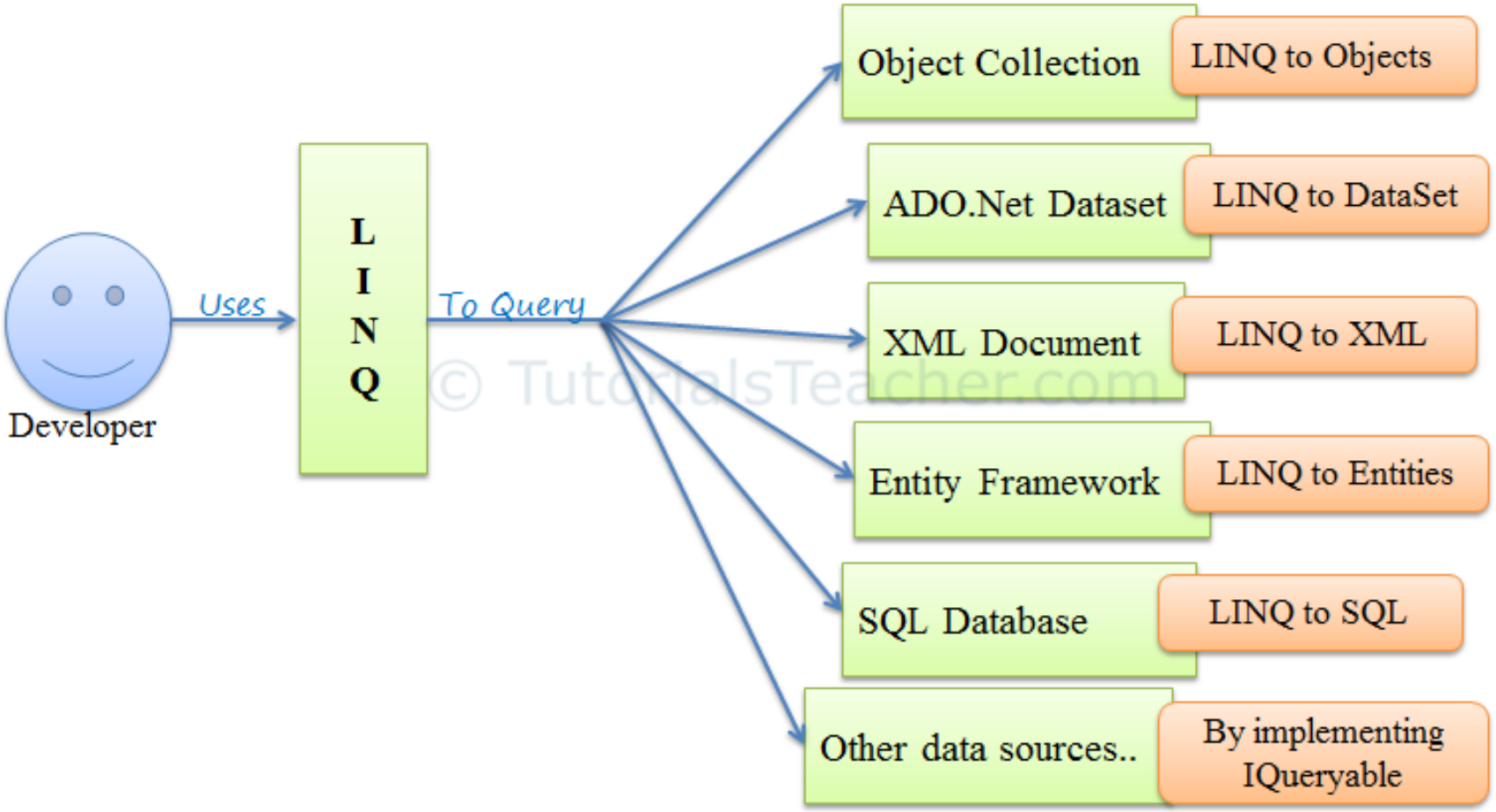
LINQ

- LINQ stands for Language Integrated Query
- LINQ is a new technology introduced in .NET 3.5 with Visual Studio 2008 or later versions.
- LINQ can be used with C# or Visual Basic.
- LINQ is a Object –oriented query language
- LINQ (Language Integrated Query) is uniform query syntax used to retrieve data from different data sources like an Object Collection, DB tables, XML etc.
- LINQ 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 uses System.Linq namespace it includes the necessary classes & interfaces for LINQ.

