


C# interview Prep

C# Interview Questions (c-sharpcorner.com).

C# Interview Questions

C sharp is an object-oriented programming language developed by Microsoft. C# is used with the .NET framework for creating websites, applications, and  <https://www.interviewbit.com/c-sharp-interview-questions/>



InterviewBit

1) LINQ

- LINQ in C# is used to work with data access from sources such as objects, data sets, SQL Server, and XML. LINQ stands for Language Integrated Query.
- LINQ is a data querying API with SQL like query syntaxes.
- LINQ is a structured query syntax built in C# and VB.NET to retrieve data from different types of data sources such as collections, ADO.Net DataSet, XML Docs, web service and MS SQL Server and other databases.

2) Difference Between String and String Builder?

- **String** is immutable, Immutable means if you create string object then you cannot modify it and It always create new object of string type in memory.
- **Example**

```
string strMyValue = "Hello Visitor";  
// create a new string instance instead of changing the old one  
strMyValue += "How Are";  
strMyValue += "You ??";
```
- **StringBuilder** is mutable, means if create string builder object then you can perform any operation like insert, replace or append without creating new instance for every time. it will update string at one place in memory doesn't create new space in memory. **Example**

```
StringBuilder sbMyValue = new StringBuilder("");  
Append("Hello Visitor");  
sbMyValue.Append("How Are You ??");  
string strMyValue = sbMyValue.ToString();
```

3)What is Mutable and Immutable? what is the main Difference?

- Mutable and immutable are English words that mean "can change" and "cannot change" respectively.
- means the mutable types are those whose data members can be changed after the instance is created
- but Immutable types are those whose data members can not be changed after the instance is created.
- When we change the value of mutable objects, value is changed in same memory. But in immutable type, the new memory is created and the modified value is stored in new memory.

4)OOPS tell 4 OOPS concepts name and tell about Abstraction and Encapsulation?

- OOPs is a concept of modern programming language that allows programmers to organize entities and objects.
- Object Oriented Programming (OOP) is a programming model where programs are organized around objects and data rather than action and logic.
- OOP allows decomposition of a problem into a number of entities called objects and then builds data and functions around these objects.
- Four key concepts of OOPs are
 1. abstraction
 2. encapsulation
 3. inheritance and
 4. polymorphism.

OOPS concepts.

Abstraction

- Abstraction is "To represent the essential feature without representing the background details."
- Abstraction lets you focus on what the object does instead of how it does it.
- Abstraction is the process of hiding the working style of an object, and showing the information of an object in an understandable manner.

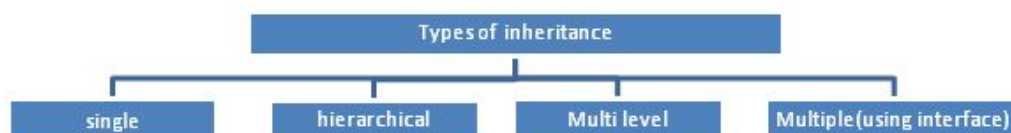
Encapsulation

- Encapsulation is defined as the **wrapping up of data under a single unit**. It is the mechanism that binds together code and the data it manipulates.
- encapsulation is a protective shield that prevents the data from being accessed by the code outside this shield.
- As in encapsulation, the data in a class is hidden from other classes, so it is also known as data-hiding.
- Encapsulation can be achieved by: **Declaring all the variables in the class as private and using C# Properties in the class to set and get the values of variables.**
- Advantages:**Data Hiding,Increased Flexibility,Reusability,Testing code is easy.**

Inheritance

- Acquiring (taking) the properties of one class into another class is called inheritance. Inheritance provides reusability by allowing us to extend an existing class.
- The reason behind OOP programming is to promote the reusability of code and to reduce complexity in code and it is possible by using inheritance.
- **Base class** - is the class from which features are to be inherited into another class.
- **Derived class** - it is the class in which the base class features are inherited.

▼ Types of inheritance:

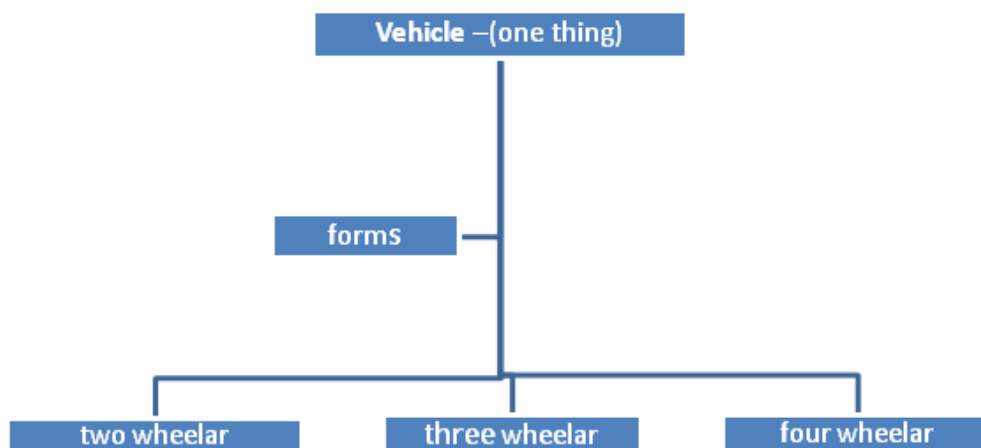


- **Single inheritance:** It is the type of inheritance in which there is one base class and one derived class.
- **Hierarchical inheritance:** This is the type of inheritance in which there are multiple classes derived from one base class. This type of inheritance is used when there is a requirement of one class feature that is needed in multiple classes.

- **Multilevel inheritance:** When one class is derived from another derived class then this type of inheritance is called multilevel inheritance
- The following are some key points about inheritance:
 1. C# does not support multiple inheritances of classes, the same thing can be done using interfaces.
 2. Private members are not accessed in a derived class when one class is derived from another.

Polymorphism

- Polymorphism means one thing having many (poly) forms.



- a vehicle is something that has various forms; two-wheeler, three-wheeler and four-wheeler and so on.
- There are two types of polymorphism; they are: **Compile time polymorphism, Run time polymorphism**

▼ Compile time polymorphism

Compile time polymorphism is done at compile time. The following are examples of compile time polymorphism.

- Method overloading
- Operator overloading

```
namespace BAL
{
```

```

public class Methodoverloading
{
    public int add(int a, int b) //Method 1
    {
        return a + b;
    }
    public int add(int a, int b, int c) //Method 2
    {
        return a + b + c;
    }
    public float add(float a, float b, float c, float d) //Method 3
    {
        return a + b + c + d;
    }
}

```

Method overloading

Creating multiple methods in a class with the same name but different parameters and types is called method overloading.

Method overloading can be done in any of the following ways:

- By changing the number of parameters used.
- By changing the order of parameters.
- By using different data types for the parameters.

Operator overloading

Operator overloading gives the ability to use the same operator to do various operations. It provides additional capabilities to C# operators when they are applied to user-defined data types. It enables to make user-defined implementations of various operations where one or both of the operands are of a user-defined class

▼ Run time polymorphism

Run time polymorphism happens at run time, in other words values are ed at runtime to the method. Runtime polymorphism can be done using method overriding.

```

namespace BAL
{
    public class methodoverriding
    {
        public virtual int balance()
        {
            return 10;
        }
    }
}

```

```

    }
    public class Amount : methodoverriding
    {
        public override int balance()
        {
            return 500;
        }
    }
}

```

Method overriding

Creating the method in a derived class with the same name, same parameters and same return type as in the base class is called method overriding.

- Method overriding is only possible in a derived class, not within the same class where the method is declared.
- Only those methods overridden in the derived class declared in the base class using the virtual keyword or abstract keyword.

▼ Some of the FAQs on polymorphism are:

Question: Can method overloading have the same number of parameters with different return types?

Ans: No, because a conflict occurs in methods when using the parameters.

Question: What is operator overloading?

Ans: We can redefine operators like +, - and * with additional functionalities.

Difference between virtual method and abstract method

Virtual method	Abstract method
We can implement the virtual method inside the abstract class.	Abstract method does not have any implementation.
It is not mandatory to override virtual method in a base class.	It must to override abstract methods in a base class.

Difference between method overloading and method overriding

Method overloading

Creating multiple methods in a class with the same name but different parameters and types is called **method overloading**.

