using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LinqDemo1

{

public class Product

{

public int ProductId { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public int Price { get; set; }

public string Category { get; set; }

static List<Product> products = new List<Product>() {

new Product(){ProductId=1,Name="Mouse",Description="Wireless",Price=1300,Category="Electronics"},

new Product(){ProductId=2,Name="Keyboard",Description="Wireless",Price=3220,Category="Electronics"},

new Product(){ProductId=3,Name="Laptop",Description="HD and Touch",Price=87000,Category="Electronics"},

new Product(){ProductId=4,Name="Cooling Pad",Description="test",Price=2000,Category="Electronics"},

new Product(){ProductId=5,Name="Monitor",Description="LED",Price=6000,Category="Electronics"},

new Product(){ProductId=6,Name="projector",Description="Wired",Price=1700,Category="Electronics"},

new Product(){ProductId=7,Name="Pencil",Description="Color pencil",Price=170,Category="Stationary"},

new Product(){ProductId=8,Name="Scale",Description="long size",Price=100, Category = "Stationary"},

};

public void Demo()

{

//var productsPriceMorethan5000 = products.Where(p => p.Price >= 5000);

//foreach (var product in productsPriceMorethan5000)

//{

// Console.WriteLine($"{product.ProductId} \t {product.Name}\t {product.Price}");

//}

////Example for order by

//var orderbyProductName = from p in products

// orderby p.Name descending

// select p;

//var orderbyProductName1 = products.OrderByDescending(p => p.Name);

//foreach (var product in orderbyProductName1)

//{

// Console.WriteLine($"{product.Name}");

//}

//var ordebyPriceName = from p in products

// orderby p.Name ascending, p.Price descending

// select new { p.Name, p.Price };

//var ordebyPriceName1 = products.OrderByDescending(p => p.Name)

// .ThenBy(p => p.Price);

//foreach (var product in ordebyPriceName1)

//{

// Console.WriteLine($"{product.Name}\t {product.Price}");

//}

//Example for Group by

//Query Syntax

var groupedProducts = from p in products

group p by p.Category;

var groupedProducts2 =products.GroupBy(p => p.Category);

foreach(var categoryGroup in groupedProducts)

{

Console.WriteLine("Category :"+categoryGroup.Key);

foreach (var product in categoryGroup)

{

Console.WriteLine($"{product.Name}\t {product.Description}\t{product.Price}");

}

}

}

}

}

Program.cs

using System.Collections;

namespace LinqDemo1

{

internal class Program

{

static void Main(string[] args)

{

//int[] numbers = [2, 43, 657, 68, 345, 634];

//Console.WriteLine("\nEven Numbers using Query syntax");

////Query syntax

//var evenNumbers=from n in numbers

// where n%2==0

// select n;

//foreach (var item in evenNumbers)

//{

// Console.WriteLine(item);

//}

//Console.WriteLine("\nEven Numbers using Method syntax");

////Method Syntax

//var evenNumbers1=numbers.Where(n => n%2==0);

//foreach (var item in evenNumbers1)

//{ Console.WriteLine(item); }

//IList mixedList = new ArrayList();

//mixedList.Add(10);

//mixedList.Add(20);

//mixedList.Add("Fransy");

//mixedList.Add("Test");

//mixedList.Add(new Product()

//{ ProductId = 1, Name = "Test Product", Description = "Test DEscription", Price = 234 });

////Example for OfType operator

//var intList = from i in mixedList.OfType<int>()

// select i;

////method syntax

//var intList1 = mixedList.OfType<int>();

//var stringList = from s in mixedList.OfType<string>()

// select s;

////method syntax

//var stringList1 = mixedList.OfType<string>();

//var productList = from p in mixedList.OfType<Product>()

// select p;

////method syntax

//var productList1 = mixedList.OfType<Product>();

//Console.WriteLine("\nInteger List filtered from mixedList");

//foreach (var i in intList1)

//{

// Console.WriteLine(i);

