From Zero to LINQ in .NET 6

Do you still find yourself writing a lot of loops, and you can't help but think there must be a better way? Well, you are probably right. LINQ can help you aggregate data, extract data from existing collections, compare data between collections, process XML, select data from a database, and much more. This seminar shows you common, and uncommon, examples where you might have used loops in the past, and how to translate those into LINQ queries. LINQ is very powerful and generally is much faster than using loops, so start using it today.

This course is for anyone who wants to learn to use LINQ syntax in their .NET applications to effectively query collections of data. In this two-day hands-on course, you learn the basics of LINQ such as selecting, ordering, and filtering data. You then move on to finding single elements and extracting data using the Take() and Skip() methods. The new Range operator in .NET 6 is used as well as the new DistinctBy(), MinBy() and MaxBy().

After the basics are explored, you can then start looking at how to answer questions about what is in a collection, look for differences between collections and how to combine collections together. Just like the SQL language, you can also perform joins and subqueries using the LINQ syntax as well as grouping. Finally you explore data aggregation, iterating over collections to perform an operation, and understanding how deferred execution works with LINQ. If time permits, you will be introduced to such technologies as LINQ to XML and using LINQ with the Entity Framework.

Learning Objectives

- Why LINQ is Important
- Selecting, Filtering and Ordering Data
- Taking, Skipping and Getting Distinct Data
- Getting Specific Data using Take, Skip and Distinct
- Union and Join Two or More Collections
- Grouping and Aggregating Data
- Iterating Over Collections
- What is Deferred Execution and Why it is Important

Who Should Attend?

This is a technical VSLive! training seminar, and as such, is geared to developers, data engineers, and architects but can also be helpful to technology product

management such as IT leaders, Chief Data Officers, and Chief Technology Officers. Coding experience is expected, and hands-on labs will be performed using the latest technologies.

Student Prerequisites

This course is designed for programmers who already have some experience with .NET and C#. You should be familiar with object-oriented programming and know how to create a console application using either Visual Studio or VS Code.

Hardware Requirements

A computer must be supplied by the student with the following technologies installed.

- Visual Studio 2022 or later
- or Visual Studio Code V1.5 or later
- .NET 6.x or later
- SQL Server (Developer Edition or higher) or SQL Server Express

Course Level

Introduction

Course Length

2 days

Module 1: Overview of LINQ

Why use LINQ

LINQ to Objects and LINQ Integrations

Differences Between LINQ and Writing Loops

Module 2: Selecting Data

Getting all Data

Getting a Subset of Columns

Creating Anonymous Classes

Module 3: Ordering Data

Order Products Ascending

Order Products Descending

Using Multiple Fields for Ordering

Module 4: Filter Data Using Where

Filtering Data Using Where

Using Multiple Fields for Filtering

Using an Extension Method

Adding Multiple Where Clauses

Module 5: Selecting Single Elements

Using Find(), First(), Last(), and Single()

Using FirstOrDefault(), LastOrDefault(), and SingleOrDefault()

Module 6: Partitioning Data using Take, Skip and Distinct

Using Take(), TakeWhile, and using the Range Operator

Using Skip() and SkipWhile()

Using Distinct() and DistinctBy()

Chunking a Collection into Smaller Collections

Module 7: Query a Collection for Any or All Items

Use All() to See if All Items in a Collection Meet a Condition

Module 8: Look for Differences Between Collections

Module 9: Combine Collections Together

Module 10: Joining Collections Together

Module 11: Grouping Collection Data

Module 12: Aggregating Data

Using Count()

Using Min() and Max()

Using MinBy() and MaxBy()

Using Average() and Sum()

Create Your Own Custom Aggregators

Module 13: Iterations using LINQ

Using ForEach() to Perform Iteration

Calling Methods While Iterating

Module 14: Understanding Deferred Execution

What is Deferred Execution and Why it is Important

Streaming and Non-Streaming Operations

Which Operators are Streaming and Which are Non-Streaming

Several Examples to Illustrate Deferred Execution