Method overriding

Creating the method in a derived class with the same name, the same parameters and the same return type as in a base class is called **method overriding**.

5)ref & out keyword in C#

- The **out** is a keyword in C# which is used for the passing the arguments to methods as a reference type. It is generally used when a method returns multiple values. The out parameter does not pass the property.
- The **ref** is a keyword in C# which is used for the passing the arguments by a reference. Or we can say that if any changes made in this argument in the method will reflect in that variable when the control return to the calling method. The ref parameter does not pass the property.

6)What are collections and give examples

- C# includes **specialized classes that store series of values or objects** are called **collections**.
- types of collections available in C#:
 1. **non-generic collections.**
 2. **generic collections.**
- The **Collectionsnamespace** contains the **non-generic collection** types and **System.Collections.Generic namespace** includes **generic collection** types.
- it is recommended to use the **generic** collections because they perform **faster** than non-generic collections and also minimize exceptions by giving compile-time errors.

Generic Collection

Generic Collections	Description
<u>List<T></u>	<u>Generic List<T> contains elements of specified type. It grows automatically as you add elements in it.</u>
<u>Dictionary<TKey,TValue></u>	<u>Dictionary<TKey,TValue> contains key-value pairs.</u>

☰ Generic Collections	Aa Description
<u>SortedList<TKey,TValue></u>	<u>SortedList stores key and value pairs. It automatically adds the elements in ascending order of key by default.</u>
<u>Queue<T></u>	<u>Queue<T> stores the values in FIFO style (First In First Out). It keeps the order in which the values were added. It provides an Enqueue(.) method to add values and a Dequeue(.) method to retrieve values from the collection.</u>
<u>Stack<T></u>	<u>Stack<T> stores the values as LIFO (Last In First Out). It provides a Push(.) method to add a value and Pop(.) & Peek(.) methods to retrieve values.</u>
<u>HashSet<T></u>	<u>HashSet<T> contains non-duplicate elements. It eliminates duplicate elements.</u>

Non-generic Collections

☰ Non-generic Collections	Aa Usage
<u>ArrayList</u>	<u>ArrayList stores objects of any type like an array. However, there is no need to specify the size of the ArrayList like with an array as it grows automatically.</u>
<u>SortedList</u>	<u>SortedList stores key and value pairs. It automatically arranges elements in ascending order of key by default. C# includes both, generic and non-generic SortedList collection.</u>
<u>Stack</u>	<u>Stack stores the values in LIFO style (Last In First Out). It provides a Push(.) method to add a value and Pop(.) & Peek(.) methods to retrieve values. C# includes both, generic and non-generic Stack.</u>
<u>Queue</u>	<u>Queue stores the values in FIFO style (First In First Out). It keeps the order in which the values were added. It provides an Enqueue(.) method to add values and a Dequeue(.) method to retrieve values from the collection. C# includes generic and non-generic Queue.</u>
<u>Hashtable</u>	<u>Hashtable stores key and value pairs. It retrieves the values by comparing the hash value of the keys.</u>
<u>BitArray</u>	<u>BitArray manages a compact array of bit values, which are represented as Booleans, where true indicates that the bit is on (1) and false indicates the bit is off (0).</u>

7)Class and object.

- A **class** is a blueprint of an object that contains variables for storing data and functions to perform operations on the data.
- The software is divided into a number of small units called **objects**. The data and functions are built around these objects.
- When an object is created using the new operator, memory is allocated for the class in the heap, the object is called an instance and its starting address will be stored in the object in stack memory.

8) Abstract class and interface

Abstract class

- An **abstract class** is a way to achieve the abstraction in C#. An Abstract class is never intended to be instantiated directly.
- This class must contain at least one abstract method, which is marked by the keyword or modifier **abstract** in the class definition.
- The Abstract classes are typically used to define a base class in the class hierarchy.

Interface

- **Interface** can have methods, properties, events, and indexers as its members.
- But interfaces will contain only the declaration of the members. The implementation of interface's members will be given by the class who implements the interface implicitly or explicitly.

9) Diff b/w abstract class and interface

oops interface vs abstract class

Interface	Abstract class
Interface support multiple inheritance	Abstract class does not support multiple inheritance
Interface doesn't contain Data Member	Abstract class contains Data Member
Interface doesn't contain Constructors	Abstract class contains Constructors
An interface contains only incomplete member (signature of member)	An abstract class contains both incomplete (abstract) and complete member
An interface cannot have access modifiers by default everything is assumed as public	An abstract class can contain access modifiers for the subs, functions, properties
Member of interface can not be Static	Only Complete Member of abstract class can be Static

10)Scopes

- The **part of the program where a particular variable is accessible** is termed as the Scope of that variable.
- C# scope rules of variables can be divided into three categories as follows:

▼ Class Level Scope

- **Declaring the variables in a class** but outside any method can be directly **accessed anywhere in the class**.
- These variables are also termed as the fields or class members.

▼ Method Level Scope

- Variables that are **declared inside a method** have method level scope. These are **not accessible outside the method**.
- However, these variables can be accessed by the nested code blocks inside a method.

▼ Block Level Scope

- These variables are **generally declared inside the for, while statement etc**.
- These variables are **also termed as the loop variables** or statements variable as they have limited their **scope up to the body of the statement in which it declared**.

11)object type variables vs dynamic variable types

- You can store any type of value in the dynamic data type variable. Type checking for these types of variables takes place at run-time.
- **The Object Type** is the ultimate base class for all data types in C# Common Type System . The object is an alias for System. Object class. The object types can be assigned values of any other types, value types, reference types, predefined or user-defined types.
- Dynamic types are similar to object types except that type checking for object type variables takes place at compile time, whereas that for the dynamic type variables takes place at runtime.
- Example of Dynamic Type -

```
dynamic z = 100;
```

Example of Object Type -

```
object obj = 100
```

12) difference between lists vs arrayList

Arraylist vs list

ArrayList	List	Title
ArrayList are not strongly typed	List are strongly typed	Untitled
In ArrayList it can store any type of data, we dont have to specify the type of data it is going to store.	In List we have to specify the type of data elements it is going to store.	Untitled
Namespace : System.Collections	Namespace : System.Collections.Generic	Untitled
ArrayList needs boxing and unboxing.	No need of boxing and unboxing	Untitled
Here the elements are stored as an object type.	Here the data type it is going to store is strongly typed and is given as generic parameter.	Untitled
ArrayList myArray = new ArrayList();myArray.Add("Dot Net");myArray.Add(100); //no error at compile time int sum=0;foreach(int i in myArray){sum += i; } // runtime error	List<string> myArray = new List<string>();myArray.Add("Dot Net");myArray.Add(100); //compile time error int sum=0;foreach(int i in myArray){sum += i; }	Untitled

13) Explain thread vs multithread

A **process** represents an application whereas a thread represents a module of the application. Process is heavyweight component whereas thread is lightweight.

A **thread** can be termed as lightweight subprocess because it is executed inside a process.

Multithreading in C# is a process in which multiple threads work simultaneously. It is a process to achieve multitasking. It saves time because multiple tasks are being executed at a time. To create multithreaded application in C#, we need to use System.Threading namespace

14)Exceptions

- An **exception is a problem that arises during the execution of a program**. A C# exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero.
- Exceptions provide a way to transfer control from one part of a program to another. C# exception handling is built upon four **keywords: try, catch, finally, and throw**.
 - **try** - A try block identifies a block of code for which particular exceptions is activated. It is followed by one or more catch blocks.
 - **catch** - A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The catch keyword indicates the catching of an exception.
 - **finally** - The finally block is used to execute a given set of statements, whether an exception is thrown or not thrown. For example, if you open a file, it must be closed whether an exception is raised or not.
 - **throw** - A program throws an exception when a problem shows up. This is done using a throw keyword.

Exception Classes in C#

- C# exceptions are represented by classes. The exception classes in C# are mainly directly or indirectly derived from the **System.Exception** class. Some of the exception classes derived from the System.Exception class are the **System.ApplicationException** and **System.SystemException** class.

15)What is a **constructor** in C#?

- A **special method** of the class that is **automatically invoked when an instance of the class is created** is called a constructor.
- The **main use** of constructors is **to initialize the private fields of the class while creating an instance for the class**.
- When you have not created a constructor in the class, **the compiler will automatically create a default constructor** of the class.