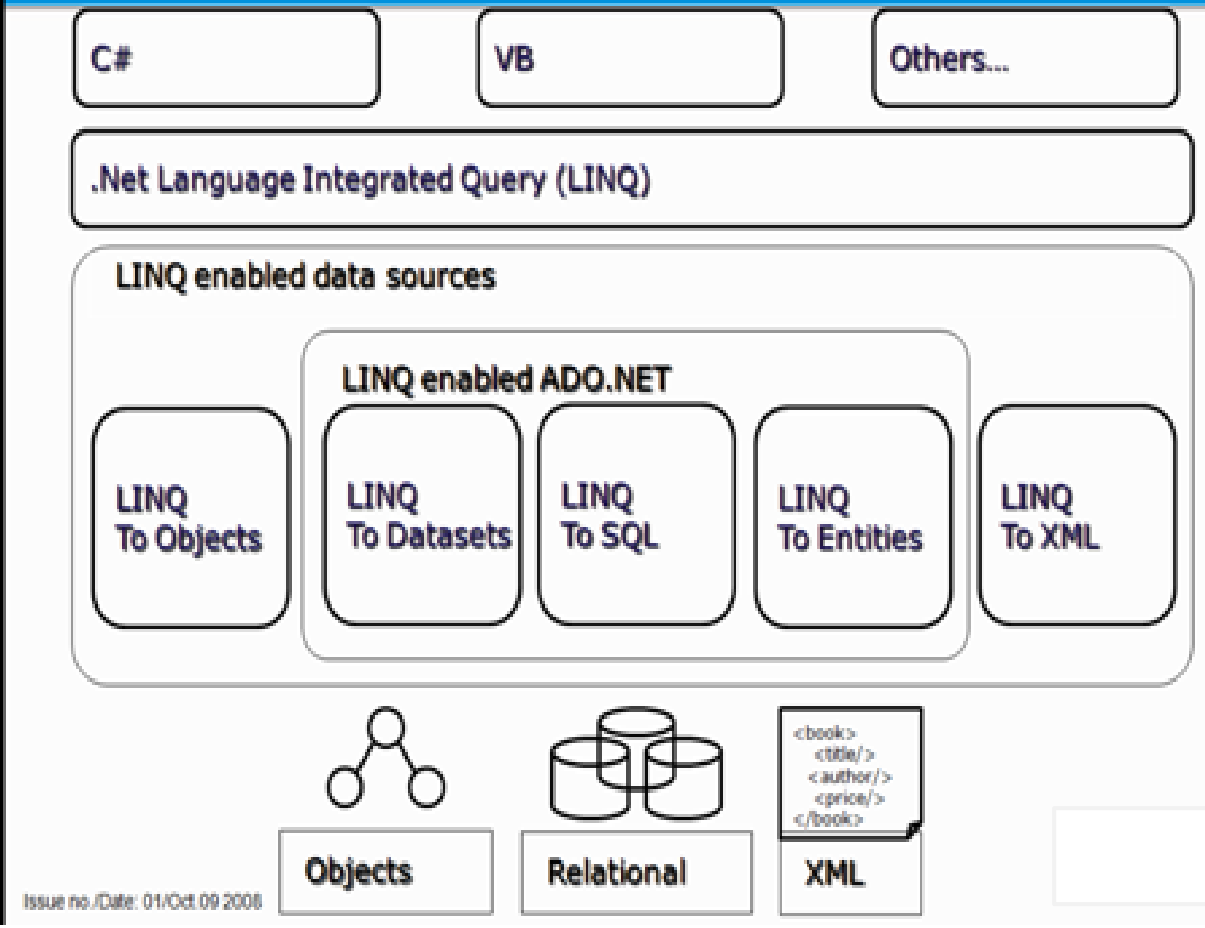
Advantages of LINQ

- **Familiar language:** Developers don't have to learn a new query language for each type of data source or data format.
- **Less coding:** It reduces the amount of code to be written as compared with a more traditional approach.
- **Readable code:** LINQ makes the code more readable so other developers can easily understand and maintain it.
- **Standardized way of querying multiple data sources:** The same LINQ syntax can be used to query multiple data sources.
- **Compile time safety of queries:** It provides type checking of objects at compile time.
- **IntelliSense Support:** LINQ provides IntelliSense for generic collections.
- **Shaping data:** You can retrieve data in different shapes.