//}

//Console.WriteLine("\nString List filtered from mixedList");

//foreach (var i in stringList1)

//{

// Console.WriteLine(i);

//}

//Console.WriteLine("\nProduct List filtered from mixedList");

//foreach (var i in productList1)

//{

// Console.WriteLine($"{i.ProductId}\t{i.Name}\t {i.Description}\t {i.Price}");

//}

Product product = new Product();

product.Demo();

Console.ReadLine();

}

}

}

Example for Joins

public class Supplier

{

public int SupplierId { get; set; }

public string SupplierName { get; set; }

public string Category { get; set; }

static List<Supplier> suppliers = new List<Supplier>()

{

new Supplier(){SupplierId = 1,SupplierName="iball",Category="iball"},

new Supplier(){SupplierId = 2,SupplierName="Hp",Category="Hp"},

new Supplier(){SupplierId = 3,SupplierName="DOMS",Category="Stationary"},

new Supplier(){SupplierId = 3,SupplierName="cooling fan",Category="Gamerz"},

};

Product product = new Product();

List<Product> products = new List<Product>();

public void Demo()

{

products = product.GetAllProducts();

////inner Join

//var result = from p in products

// join s in suppliers on p.Category equals s.Category

// select new

// {

// ProductName = p.Name,

// Supplier = s.SupplierName

// };

//foreach ( var item in result )

//{

// Console.WriteLine($"{item.ProductName} \t {item.Supplier}");

//}

////Example for Left Join

//var leftJoinResult = from s in suppliers

// join p in products on s.Category equals p.Category into prodGroup

// from pg in prodGroup.DefaultIfEmpty()

// select new

// {

// supplier = s.SupplierName,

// product = pg != null ? pg.Name : "No Products"

// };

//foreach (var item in leftJoinResult)

// Console.WriteLine($"{item.supplier} supplies {item.product}");

////Example for Group Join

//var groupjoinResult = from s in suppliers

// join p in products on s.Category equals p.Category into prodGroup

// select new

// {

// supplier = s.SupplierName,

// products = prodGroup

// };

//foreach (var group in groupjoinResult)

//{

// Console.WriteLine($"{group.supplier} supplies. ");

// foreach (var p in group.products)

// {

// Console.WriteLine($"- {p.Name}");

// }

//}

////Example for Anonymous type +Custom Message

//var customJoin = from p in products

// join s in suppliers on p.Category equals s.Category

// where p.Price > 5000

// select new {

// Message =$"{s.SupplierName} provides {p.Name} for {p.Price}"

// };

//foreach (var item in customJoin)

//{

// Console.WriteLine(item.Message);

//}

//Example for filtering by category and price

var filteredResult = from p in products

join s in suppliers on p.Category equals s.Category

where p.Category == "Stationary" && p.Price < 5000

select new

{

Product = p.Name,

Price = p.Price,

Supplier = s.SupplierName

};

foreach (var item in filteredResult)

{

Console.WriteLine($"{item.Product} ({item.Price}) - supplied by {item.Supplier}");

}

}

}

}

Element Operators

//Element operators

IList<int> intList = new List<int>() { 7, 10, 21, 30, 45, 50, 87 };

IList<string> stringList = new List<string>() { null, "One", "Three", "Two", "Five", "Four" };

IList<string> emptyList = new List<string>();

//Example for First()

Console.WriteLine($"First Element in the List {intList.First()}");

Console.WriteLine($"1st Even Element in the List{intList.First(i =>i%2==0)}");

Console.WriteLine($"First Element in the string List {stringList.First()}");

// Console.WriteLine($"1st Element in the emptyList{emptyList.First()}");

//Example for FirstOrDefault

Console.WriteLine($"First Element in the List {intList.FirstOrDefault()}");

Console.WriteLine($"1st Even Element in the List{intList.FirstOrDefault(i => i % 2 == 0)}");

Console.WriteLine($"First Element in the string List {stringList.FirstOrDefault()}");