- The default constructor initializes all numeric fields in the class to zero and all string and object fields to null.

C#, constructors can be divided into 5 types

1. Default Constructor
2. Parameterized Constructor
3. Copy Constructor
4. Static Constructor
5. Private Constructor

click below link

Constructors and Its Types in C#

A special method of the class that is automatically invoked when an instance of the class is created is called a constructor. The main use of constructors is to

<https://www.c-sharpcorner.com/UploadFile/0c1bb2/constructors-and-its-types-in-C-Sharp/>



▼ Some of the key points regarding constructor are

- A class can have any number of constructors.
- A constructor doesn't have any return type, not even void.
- A static constructor can not be a parametrized constructor.
- Within a class, you can create one static constructor only.

16)Array vs array list



Array	ArrayList
They are fixed length.	They are resizable and variable length.
They are compiled strong type collection.	They are flexible and can accommodate any data types.
Because arrays are of fixed size and strong type collection performance is faster.	In arraylist lots of boxing and unboxing are done there for its performance is slower.

17)Access specifiers

C# Access modifiers or specifiers are the keywords that are used to specify accessibility or scope of variables and functions in the C#

application

Access specifiers

 Access Specifier	 Description
<u>Public</u>	It specifies that access is not restricted.
<u>Protected</u>	It specifies that access is limited to the containing class or in derived class.
<u>Internal</u>	It specifies that access is limited to the current assembly.
<u>protected</u> <u>internal</u>	It specifies that access is limited to the current assembly or types derived from the containing class.
<u>Private</u>	It specifies that access is limited to the containing type.

18)diff between Private n Protected

The difference is who can access those functions.

- **Private** = only members of the same class can access the function.
- **Protected** = Same as private but derived classes can also access.

static modifier

The static modifier on a class means that the class cannot be instantiated, and that all of its members are static. A static member has one version regardless of how many instances of its enclosing type are created.

does multiple inheritance supports in c # :No but it can be achieved by Interfaces

19)what is the keyword used for inheritance base class to sub class:

- The base keyword is used to access members of the base class from within a derived class

Interview Preparation Notes

User Interface:

1. Difference between **HTML** and HTML5?

1. HTML5 supports both audio and video while none of them were part of HTML4.
2. HTML cannot allow JavaScript to run within the web browser, while HTML5 provides full support for running JavaScript.
3. HTML Works with all older browsers while HTML5 works with new browsers.
4. HTML5 supports new types of form controls, such as date and time, email, number, category, title, Url, search, etc.
5. Many elements have been introduced in HTML5. Some of the most important are time, audio, description, embed, fig, shape, footer, article, canvas, navy, output, section, source, track, video, etc.
6. HTML Uses cookies to store data. HTML5 uses local storage to store data.
7. Doctype declaration in html is too long `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`
The DOCTYPE declaration in html5 is very simple `<!DOCTYPE html>`

2. What is CSS?

CSS stands for Cascading Style Sheets. CSS describes **how HTML elements are to be displayed on screen** or in other media.

CSS saves a lot of work. It can control the layout of multiple web pages all at once.

3. What is Bootstrap?

Bootstrap is the most popular CSS Framework for developing responsive websites. Bootstrap 4 is the newest version of Bootstrap.

4. What if Bootstrap bootstrap is not there?

We would be end up using more lines of code in CSS.

5. What is the need of JavaScript?

JavaScript is mainly used for web-based applications and web browsers.

6. Adding interactive behavior to web pages

JavaScript allows users to interact with web pages.

Example: Show or hide more information with the click of a button .

Change the color of a button when the mouse hovers over it.

Slide through a carousel of images on the homepage.

Zooming in or zooming out on an image.

7. Game development

8. Creating web and mobile apps

9. What is the output of "10"+20+30 and 10+20+"30" in js?

102030 and 3030 --- associativity is from left to right

[Eg: 10+20+"30"+40+50 → 30304050 10+"20"+30 → 102030]

1. Explain NULL and Undefined in Js.

In JavaScript, undefined is a type, whereas null an object.

Undefined: It means a variable declared, but no value has been assigned.

Eg: var demo;

alert(demo); //shows undefined

alert(typeof demo); //shows undefined

Null: Whereas, null in JavaScript is an assignment value. You can assign it to a variable.

For example,

var demo = null;

alert(demo); //shows null

alert(typeof demo); //shows object

2. List the datatypes of JS.

There are five types of primitive data types in JavaScript. They are as follows:

Data Type	Description
String	represents sequence of characters e.g. "hello"
Number	represents numeric values e.g. 100
Boolean	represents boolean value either false or true
Undefined	represents undefined value
Null	represents null i.e. no value at all

1. What is Bootstrap? Why to use bootstrap?

Bootstrap is a framework to help you design websites faster and easier.

It includes HTML and CSS based design templates for typography, forms,

buttons, tables, navigation, modals, image carousels, etc.

It also gives you support for JavaScript plugins.

Reasons to use bootstrap:

1. Time Saving
2. Easy-to-use
3. Customizable
4. Responsive Grid System.
5. Open Source

6. What is DOM? What is the purpose of DOM?

When a web page is loaded, the browser creates a Document Object Model of the page.

The Document Object Model (DOM) is a programming interface for HTML and XML(Extensible markup language) documents.

It defines the logical structure of documents and the way a document is accessed and manipulated.

DOM can be thought of as a Tree or Forest(more than one tree).

1. Window Object: Window Object is always at top of the hierarchy.
2. Document object: When an HTML document is loaded into a window, it becomes a document object.
3. Form Object: It is represented by form tags.
4. Link Objects: It is represented by link tags.
5. Anchor Objects: It is represented by a href tags.
6. Form Control Elements: Form can have many control elements such as text fields, buttons, radio buttons, and checkboxes, etc.

7. Types of CSS

There are three types of CSS:

1. **Inline CSS** :Inline CSS contains the CSS property in the body section attached with element is known as inline CSS.

This kind of style is specified within an HTML tag using the style attribute.

Eg:<h1 style="color:green">

2. **Internal or Embedded CSS** : This can be used when a single HTML document must be styled uniquely.

The CSS rule set should be within the HTML file in the head section i.e the CSS is embedded within the HTML file.

Eg: <head>

<style>

h1{

background-color:Grey;

}

</style>

</head>

3. **External CSS** : External CSS contains separate CSS file which contains only style property with the help of tag attributes.

CSS property written in a separate file with .css extension and should be linked to the HTML document using link tag.

Eg: <link rel="stylesheet" href="geeks.css"/>

As Inline has the highest priority, any styles that are defined in the internal and external style sheets are overridden by Inline styles.

Internal or Embedded stands second in the priority list and overrides the styles in the external style sheet.

External style sheets have the least priority. If there are no styles defined either in inline or internal style sheet then external style sheet rules are applied for the HTML tags.

8. Difference Between **Span and div tag**

Div tag is a block level element whereas span tag is inline element.

Div tag accepts align attribute, span tag doesn't

Div tag should be used to wrap a section, for highlighting that section. Span should be used to wrap any specific word that you want to highlight.

Eg: <p>Hi Anusha

</p>

<div>

<table>.....</table>

</div>

9. Explain Sum of 2 numbers in HTML.

We can't add using HTML because html is not a programming language. We can add using JS.

<html>

```

<head>
<script>
var a=15;
var b=20;
var c=a+b;
document.write("sum of numbers is: " +c);
</script>
</head>
</html>

```

10. What is HTML.

HTML stands for Hyper Text Markup Language. HTML is the **standard markup language for creating Web pages**. HTML describes the **structure of a Web page**

11. What is the difference between <a> and <link> tag?

The **anchor element is used** to **link to another page or to a certain part of the page if you use its ID.**

```

<a href="index.html">Home</a> OR <p id="id1">Hey Anusha!!</p><a href="#id1">go to top</a>

```

And **The link tag defines** **a link between a document and an external resource.**

The link tag is used to link to external style sheets.

```

<head>
<link rel="stylesheet" type="text/css" href="theme.css">
</head>

```

12. What is jQuery? Why we use jQuery instead of JS?

jQuery is a JavaScript Library. jQuery greatly simplifies JavaScript programming.

jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.

jQuery is easier to use compared to JavaScript and its other JavaScript libraries.

You need to write fewer lines of code while using jQuery, in comparison with JavaScript.

