Aia Al-Qasab

T5616SN

OBJECT-ORIENTED PROGRAMMING

ASSIGNMENT 3

VISUAL STUDIO CONSOLE APPLICATION

REVISITING LINQ

September 2017



VISUAL STUDIO CONSOLE APPLICATION

REVISITING LINQ

LINQ (Language Integrated Query) is uniform query syntax in C# and VB.NET used to save and retrieve data from different sources. It integrated in C# or VB, thereby eliminating the mismatch between programming languages and databases, as well as providing a single querying interface for different types of data sources. It always works with objects so you can use the same basic coding patterns to query and transform data in XML documents, SQL databases, ADO.NET Datasets, .NET collections, and any other format for which a LINQ provider is available.

LINQ to Objects, the LINQ features designed to add query functionality

Over in-memory collections, makes this scenario very easy to implement.

Although the syntax is foreign now.

The benefits of LINQ is LINQ appeals to different people for different reasons. Some benefits might not be completely obvious with the current state of the many LINQ

Elements that have shipped. The extensibility designed into the LINQ libraries

and compilers will ensure that LINQ will grow over time, remaining a current and important technology to understand for many years to come.

It has been designed with extensibility in mind. Not only can new operators be added when a need arises, but entire new data sources can be added to the LINQ framework (caveat: operator implementation often needs to consider data source, and this can be my point is that it’s possible, and for LINQ to Objects, actually pretty simple).Not only are the LINQ extension points exposed, Microsoft had

implemented their specific providers using these same extension points. This will

ensure that any provider, whether it be from open-source community projects

to competitive data-access platforms, will compete on a level playing field, Extension methods declared in a namespace are available to call from any file that includes a

“using” clause for that namespace. For instance, to make the LINQ to Objects extension methods available to your code, include the “using System.Linq;” clause at the top of the class code file. The compiler will automatically give precedence to any instance methods defined for a type, meaning that it will use a method defined in a class if it exists before it looks for an extension method that satisfies the method name and signature.

C# introduced an object initialization shortcut syntax that allows a single C# statement to both construct a new instance of a type and assign property values in one statement. While it is good programming practice to use constructor arguments for all critical data in order to ensure that a new type is stable and ready for use immediately after it is initialized (not allow objects to be instantiated into an invalid state), Object Initializers

reduce the need to have a specific parameterized constructor for every variation of noncritical data argument set needed over time.

Query expressions is the feature where all of the previous new language constructs merge, and their pivotal role in LINQ can be seen. The following is a brief overview; future chapters cover how to write query expressions in detail, but for now focus on how the language enhancements combine into a query language integrated into C#.

LINQ to Objects allows .NET developers to write “queries” over collections of objects. Microsoft provides a large set of query operators out of the box, and these operators offer a similar depth of functionality to what is expected from any SQL language working with a relational database. Traditionally, working with collections of objects meant writing a lot of looping code using for loops or “foreach” loops to iterate through a collection, carrying out filtering using if statements, while performing other computations like keeping a running sum of a total property. LINQ frees you from having to write looping code; it allows you to write queries that filter a list or calculate aggregate functions on elements in an in-memory collection, among many other capabilities. You

can write queries against any collection type that implements an inter-

face called “IEnumerable<T>”, which most built-in collection classes available in the .NET Framework class libraries certainly do, including simple arrays.

A key aspect of accessing relational data is the concept of joining to related data. Relational database systems (such as Microsoft SQL Server) have powerful join capabilities built in to allow queries to be written against normalized data, which is a fancy term for “not repeating data” by separating data across multiple tables and linking by a common value.

a. LINQ to Objects in C#. Use examples to clarify.

b. Query Syntax and Method Syntax in LINQ. Use examples to clarify.

