LINQ


Developer
Uses
L
I
N
Q
To Query
Object Collection
LINQ to Objects
ADO.Net Dataset
LINQ to DataSet
XML Document
LINQ to XML
Entity Framework
LINQ to Entities
SQL Database
LINQ to SQL
Other data sources..
By implementing IQueryable
© TutorialsTeacher.com

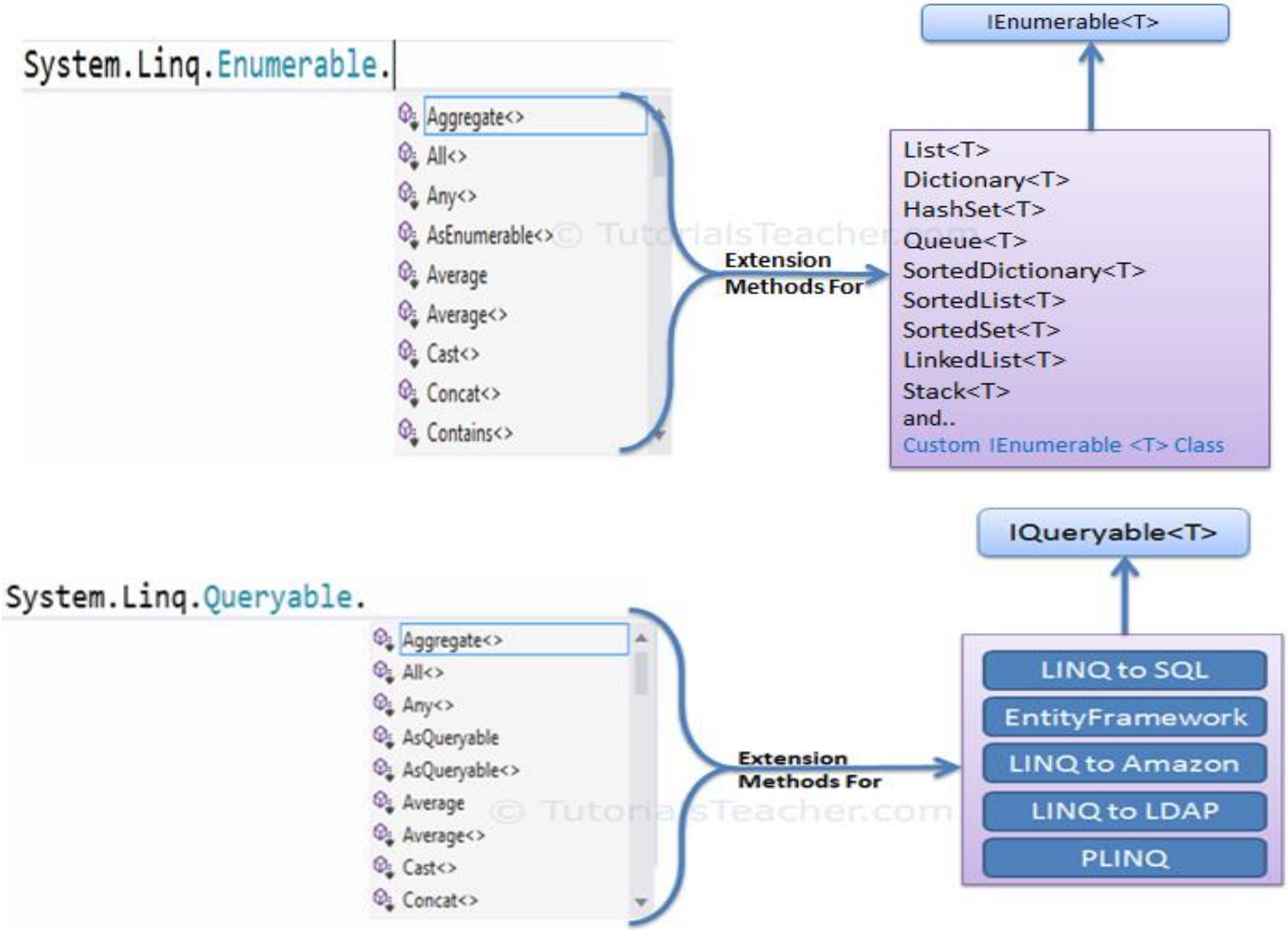
LINQ Architecture



Core Assemblies in LINQ

- The core assemblies in LINQ are:
  - using System.Linq
    - Provides Classes & Interface to support LINQ Queries
  - using System.Collections.Generic
    - Allows the user to create Strongly Typed collections that provide type safety and performance (LINQ to Objects)
  - using System.Data.Linq
    - Provides the functionality to access relational databases (LINQ to SQL)
  - using System.Xml.Linq
    - Provides the functionality for accessing XML documents using LINQ (LINQ to XML)
  - using System.Data.Entity
    - Provides the functionality to access relational databases (Entity Framework)