Eg:

```

In JS: function changeColor(color) {
document.body.style.background = color;

```

```
}  
Onload="changeColor('blue');"  
In jQuery: $ ('body') .css ('background', '#0000FF');
```

13. Javascript Operators

1. Javascript Arithmetic operators: +, -, *, /, %, **, ++, --
2. Javascript Assignment operators: =, +=, -=, *=, /=, %=, **=
3. Javascript string operators: +(concat)
4. Javascript Comparison operator: ==, === (equal value and type), !=, !== (not equal value and type), >, <, <=, >=, ? (ternary)
5. Logical operators : &&, ||, !
6. Bitwise operators: & (AND), | (OR), ^ (XOR), ~ (NOT)

DATABASE:

1. What is **indexing**? What are the **types of indexing**?

Indexes are special lookup tables that the database search engine can use to speed up data retrieval.

An index helps to speed up SELECT queries and WHERE clauses, but it slows down data input, with the UPDATE and the INSERT statements.

Eg: Sql Indexes are similar to textbook indexes. In textbooks, if you need to go to a particular chapter, you go to the index, find the page number of the chapter and go directly to that page.

The basic syntax of a CREATE INDEX is as follows:

```
CREATE INDEX index_name ON table_name;
```

Single-Column Indexes:

```
CREATE INDEX index_name ON table_name (column_name);
```

Unique Indexes:

```
CREATE UNIQUE INDEX index_name on table_name (column_name);
```

Composite Indexes:

```
CREATE INDEX index_name on table_name (column1, column2);
```

Drop Index:

```
DROP INDEX index_name;
```

Types of Indexing:

1. Clustered Index:

With a clustered index the rows are stored physically on the disk in the

same order as the index.

Therefore, there can be only one clustered index.

Syntax: CREATE CLUSTERED INDEX IX_tblStudent_Gender_Score ON student(gender ASC, total_score DESC)

2. Non-clustered Index:

A non-clustered index doesn't sort the physical data inside the table. In non-clustered index the logical order of the index does not match the physical stored order of the rows on disk. In fact, a non-clustered index is stored at one place and table data is stored in another place.

This allows more than one index per table.

Syntax: CREATE NONCLUSTERED INDEX IX_tblStudent_Name ON student(name ASC)

3. What are the types of Joins?

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

1. **INNER JOIN**: Returns records that have matching values in both tables
2. **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table
3. **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table
4. **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table
5. **SELF JOIN**: A self join is a regular join, but the table is joined with itself.

4. What is cross join?

The CROSS JOIN is used to generate a paired combination of each row of the first table with each row of the second table.

This join type is also known as cartesian join.

Eg: (1,2,3)X(4,5,6)=[(1,4),(2,4),(3,4),(1,5),(2,5),(3,5),(1,6),(2,6),(3,6)]

Syntax: SELECT ColumnName_1, ColumnName_2, ColumnName_N FROM [Table_1] CROSS JOIN [Table_2]

Total no of rows after cross join: $m*n$

1. What is Stored procedure?

Stored Procedures are created to perform one or more DML operations on

Database. It is nothing but the group of SQL statements that accepts some input in the form of parameters and performs some task and may or may not returns a value.

Parameter types:

1. **IN:** This is the Default Parameter for the procedure. It always receives the values from calling program.
2. **OUT:** This parameter always sends the values to the calling program.
3. **IN OUT:** This parameter performs both the operations. It Receives value from as well as sends the values to the calling program.

Syntax: CREATE PROCEDURE procedure_name

AS

sql_statement

GO;

Ex1: Stored procedure with one parameter

```
CREATE PROCEDURE SelectAllCustomers (@City nvarchar(30))
```

AS

```
SELECT * FROM Customers WHERE City = @City
```

GO;

```
EXEC SelectAllCustomers @City = 'London'; //executing stored proc
```

Ex2: Stored procedure with multiple parameters

```
CREATE PROCEDURE SelectAllCustomers (@City nvarchar(30), @PostalCode  
nvarchar(10))
```

AS

```
SELECT * FROM Customers WHERE City = @City AND PostalCode =
```

```
@PostalCode
```

GO;

```
EXEC SelectAllCustomers @City = 'London', @PostalCode = 'WA1 1DP';
```

```
//executing stored proc
```

1. What are the difference between function and stored procedure?
 1. The function must return a value but in Stored Procedure it is optional.
 2. Functions can have only input parameters for it whereas Procedures can have input or output parameters.
 3. Functions can be called from Procedure whereas Procedures cannot be called from a Function.

2. Among Stored procedure and inline query which is more efficient?
Stored procedure is efficient. It is easier to troubleshoot a stored procedure than inline query as we can isolate it.

3. What are the **difference between delete and drop?**

DML

Delete is a Data Manipulation Language command, Drop is a Data Definition Language command.

Delete removes some or all tuples from table whereas drop removes entire schema, table, domain, or constraints from the database.

Actions performed by DELETE can be rolled back as it uses buffer. Actions performed by DROP can't be rolled back because it directly works on actual data.

In delete, space occupied by the table in the memory is not freed even if you delete all the tuples of the table using DELETE. drop frees the table space from memory.

4. What are the **difference between drop and truncate?**

The DROP command is used to remove table definition and its contents.

Whereas the TRUNCATE command is used to delete all the rows from the table.

In the DROP command, table space is freed from memory. While the

TRUNCATE command does not free the table space from memory.

DROP is a DDL (Data Definition Language) command. Whereas the

TRUNCATE is also a DDL (Data Definition Language) command.

In the DROP command, integrity constraints will be removed. While in this command, integrity constraints will not be removed.

DROP TABLE table_name; TRUNCATE TABLE table_name;

5. What are the **difference between Delete and truncate?**

The DELETE command is used to delete specified rows (one or more). While this command is used to delete all the rows from a table.

It is a DML (Data Manipulation Language) command. While it is a DDL (Data Definition Language) command.

There may be WHERE clause in DELETE command in order to filter the records. While there may not be WHERE clause in TRUNCATE command. DELETE command is slower than TRUNCATE command. While TRUNCATE command is faster than DELETE command.

Identity of column retains the identity after using DELETE Statement on table. In truncate command Identity of the column is reset to its seed value.

6. What are **Subqueries?** and their types

Subqueries are queries nested within other queries.

Types:

1. Non-correlated subquery:

The inner query can run independently of the outer query.

Ex: `SELECT * FROM CUSTOMERS WHERE ID IN (SELECT ID FROM CUSTOMERS WHERE SALARY > 4500) ;`

2. Correlated subquery:

The inner query can't run independently of the outer query.

Ex: `SELECT EMPLOYEE_ID, salary, department_id FROM employees E WHERE salary > (SELECT AVG(salary) FROM EMP T WHERE E.department_id = T.department_id)`

3. Single Row Sub Query: Sub query which returns single row output.

Ex: `SELECT first_name, salary, department_id FROM employees WHERE salary = (SELECT MIN (salary) FROM employees);`

4. Multiple row sub query: Sub query returning multiple row output. They make use of multiple row comparison operators like IN, ANY, ALL.

Ex: `SELECT first_name, department_id FROM employees WHERE department_id IN (SELECT department_id FROM departments WHERE LOCATION_ID = 100)`

7. Difference between subquery and joins?

Subqueries can be used to return either a scalar (single) value or a row set; whereas, joins are used to return rows.

8. Difference between **primary key and unique key?**

1. Primary key will not accept NULL values whereas Unique key can accept one NULL value.

2. A table can have only primary key whereas there can be multiple unique key on a table.

3. A Clustered index automatically created when a primary key is defined whereas Unique key generates the non-clustered index.

9. What is SQL?

SQL stands for Structured Query Language. SQL lets you access and manipulate databases.

