1. Extension Methods:

Sometimes we will be in the need of extending the functionality of existing types. In such cases we generally do it by inheriting such type into our own types and provide additional functionality or by composition. But this is all way to heavy, instead we can use Extension Methods to do this.

Basically, extension methods provide a way to extend existing classes, structs and interfaces. Even generic types can also be extended. The types in any place of inheritance hierarchy can be easily extended.

Lambdas are replacement of anonymous methods

**Lambda vs Named Methods vs Anonymous Methods**

Graphical user interface, text, application, chat or text message

Description automatically generated

Multiple Enumeration Pitfalls:

Suppose if you are iterating\enumerating a linq query result more than once (due to lazy evaluation), the WHERE clause of the query execute multiple times (as many no. of times you make iterations). This execution of multiple WHERE clauses even may query the database. In such cases it will hits the database multiple times and that is really a bad idea. It may go even worse when the data changes between these multiple calls and you may get stale data between iterations.

To avoid such situations when you are ought to do multiple iterations, just use .ToList() on your Linq query, which avoids multiple time execution of query for each of your iterations.

Ex: var result = list.Where(emp => emp.Rating > 200);

//1st iteration

Foreach(var emp in result)

Console.Write(emp.FirstName); //Queries the data store

//2nd iteration

Foreach(var emp in result)

Console.Write(emp.Department); //Queries the data store again

Fix: Rewrite the Linq query

var result = list.Where(emp => emp.Rating > 200).ToList();

How to alter a List via iteration?

Graphical user interface, text, application

Description automatically generated

RemoveAll(). Is the best among all techniques.