.CustomerID equals o.CustomerID into co
select new {
    c.Name, TotalOrders = co.Sum(o => o.Total)
};
var custTotalOrders = from c in customers
join o in orders on c.CustomerID equals o.Custor
select new {
    c.Name, o.OrderDate, o.Total
};
var custTotalOrders = from c in customers
join o in orders on c.CustomerID equals o.Custor
from o in co
select new {
    c.Name, o.OrderDate, o.Total
};
var custTotalOrders = from c in customers
join o in orders on c.CustomerID equals o.CustomerID into co
from o in co.DefaultIfEmpty(emptyOrder)
select new {
```

```
IEnumerable < string > locations = customers.Select(c => c.City).
Concat(customers.Select(c => c.Region)).
Concat(customers.Select(c => c.Country)).
Distinct();
IEnumerable < string > locations = new [] {
    customers.Select(c => c.City),
        customers.Select(c => c.Region),
        customers.Select(c => c.Country),
}.
SelectMany(s => s).
Distinct();
```

OrderBy / ThenBy

```
IEnumerable<Product> orderedProducts1 =
products.
OrderBy(p => p.Category).
ThenByDescending(p => p.UnitPrice).
ThenBy(p => p.Name);
IEnumerable<Product> orderedProducts1 =
from p in products
orderby p.Category, p.UnitPrice descending, p.Name
select p;
IEnumerable<Product> orderedProducts2 =
products.
Where(p => p.Category == "Beverages").
OrderBy(p => p.Name, StringComparer.CurrentCultureIgnoreCase);
IEnumerable<string> orderedProductNames =
products.
Where(p => p.Category == "Beverages").
Select(p => p.Name).
OrderBy(x => x);
```

GroupBy

```
IGrouping<string, Product>> productsByCategory =products.GroupBy(p => p.Category);
IGrouping<string, string>> productNamesByCategory =products.GroupBy(p => p.Category, p => p.Name);
```

Distinct

```
IEnumerable<string> productCategories =products.Select(p => p.Category).Distinct();
```

AsEnumerable

```
Table<Customer> custTable = GetCustomersTable()
var query = custTable.AsEnumerable().Where(c =>
```

ToArray

```
string[] customerCountries =customers.Select(c =
```

ToList

```
List<Customer> customersWithOrdersIn2005 =customers.
Where(c => c.Orders.Any(o => o.OrderDate.Year == 2005)).
ToList();
```

```
Dictionary<int,Order> orders =customers.
SelectMany(c => c.Orders).
Where(o => o.OrderDate.Year == 2005).
ToDictionary(o => o.OrderID);
Dictionary<string,decimal> categoryMaxPrice =products.
GroupBy(p => p.Category).
ToDictionary(g => g.Key, g => g.Group.Max(p => p.UnitPrice));
```

ToLookup

```
Lookup<string,Product> productsByCategory =products.ToLookup(p => p.Category);
IEnumerable<Product> beverages = productsByCategory["Beverage"];
```

OfType

```
List<Person> persons = GetListOfPersons();
IEnumerable<Employee> employees = persons.OfType<Employee>();
```

Cast

```
ArrayList objects = GetOrders();
IEnumerable<Order> ordersIn2005 =
objects.
Cast<Order>().
Where(o => o.OrderDate.Year == 2005);
ArrayList objects = GetOrders();
IEnumerable<Order> ordersIn2005 =
from Order o in objects
where o.OrderDate.Year == 2005
select o;
```

First

```
string phone = "206-555-1212";
Customer c = customers.First(c => c.Phone == phone);
```

Single

```
int id=12345;
Customer c = customers.Single(c => c.CustomerID == id);
```

ElementAt

```
Product thirdMostExpensive = products.OrderByDescending(p => p.UnitPrice).ElementAt(2);
```

Range

```
int[] squares = Enumerable.Range(0, 100).Select
```

Repeat

```
long[] x = Enumerable.Repeat(-1L, 256).ToArray()
```

Empty

Any

```
bool b = products.Any(p => p.UnitPrice >= 100 && p.UnitsInStock == 0);
```

All