10. Explain DDL, DML, DCL, TCL

1. DDL(Data Definition Language) : DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema.
Examples of DDL commands:
CREATE – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
DROP – is used to delete objects from the database.
ALTER-is used to alter the structure of the database.
TRUNCATE-is used to remove all records from a table.
COMMENT –is used to add comments to the data dictionary.
RENAME –is used to rename an object existing in the database.
2. DML(Data Manipulation Language): The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language.
Examples of DML:
INSERT – is used to insert data into a table.
UPDATE – is used to update existing data within a table.
DELETE – is used to delete records from a database table.
11. DCL(Data Control Language): DCL includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions and other controls of the database system.
Examples of DCL commands:
GRANT-gives user's access privileges to the database.
REVOKE-withdraw user's access privileges given by using the GRANT command.
12. TCL(transaction Control Language): TCL commands deal with the transaction within the database.
Examples of TCL commands:
COMMIT- commits a Transaction.
ROLLBACK- rollbacks a transaction in case of any error occurs.
SAVEPOINT-sets a savepoint within a transaction.
SET TRANSACTION-specify characteristics for the transaction.
13. What is Database?
A database is an organized collection of data, so that it can be easily accessed and managed.

14. Difference between primary key and foreign key

A primary key is used to ensure data in the specific column is unique.

A foreign key is a column or group of columns in a relational database table that provides a link between data in two tables.

Only one primary key is allowed in a table. Whereas more than one foreign key are allowed in a table.

primary key does not allow NULL values. Foreign can contain NULL values.

Primary key value cannot be deleted from the parent table. Foreign key value can be deleted from the child table.

Primary key cannot contain duplicate values whereas foreign key can contain duplicate values.

15. What is ENUM?

The ENUM data type in MySQL is a string object.

It allows us to limit the value chosen from a list of permitted values in the column specification at the time of table creation.

Ex: CREATE TABLE shirts (
id INT PRIMARY KEY AUTO_INCREMENT,
size ENUM('small', 'medium', 'large', 'x-large')
);

16. Explain IsNull() and IFNULL()

IFNULL() : lets you return an alternative value if an expression is NULL

Eg: SELECT ProductName, UnitPrice * (UnitsInStock +
IFNULL(UnitsOnOrder, 0)) FROM Products;

ISNULL(): lets you return an alternative value when an expression is NULL

Eg: SELECT ProductName, UnitPrice * (UnitsInStock +
ISNULL(UnitsOnOrder, 0)) FROM Products;

17. Explain UNION and UNION ALL

UNION: The UNION command combines the result set of two or more SELECT statements (only distinct values)

Eg: SELECT City FROM Customers

UNION

SELECT City FROM Suppliers

ORDER BY City;

UNION ALL: The UNION ALL command combines the result set of two or more SELECT statements (allows duplicate values).

Eg: SELECT City FROM Customers

UNION ALL

SELECT City FROM Suppliers
ORDER BY City;

CLOUD COMPUTING:

1. What is cloud computing? Explain its benefits.

cloud computing is the **delivery of computing services—including servers, storage, databases, networking, software, analytics, and intelligence—over the Internet.**

Benefits:

1. **Cost**

Cloud computing **eliminates the capital expense of buying hardware** and software and setting up and running on-site datacenters.

2. **Speed**

Most cloud computing services are provided self service and on demand, so even vast amounts of computing resources can be provisioned in minutes.

3. Performance:

Improves the performance.

4. Secure:

Many cloud providers offer a broad set of policies, technologies and controls that strengthen your security posture overall, helping protect your data, apps and infrastructure from potential threats.

5. Reliability:

Cloud computing makes data backup, disaster recovery and business continuity easier and less expensive because data can be mirrored at multiple redundant sites on the cloud provider's network.

2. Types of Cloud computing

1. Public Cloud:

Public clouds are owned and operated by a third-party cloud service providers, which deliver their computing resources like servers and storage over the Internet.

Which can be accessed by the public.

Eg: Microsoft Azure

2. Public cloud:

A private cloud refers to cloud computing resources used exclusively by a single business or organisation.

3. Hybrid cloud:

Hybrid clouds combine public and private clouds. By allowing data and applications to move between private and public clouds, a hybrid cloud gives your business greater flexibility, more deployment options and helps optimise your existing infrastructure, security and compliance.

3. Types of cloud services: IaaS, PaaS, SaaS

1. Infrastructure-as-a-service(IaaS):

The most basic category of cloud computing services. With IaaS, you rent IT infrastructure—servers and virtual machines (VMs), storage, networks, operating systems—from a cloud provider on a pay-as-you-go basis.
Eg: AWS, EC2, Rackspace, Google Compute Engine

2. Platform-as-a-service(PaaS):

Platform-as-a-service (PaaS) is a type of cloud computing offering in which a service provider delivers a platform to clients, enabling them to develop, run, and manage business applications without the need to build and maintain the infrastructure.
Eg: AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos.

3. Software-as-a-service(SaaS):

You are provided with access to software. You just have to pay and use the software.
Eg: Google Apps, Microsoft Office 365

4. If I create a virtual machine, in which service will it be stored?
IaaS

5. What is REST (Representational State Transfer)?

It is an architectural style for providing standards between computer systems on the web. Representation of state can be in JSON format, XML or HTML format.

REST Endpoint: A REST Service Endpoint is an endpoint which services a

set of REST resources.

Each endpoint is the location from which APIs can access the resources they need to carry out their function.

6. How does cloud reduces the cost of project?

1. Cloud deployment reduces the Hardware cost. Instead of purchasing in-house equipment, hardware needs are left to the vendor.
2. Cloud solutions can also lead to dramatic decrease in labor and maintenance costs. As a result of the hardware being owned by the vendors and stored in off-site locations, there is less demand for in-house staff.
3. Typically, cloud solutions are available in a pay-as-you-go pricing model. This format provides savings and flexibility in several ways. Company doesn't have to pay for the software that isn't being utilized.

jquery-

jQuery is a JavaScript Library.
jQuery greatly simplifies JavaScript programming.
jQuery is easy to learn.

What is jQuery?

jQuery is a lightweight, "write less, do more", JavaScript library.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

Why jQuery?

There are lots of other JavaScript libraries out there, but jQuery is probably the most popular, and also the most extendable.

Many of the biggest companies on the Web use jQuery, such as:

- Google
- Microsoft
- IBM
- Netflix

CDN-

A **content delivery network (CDN)** refers to a geographically distributed group of servers which work together to provide fast delivery of Internet content.

A CDN allows for the quick transfer of assets needed for loading Internet content including HTML pages, javascript files, stylesheets, images, and videos. The popularity of CDN services continues to grow, and today the majority of web traffic is served through CDNs, including traffic from major sites like Facebook, Netflix, and Amazon.

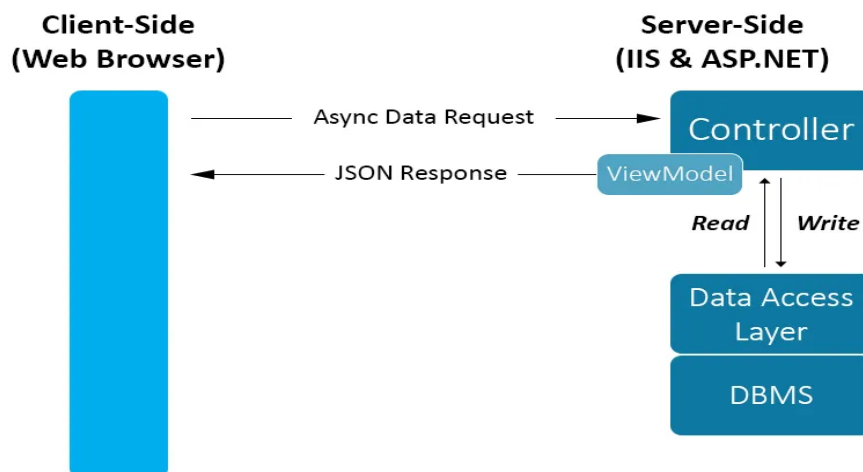
A properly configured CDN may also help protect websites against some common malicious attacks, such as Distributed Denial of Service (DDOS) attacks.

The HTTP Request-Response Data Flow in Native Web Applications -

As most of us already know, the Hypertext Transfer Protocol (HTTP) is designed to enable communications between clients and servers. HTTP works as a request-response protocol between a client and server: in a standard WWW-based scenario the client is usually a web browser, while the server role is played by a Web Application hosted on a remote computer called the Web Server.

The browser submits a HTTP request to the server; that request is processed by the server, which returns a HTTP response to the client. The response contains status information about the request and may also contain the requested content.

Request/Response Flow



What is the difference between session and cookies?

Cookies and Sessions are used to store information. Cookies are only stored on the client-side machine, while sessions get stored on the client as well as a server.

Session

A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit.

