

Use LINQ to Select Data Within Collections



Paul D. Sheriff

Business / IT Consultant

psheriff@pdsa.com | www.pdsa.com



Module Goals



An overview of the console application

Selecting all data

Selecting specific columns

Building an anonymous class





The Demo Classes



Sample Console Application

The image shows a Visual Studio interface with a console application project named 'LINQSamples'. The project is configured for .NET 8.0 and is a console application. The Solution Explorer on the right shows the project structure, including a 'LINQSamples' folder with files like 'Dependencies', 'Program.cs', 'Readme.txt', and 'SamplesViewModel.cs'. Below this, there are three sub-folders: 'LINQSamples.Common', 'LINQSamples.Domain', and 'LINQSamples.Infrastructure'. Each folder contains several C# files. The 'LINQSamples.Common' folder contains 'Dependencies', 'ComparerClasses' (with 'ProductComparer.cs' and 'ProductIdComparer.cs'), 'HelperClasses' (with 'LinqHelper.cs' and 'ProductHelper.cs'), and 'BaseViewModel.cs'. The 'LINQSamples.Domain' folder contains 'Dependencies', 'CompositeClasses' (with 'ProductOrder.cs', 'ProductSales.cs', 'ProductStats.cs', and 'SaleProducts.cs'), 'EntityClasses' (with 'Product.cs', 'ProductSimple.cs', and 'SalesOrder.cs'), and 'RepositoryClasses' (with 'ProductRepository.cs', 'ProductSimpleRepository.cs', and 'SalesOrderRepository.cs'). The 'LINQSamples.Infrastructure' folder contains 'Dependencies' and 'ProductRepository.cs'. The Properties window on the left shows the project configuration, including the target framework (.NET 8.0), target OS (None), startup object (Not set), and assembly name.

Using a Console Application

All the demos now use .NET 8

Console app runs each sample

Classes to help us query

Product and Sale Entity Classes.

Classes to Build Collections



Product Entity Class

```
public partial class Product {  
    public int ProductID { get; set; }  
    public string Name ...  
    public string Color ...  
    public decimal StandardCost ...  
    public decimal ListPrice ...  
    public string Size ...  
}
```

◀ Represents a "Product"

◀ Each property would have a { get; set; }

◀ Eliminated here for brevity



Product Repository Class

```
public partial class ProductRepository
{
    public static List<Product> GetAll() {
        return new List<Product> {
            new Product {
                ProductID = 680,
                Name = "HL Road Frame",
                Color = "Black",
                StandardCost = 1059.31M,
                ListPrice = 1431.50M,
                Size = "58",
            },

            // MORE CODE HERE
        }
    }
}
```

- ◀ Class to retrieve collection of product data
- ◀ Method to retrieve all products from a data source
- ◀ Using hard-coded values for this course



View Model Base Class

```
public class ViewModelBase {  
    public void GetProducts() {  
        return ProductRepository.GetAll();  
    }  
  
    public void GetSales() {  
        return SalesOrderRepository.GetAll();  
    }  
  
    public Display*() {  
    }  
}
```

- ◀ **Base class used by SamplesViewModel class**
- ◀ **Method to retrieve set of Product objects**
- ◀ **Method to retrieve set of SalesOrder objects**
- ◀ **Overloaded methods to display Products, SalesOrder objects, as well as many different lists**



SamplesViewModel Class

```
public class SamplesViewModel :  
    ViewModelBase {  
  
    public void GetAllQuery() {  
    }  
  
    public void GetAllMethod() {  
    }  
}
```

- ◀ View model class used to teach LINQ samples
- ◀ One method to illustrate LINQ query syntax
- ◀ One method to illustrate LINQ method syntax



SamplesViewModel Method

```
public List<Product> GetAllQuery()  
{  
    List<Product> products =  
        GetProducts();  
    List<Product> list;  
  
    // Write Query Syntax Here  
    list = (from prod in products  
            select prod).ToList();  
  
    return list;  
}
```

- ◀ Method returns a set of data
- ◀ Build a collection to query
- ◀ Create variable to hold results
- ◀ Write a query using LINQ methods
- ◀ Return the results



Program.cs

```
using LINQSamples;

// Create instance of view model
SamplesViewModel vm = new();

// Call Sample Method
var result = vm.GetAllQuery();

// Display Results
vm.Display(result);
```

- ◀ **Bring in LINQSamples namespace**
- ◀ **Create instance of sample view model class**
- ◀ **Call the sample method you just wrote**
- ◀ **Display results in console**





Selecting



Demo



Select all items using LINQ



Sample Methods

```
public List<Product> GetAllQuery() {  
    List<Product> products =  
        GetProducts();  
  
    List<Product> list = new();  
    ... REST OF CODE HERE  
}
```

```
public List<Product> GetAllMethod() {  
    List<Product> products =  
        GetProducts();  
  
    List<Product> list = new();  
    ... REST OF CODE HERE  
}
```

◀ There are many methods in the **SamplesViewModel** class

◀ Set the 'list' variable to **new()** so all methods can compile

◀ Once you set 'list' to the results of the query, you can eliminate the '**= new()**'



Sample Methods

```
public List<Product> GetAllQuery() {  
    List<Product> products =  
        GetProducts();  
  
    return (from prod in products)  
        .ToList()  
}
```

- ◀ In a real-world scenario, you can eliminate the 'list' variable
- ◀ Simply return the results of the query
- ◀ While training, I will be showing you the results of the 'list' variable in the debugger, so it makes it easier to add the variable in this course



Why Show Query and Method Syntax?

**Some queries must
be done with the
method syntax**

**You may run across
samples or articles
where the author
used one or the
other**

**So, you should know
both methods**



Demo



Get a single column



Demo



Get specific columns



Demo



Build an anonymous class



Module Summary



Query syntax is readable, if a little verbose

Method syntax is very compact

Can select single properties

Can select multiple properties

**Project new columns using anonymous
classes**

Up Next:

Use LINQ to Order Data