```
IEnumerable<string> fullyStockedCategories =products.
GroupBy(p => p.Category).
Where(g => g.Group.All(p => p.UnitsInStock > 0)).
Select(g => g.Key);
```

Count

```
int count = customers.Count(c => c.City == "London");
```

Sum

```
int year = 2005;
var namesAndTotals = customers.
Select(c => new {
    c.Name,
        TotalOrders = c.Orders.
    Where(o => o.OrderDate.Year == year).
    Sum(o => o.Total)
});
```

Min

```
var minPriceByCategory = products.
GroupBy(p => p.Category).
Select(g => new {
    Category = g.Key,
        MinPrice = g.Group.Min(p => p.UnitPrice)
});
```

Max

```
decimal largestOrder = customers.
SelectMany(c => c.Orders).
Where(o => o.OrderDate.Year == 2005).
Max(o => o.Total);
```

Page


```
var averageOrderTotals = customers.
Select(c => new {
    c.Name,
        AverageOrderTotal = c.Orders.Average(o =
});
```


Aggregate

```
var longestNamesByCategory = products.
GroupBy(p => p.Category).
Select(g => new {
    Category = g.Key,
        LongestName = g.Group.
    Select(p => p.Name).
    Aggregate((s, t) => t.Length > s.Length ? t : s)
```

C# 3.0 Lambda Expression LINQ LINQ Oper ASK A QUESTION

Manikavelu Velayutham TOP 500

I have over 11 years of IT industry experience with Microsoft technologies. I hold Masters degree in Computer Science and Applications. I am responsible for providing solution in Service Oriented Architecture and involvi... Read more

http://www.c-sharpcorner.com/members/manikavelu-velayutham

196 1.6m 1

4 9

Type your comment here and press Enter Key (Minimum 18 characters)

Nice article, Plz share some basic step by step query writing in linq for CRUD operation.Thanks in advance.

Raghav P Sep 01, 2017

1502 1 0 0 0 Reply

Very nice for article of Manikavelu Velayutham. Hope the features, me try more articles...!

Truong Minh Tuan Jul 26, 2017

1502 1 0 0 0 Reply

good one thannks

Sonu Chaudhary Jun 07, 2016

590 1.6k 101.1k 0 0 Reply

Very Nice...

Mukesh Kumar Tiwari Dec 10, 2014

872 764 111.2k 0 0 Reply

great work..really helpful for the new developers.Thanks

Satya

1502 1 0

Very good post but not for beginners.

Santosh Yadav

1255 249 68.2k

Very Useful and Handy.....

Siva Kumar ss

1499 4 1.8k 0 0

Nice and practical approach :)

Don't miss Manikavelu Velayutham's next article

Manikavelu Velayutham

196 7.5k 1.6m

+ Follow

C#Corner

ASK A QUESTION

Great, Thanks

Farooq Md

1501 2 0

Oct 25, 2011

0 0 Reply

Comment Using

0 Comments Sort by **Oldest**

Add a comment...

Facebook Comments Plugin

TRENDING UP

01 Getting Started With Angular 5 And ASP.NET Core

02 Angular 5 App With ASP.NET Core 2.0 Web API

03 Server Side Custom Paging In Angular-UI Grid Using Web

04 Angular 5, ASP.NET Core CRUD For Inventory Management Using EF And WEB API

05 Top 10 Web Application Security Risks In 2017

Don't miss Manikavelu Velayutham's next article

Manikavelu Velayutham

196 7.5k 1.6m

+ Follow

07 Angular 5 Basics

08 What Is SQL Operations Studio

09 Audit Made Easy Without Audit Log - Part One

10 Angular 5 Basic Demo Project Overview

ASK A QUESTION

View All

Don't miss Manikavelu Velayutham's next article

Manikavelu Velayutham

196 7.5k 1.6m

+ Follow

Phila

New

Lond

Delhi