A session ends when the user closes the browser or after leaving the site, the server will terminate the session after a predetermined period of time, commonly 30 minutes duration.

Cookies

Cookies are text files stored on the client computer and they are kept of use tracking purpose. Server script sends a set of cookies to the browser. For example name, age, or identification number etc. The browser stores this information on a local machine for future use.

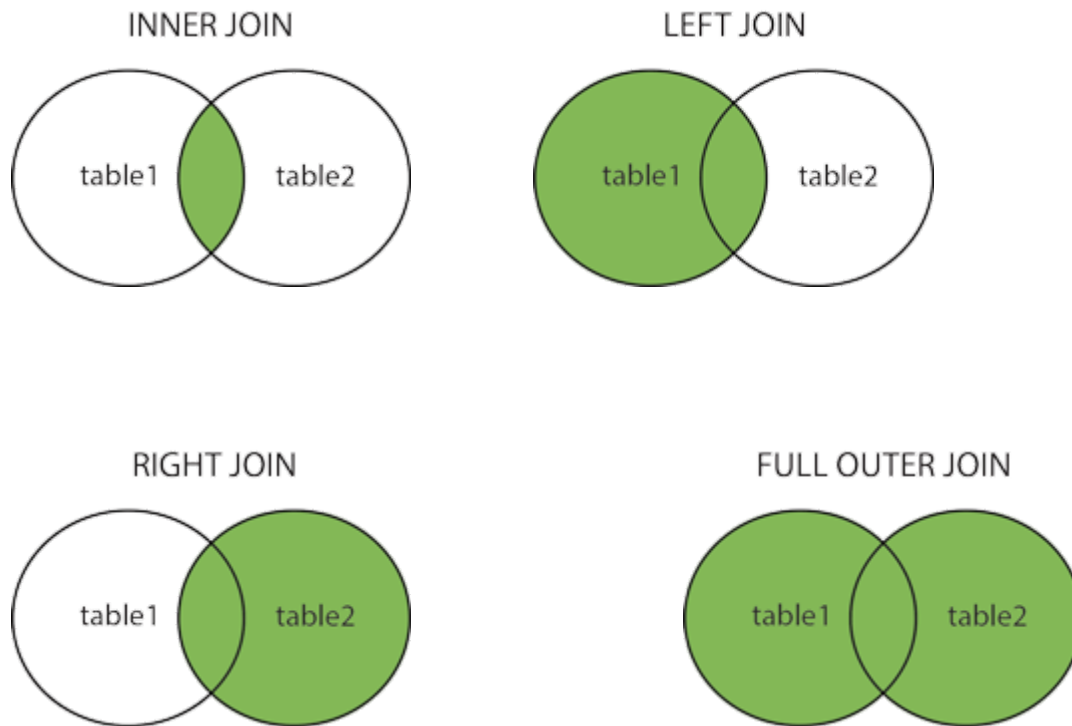
When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user.

stored procedure

A stored procedure is a set of Structured Query Language ([SQL](#)) statements with an assigned name, which are stored in a relational database management system ([RDBMS](#)) as a group, so it can be reused and shared by multiple programs.

Stored procedures can access or modify data in a [database](#), but it is not tied to a specific database or object, which offers a number of advantages.

SQL JOIN



Here are the different types of the JOINS in SQL:

- (INNER) JOIN: Returns records that have matching values in both tables

```
Select column_1, column_2, column_3 FROM table_1 INNER JOIN table_2 ON  
table_1.column = table_2.column;
```

- LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table

```
Select column_1, column_2, column(s) FROM table_1 LEFT JOIN table_2 ON  
table_1.column_name = table_2.column_name;
```

- **RIGHT (OUTER) JOIN:** Returns all records from the right table, and the matched records from the left table

```
Select column_1, column_2, column(s) FROM table_1 RIGHT JOIN table_2 ON  
table_1.column_name = table_2.column_name;
```

- **FULL (OUTER) JOIN:** Returns all records when there is a match in either left or right table

```
Select column_1, column_2, column(s) FROM table_1 FULL JOIN table_2 ON  
table_1.column_name = table_2.column_name;
```

CROSS JOIN

It is also known as **CARTESIAN JOIN**, which returns the Cartesian product of two or more joined tables. The [CROSS JOIN](#) produces a table that merges each row from the first table with each second table row. It is not required to include any condition in CROSS JOIN.

1. `Select * from table_1 cross join table_2;`

Log4net

Log4net is a logging framework for the .NET platform. It's definitely not the only one, but it's one of the most popular frameworks out there.

A logging framework is a tool that can dramatically reduce the burden of dealing with logs.

abstract class

An abstract class is a special type of class that cannot be instantiated. An abstract class is designed to be inherited by subclasses that either implement or override its methods. In other words, abstract classes are either partially implemented or not implemented at all. You can have functionality in your abstract class—the methods in an abstract class can be both abstract and concrete. An abstract class can have constructors—this is one major difference between an abstract class and an interface. You can take advantage of abstract classes to design components and specify some level of common functionality that must be implemented by derived classes.

ERROR HANDLING

1. Proactive approach

Errors

Programming errors where there is no way to recover/continue gracefully and usually need a programmer to step into and change the code to make the fix. Errors can sometimes be turned into exceptions so that they can be handled within the code.

Error handling best practices

Errors can usually be avoided with simple checks and if simple checks won't suffice errors can also turn into exceptions, so that the application can handle the situation gracefully.

4 main ones I know: try/catch, explicit returns, either, and supervising crashes.

Object Relational Mapping (ORM)

When we work with an object-oriented system, there is a mismatch between the object model and the relational database. RDBMSs represent data in a tabular format whereas object-oriented languages, such as Java or C# represent it as an interconnected graph of objects.

ORM stands for **Object-Relational Mapping** (ORM) is a programming technique for converting data between relational databases and object oriented programming languages such as Java, C#, etc.

Exception Handling

Exceptions provide a way to transfer control from one part of a program to another. C# exception handling is built upon four keywords: **try**, **catch**, **finally**, and **throw**.

- **try** – A try block identifies a block of code for which particular exceptions is activated. It is followed by one or more catch blocks.
- **catch** – A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The catch keyword indicates the catching of an exception.
- **finally** – The finally block is used to execute a given set of statements, whether an exception is thrown or not thrown. For example, if you open a file, it must be closed whether an exception is raised or not.

- **throw** – A program throws an exception when a problem shows up. This is done using a throw keyword.

System.IO.IOException

Handles I/O errors.

System.IndexOutOfRangeException

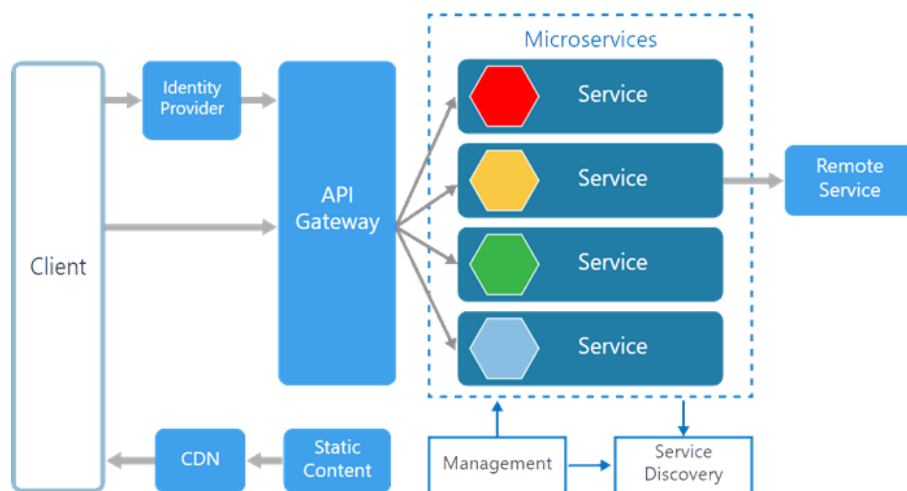
Handles errors generated when a method refers to an array index out of range.

System.DivideByZeroException

Handles errors generated from dividing a dividend with zero.

Microservices

The term microservices portrays a software development style that has grown from contemporary trends to set up practices that are meant to increase the speed and efficiency of developing and managing software solutions at scale. Microservices is more about applying a certain number of principles and architectural patterns as architecture. Each microservice lives independently, but on the other hand, also all rely on each other. All microservices in a project get deployed in production at their own pace, on-premise on the cloud, independently, living side by side..



