What is indexer?

ANS=>An indexer allows an object to be indexed such as an array. When you define an indexer for a class, this class behaves similar to a virtual array. You can then access the instance of this class using the array access operator ([ ]). Indexers allow instances of a class or struct to be indexed just like arrays. The indexed value can be set or retrieved without explicitly specifying a type or instance member. Indexers resemble properties except that their accessors take parameters.

Example:- using System;

class SampleCollection<T>

{​​​​​​​

 // Declare an array to store the data elements.

private T[] arr = new T[100];

   // Define the indexer to allow client code to use [] notation.

 public T this[int i]

   {​​​​​​​

    get {​​​​​​​ return arr[i];

}​​​​​​​

  set {​​​​​​​ arr[i] = value;

}​​​​​​​    }​​​​​​​ }​​​​​​​

class Program {​​​​​​​

static void Main()    {​​​​​​

​var stringCollection = new SampleCollection<string>();

stringCollection[0] = "Hello, World";

Console.WriteLine(stringCollection[0]);

 }​​​​​​​ }

​​​​​​​ // The example displays the following output:

// Hello, World.

What is the difference between IEnumerable and IQuerryable?

ANS=>

|  |  |
| --- | --- |
| IEnumerable | IQuerryable |
| IEnumerable exists in System.Collections Namespace. | IQueryable exists in System. Linq Namespace |
| IEnumerable doesn’t support lazy loading | IQueryable support lazy loading |
| Querying data from a database, IEnumerable execute a select query on the server side, load data in-memory on a client-side and then filter data. | Querying data from a database, IQueryable execute the select query on the server side with all filters |
| IEnumerable Extension methods take functional objects | IQueryable Extension methods take expression objects means expression tree |

What is the use of dynamic keyword?

ANS=> The dynamic keyword is used to declare dynamic types. The dynamic types tell the compiler that the object is defined as dynamic and skip type-checking at compiler time; delay type-checking until runtime. All syntaxes are checked and errors are thrown at runtime.

Example:-  static void Main(string[] args)

{​​​​​​​

ExampleClass ec = new ExampleClass();

// The following call to exampleMethod1 causes a compiler error

// if exampleMethod1 has only one parameter. Uncomment the line

// to see the error.

//ec.exampleMethod1(10, 4);

dynamic dynamic\_ec = new ExampleClass();

// The following line is not identified as an error by the

// compiler, but it causes a run-time exception.

dynamic\_ec.exampleMethod1(10, 4);

// The following calls also do not cause compiler errors, whether

// appropriate methods exist or not.

dynamic\_ec.someMethod("some argument", 7, null);

dynamic\_ec.nonexistentMethod(); }​​​​​​​

Fluent API?

Ans=>

Fluent API is an advanced way of specifying model configuration that covers everything that data annotations can do in addition to some more advanced configuration not possible with data annotations. Data annotations and the fluent API can be used together, but Code First gives precedence to Fluent API > data annotations > default conventions.

* Fluent API is another way to configure your domain classes.
* The Code First Fluent API is most commonly accessed by overriding the OnModelCreating method on your derived DbContext.
* Fluent API provides more functionality for configuration than DataAnnotations. Fluent API supports the following types of mappings.

Web Assembly?

Ans=>

WebAssembly is a new computer programming language for the web. WebAssembly code is a low level binary format, that is compatible with the web and can easily run in modern web browsers. The file size generated is small and it loads and executes faster. You can now compile languages like C, C++, Rust, etc. to binary format and it can run on the web just like javascript.

Goals of WebAssembly

The open standards for WebAssembly are developed in a W3C Community Group that includes representatives from all major browsers as well as a W3C Working Group.

The main goals of WebAssembly are mentioned below −

* **Faster**, **Efficient and Portable** − WebAssembly code is meant to run faster on different platforms taking advantage of the hardware available.
* **Easy to read and debug** − WebAssembly, being a low level assembly language, has text format support, that allows you to debug the code for any issues and also to rewrite the code, if necessary.
* **Security** − WebAssembly is safe to run on the web browsers, as it takes care of permissions and same-origin policies.

GUID?

A GUID (Global Unique IDentifier) is a 128-bit integer used as a unique identifier. ... NET using C# Guid class. GUID stands for Global Unique Identifier. A GUID is a 128-bit integer (16 bytes) that you can use across all computers and networks wherever a unique identifier is required.

we use GUID because it have very low probability of being duplicated as it is 128-bit integer(16 bytes) which allow to use GUID across all databse and computer without data collision.