Console.WriteLine($"1st Element in the emptyList{emptyList.FirstOrDefault()}");

//Example for Last()

Console.WriteLine($"Last Element in the List {intList.Last()}");

Console.WriteLine($"last Even Element in the List{intList.Last(i => i % 2 == 0)}");

Console.WriteLine($"Last Element in the string List {stringList.Last()}");

var firstExpensiveProduct = products.FirstOrDefault(p => p.Price > 10000);

var firstCheapProduct = products.First(p => p.Price < 500);

Console.WriteLine($"FirstExpensiveProduct= {firstExpensiveProduct.Name}");

Console.WriteLine($"First Cheap Product = {firstCheapProduct.Name}");

//single

var onlyProjector = products.Single(p => p.Name == "projector");

var onlyScanner = products.SingleOrDefault(p => p.Name == "Scanner");

Console.WriteLine($"Only Projector = {onlyProjector.Name}");

// Console.WriteLine($"Only Scanner = {onlyScanner.Name}"); //error as scanner is not in the list

//Any and All

bool anyStationary = products.Any(p => p.Category == "Stationary");

bool allConstly = products.All(p => p.Price > 500);

Console.WriteLine($"Any product as stationay = {anyStationary}");

Console.WriteLine($"All products are price is >500 = {allConstly}");

//Aggregate Function

int totalPrice = products.Sum(p => p.Price);

int totalCount=products.Count();

int maxPrice=products.Max(p => p.Price);

int minPrice=products.Min(p => p.Price);

double avgPrice=products.Average(p => p.Price);

Console.WriteLine($"TotalPrice = {totalPrice}");

Console.WriteLine($"Total Product Count= {totalCount}");

Console.WriteLine($"Maximum price= {maxPrice}");

Console.WriteLine($"Minimum Price= {minPrice}");

Console.WriteLine($"Average Price= {avgPrice}");

////Distinct

//var uniqueProducts=products.Select(p=> p.Category).Distinct();

//Console.WriteLine("\n unique Categories (Distinct)\n");

//foreach (var product in uniqueProducts)

//{

// Console.WriteLine($"{product}");

//}

////intersect

//var commonCategories = products.Select(p => p.Category)

// .Intersect(suppliers.Select(s => s.Category));

//Console.WriteLine("\n Common Categories (intersect)\n") ;

//foreach (var item in commonCategories)

//{

// Console.WriteLine($"{item}");

//}

////union

//var allNames = products.Select(p => p.Category).Union(suppliers.Select(s => s.Category));

//Console.WriteLine("All Names from suppliers and products");

//foreach (var item in allNames)

//{

// Console.WriteLine($"{item}");

//}

////except

//var productsOnlyCategories=products.Select(p => p.Category)

// .Except(suppliers.Select(s => s.Category));

//Console.WriteLine(" \n Except \n");

//foreach (var item in productsOnlyCategories)

//{

// Console.WriteLine(item);

//}

//skip & SkipWhile

var skipFirst2 = products.Skip(2);

var skipwhilePriceis6000 = products.SkipWhile(p =>p.Price<1300);

Console.WriteLine("\nAll Products \n");

foreach (var item in products)

{

Console.WriteLine(item.Name);

}

////Console.WriteLine("\nSkip(2) \n");

////foreach (var item in skipFirst2)

////{

//// Console.WriteLine($"{ item.Name}");

////}

//Console.WriteLine("\nSkipwhile (6000) \n");

//foreach (var item in skipwhilePriceis6000)

//{

// Console.WriteLine($"{item.Name}");

//}

//Example for Take

var first2products = products.Take(2);

var takewhileCheapPrice = products.TakeWhile(p => p.Price < 2000);

Console.WriteLine("\nTake (2)\n");

foreach (var item in first2products)

{

Console.WriteLine(item.Name);

}

Console.WriteLine("\nTake (2)\n");

foreach (var item in takewhileCheapPrice)

{

Console.WriteLine(item.Name);

}