Management. Maintains the nodes for the service.

Identity Provider. Manages the identity information and provides authentication services within a distributed network.

Service Discovery. Keeps track of services and service addresses and endpoints.

API Gateway. Serves as client's entry point. Single point of contact from the client which in turn returns responses from underlying microservices and sometimes an aggregated response from multiple underlying microservices.

CDN. A content delivery network to serve static resources for e.g. pages and web content in a distributed network

Static Content The static resources like pages and web content

[microservice program](#)

Design Pattern

Design patterns represent the best practices used by experienced object-oriented software developers. Design patterns are solutions to general problems that software developers faced during software development. These solutions were obtained by trial and error by numerous software developers over quite a substantial period of time.

Creational Patterns

These design patterns provide a way to create objects while hiding the creation logic, rather than instantiating objects directly using new operator. This gives program more flexibility in deciding which objects need to be created for a given use case.

Structural Patterns

These design patterns concern class and object composition. Concept of inheritance is used to compose interfaces and define ways to compose objects to obtain new functionalities.

Behavioral Patterns

These design patterns are specifically concerned with communication between objects.

J2EE Patterns

These design patterns are specifically concerned with the presentation tier. These patterns are identified by Sun Java Center.

Dependency Injection

Dependency Injection (DI) is a software design pattern that allows us to develop loosely coupled code. DI is a great way to reduce tight coupling between software components. DI also enables us to better manage future changes and other complexity in our software. The purpose of DI is to make code maintainable.

Triggers

Triggers are stored programs, which are automatically executed or fired when some events occur. Triggers are, in fact, written to be executed in response to any of the following events –

- A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE)
- A **database definition (DDL)** statement (CREATE, ALTER, or DROP).
- A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

Triggers can be defined on the table, view, schema, or database with which the event is associated.

Benefits of Triggers

Triggers can be written for the following purposes –

- Generating some derived column values automatically
- Enforcing referential integrity
- Event logging and storing information on table access
- Auditing
- Synchronous replication of tables
- Imposing security authorizations
- Preventing invalid transactions

Delegates

C# delegates are similar to pointers to functions, in C or C++. A **delegate** is a reference type variable that holds the reference to a method. The reference can be changed at runtime.

Delegates are especially used for implementing events and the call-back methods. All delegates are implicitly derived from the **System.Delegate** class.

Declaring Delegates

```
public delegate int MyDelegate (string s);
```

Bundling and Minification

Bundling and minification techniques were introduced in MVC 4 to improve request load time. Bundling allows us to load the bunch of static files from the server in a single HTTP request.

Minification

Minification technique optimizes script or CSS file size by removing unnecessary white space and comments and shortening variable names to one character.

Bundle Types

MVC 5 includes following bundle classes in System.web.Optimization namespace:

ScriptBundle: ScriptBundle is responsible for JavaScript minification of single or multiple script files.

StyleBundle: StyleBundle is responsible for CSS minification of single or multiple style sheet files.

DynamicFolderBundle: Represents a Bundle object that ASP.NET creates from a folder that contains files of the same type.

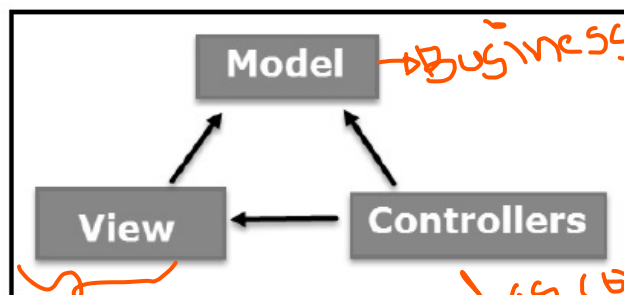
Generic Collections In C#

C# includes specialized classes that store series of values or objects are called collections.

A generic collection is strongly typed (you can store one type of objects into it) so that we can eliminate runtime type mismatches, it improves the performance by avoiding boxing and unboxing.

Generic Collections	Description
List<T>	Generic List<T> contains elements of specified type. It grows automatically as you add elements in it.
Dictionary<TKey,TValue>	Dictionary<TKey,TValue> contains key-value pairs.
SortedList<TKey,TValue>	SortedList stores key and value pairs. It automatically adds the elements in ascending order of key by default.
Queue<T>	Queue<T> stores the values in FIFO style (First In First Out). It keeps the order in which the values were added. It provides an Enqueue() method to add values and a Dequeue() method to retrieve values from the collection.
Stack<T>	Stack<T> stores the values as LIFO (Last In First Out). It provides a Push() method to add a value and Pop() & Peek() methods to retrieve values.
HashSet<T>	HashSet<T> contains non-duplicate elements. It eliminates duplicate elements.

MVC Framework



return
ctrls

business logic
uses collection
ctrl to V.

Model

The Model component corresponds to **all the data-related logic that the user works with**. This can represent either the data that is being transferred between the View and Controller components or any other **business logic-related data**. **For example**, a **Customer object will retrieve the customer information from the database, manipulate it and update it data back to the database or use it to render data.**

View

The View component is **used for all the UI logic of the application**. **For example**, the Customer view will include all the UI components such as **text boxes, dropdowns, etc.** that the final user interacts with.

Controller

Controllers act as an **interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output**. **For example**, the **Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data.**

ASP.NET MVC

ASP.NET supports three major development models: Web Pages, Web Forms and MVC (Model View Controller). ASP.NET MVC framework is a lightweight, highly testable presentation framework that is integrated with the existing ASP.NET features, such as master pages, authentication, etc. Within .NET, this framework is defined in the System.Web.Mvc assembly. The latest version of the MVC Framework is 5.0. We use Visual Studio to create ASP.NET MVC applications which can be added as a template in Visual Studio.

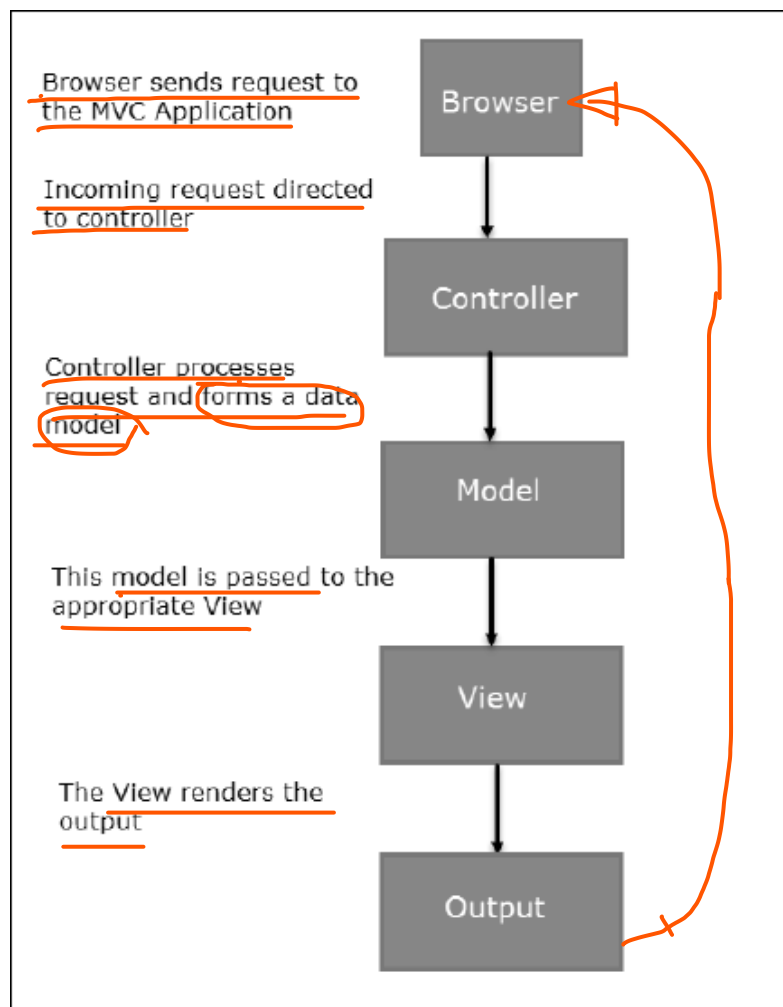
ASP.NET MVC Features

ASP.NET MVC provides the following features –

- Ideal for developing complex but lightweight applications.
- Provides an extensible and pluggable framework, which can be easily replaced and customized. For example, if you do not wish to use the in-built Razor or ASPX View Engine, then you can use any other third-party view engines or even customize the existing ones.