LINQ


System.Linq.Enumerable.
Aggregate<>
All<>
Any<>
AsEnumerable<>
Average
Average<>
Cast<>
Concat<>
Contains<>
Extension Methods For
IEnumerable<T>
List<T>
Dictionary<T>
HashSet<T>
Queue<T>
SortedDictionary<T>
SortedList<T>
SortedSet<T>
LinkedList<T>
Stack<T>
and..
Custom IEnumerable <T> Class
System.Linq.Queryable.
Aggregate<>
All<>
Any<>
AsQueryable
AsQueryable<>
Average
Average<>
Cast<>
Concat<>
Extension Methods For
IQueryable<T>
LINQ to SQL
EntityFramework
LINQ to Amazon
LINQ to LDAP
PLINQ

LINQ Query Syntax

- There are two basic ways to write a LINQ query to IEnumerable collection or IQueryable data sources.
  - Query Syntax or Query Expression Syntax
  - Method Syntax or Method extension syntax
- Query Syntax:
- Query syntax is similar to SQL (Structured Query Language) for the database. It is defined within the C# or VB code.

from <*range variable*> in <*IEnumerable<T> or IQueryable<T> Collection*>
<Standard Query Operators>
<select or groupBy operator> <*result formation*>

LINQ Query Syntax

- Declarative query syntax :

```
var x= from item in data_source
       where condition
       select item
```

- The query variable x only stores the query commands
- The actual execution happens only when some operation is requested like iteration. This is refereed to as *deferred execution*.
- While the syntax allows usage of "var" keyword, what the query really returns is a **`IEnumerable`** object.
- Hence **`foreach`** can be used with the result of LINQ.
- Note that LINQ query is case sensitive.

Standard Query Operators

- There are over 50 standard query operators available in LINQ that provide different functionalities like filtering, sorting, grouping, aggregation, concatenation, etc.

Classification	Standard Query Operators
Filtering	Where, OfType
Sorting	OrderBy, OrderByDescending, ThenBy, ThenByDescending, Reverse
Grouping	GroupBy, ToLookup
Join	GroupJoin, Join
Projection	Select, SelectMany
Aggregation	Aggregate, Average, Count, LongCount, Max, Min, Sum
Quantifiers	All, Any, Contains
Elements	ElementAt, ElementAtOrDefault, First, FirstOrDefault, Last, LastOrDefault, Single, SingleOrDefault
Set	Distinct, Except, Intersect, Union
Partitioning	Skip, SkipWhile, Take, TakeWhile
Concatenation	Concat
Equality	SequenceEqual
Generation	DefaultEmpty, Empty, Range, Repeat
Conversion	AsEnumerable, AsQueryable, Cast, ToArray, ToDictionary, ToList

LINQ with Objects

- The term "LINQ to Objects" refers to the use of LINQ queries Objects that implement IEnumerable , meaning all collection classes like List, Dictionary as well as arrays and string can use LINQ.
- The collection name become the data source.

```
var x= from item in data_source
       where condition
       select item
```

- The from clause similar to the for-each statement.
- An identifier is used to refer to individual item in the collection. The where clause uses this identifier name to filter the collection.
- This is a very powerful tool since a collection can be filtered using multiple where conditions. Where clause can use any C# condition that evaluated to a boolean value.
- The query returns IEnumerable object.

Query Methods

- Query methods provide a short cut way of writing queries.
- These methods can be used on any enumerable object.
- **`System.Linq.Enumerable`** methods have query methods and these extend the functionality of **`IEnumerable<T>`**
- Methods:
  - **`Select`**
  - **`Where`**
  - **`OrderBy, OrderByDescending`**
  - **`Join`**
  - **`GroupBy`**

Example: **`var l= FlowerList.Select(e => e.Name);`**

Other LINQ Methods

- **`All()`**
- **`Any()`**
- **`Contains()`**

Uses the **`Equals()`** method of the class to determine if the element specified is in the collection.

- **`First()`**
- **`Last()`**

Aggregate methods

- `Count()`
- `Sum()`
- `Min()`
- `Max()`
- `Average()`

- These LINQ methods produce single (non-sequential) result. So in such cases, immediate execution takes place.