- Utilizes the component-based design of the application by logically dividing it into Model, View, and Controller components. This enables the developers to manage the complexity of large-scale projects and work on individual components.
- MVC structure enhances the test-driven development and testability of the application, since all the components can be designed interface-based and tested using mock objects. Hence, ASP.NET MVC Framework is ideal for projects with large team of web developers.
- Supports all the existing vast ASP.NET functionalities, such as Authorization and Authentication, Master Pages, Data Binding, User Controls, Memberships, ASP.NET Routing, etc.
- Does not use the concept of View State (which is present in ASP.NET). This helps in building applications, which are lightweight and gives full control to the developers.

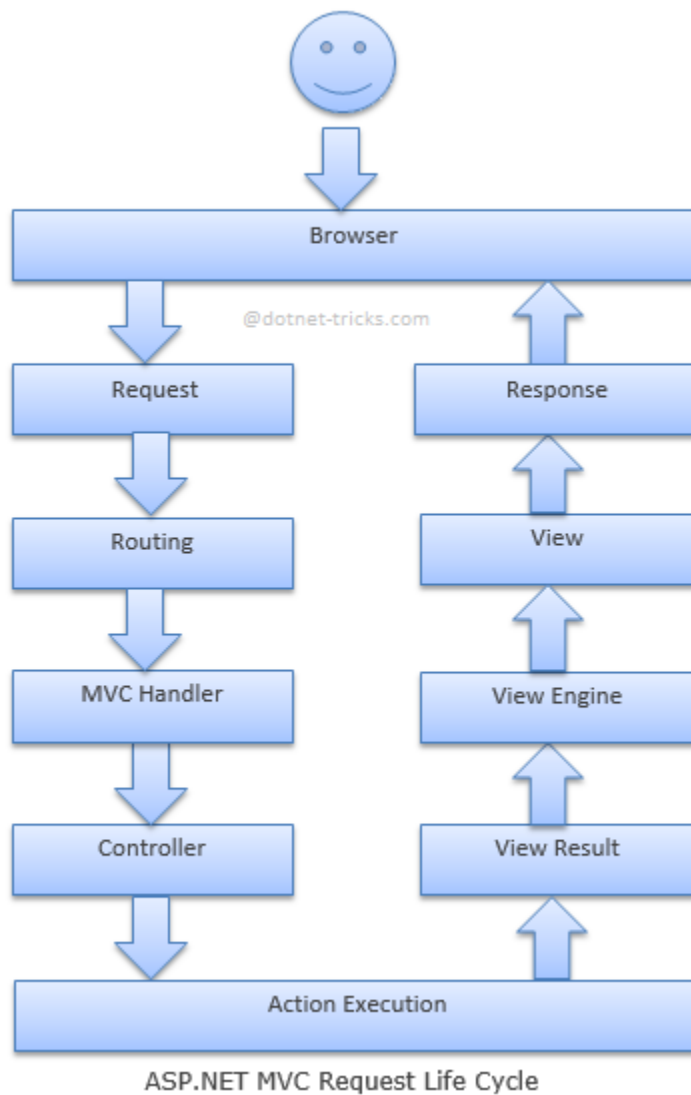
Thus, you can consider MVC Framework as a major framework built on top of ASP.NET providing a large set of added functionality focusing on component-based development and testing.

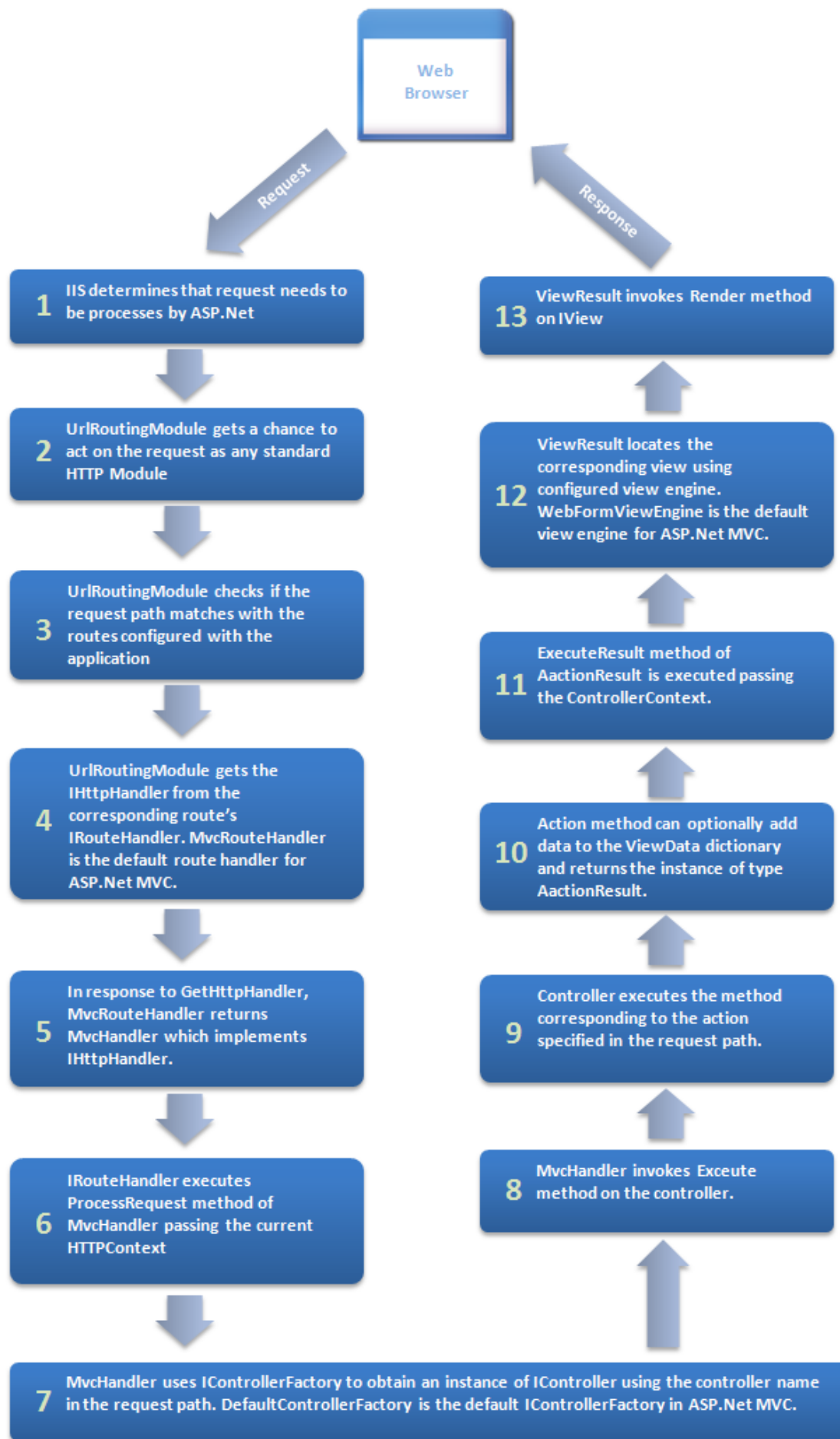


1. MVC stands for Model, View and Controller.
2. Model represents the data
3. View is the User Interface.
4. Controller is the request handler.

Model: Model represents the shape of the data. A class in C# is used to describe a model. Model objects store data retrieved from the database.

ASP.NET MVC Page Life Cycle:





Stored Procedure

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

Stored Procedure Syntax

```
CREATE PROCEDURE procedure_name
```

```
AS
```

```
sql_statement
```

```
GO;
```

Execute a Stored Procedure

```
EXEC procedure_name;
```

SOLID

SOLID is an acronym for the first five object-oriented design (OOD)

These principles establish practices that lend to developing software with considerations for maintaining and extending as the project grows. Adopting these practices can also contribute to avoiding code smells, refactoring code, and Agile or Adaptive software development.

SOLID stands for:

-
- **S** - Single-responsibility Principle
- A class should have one and only one reason to change. meaning that a class should have only one job.
-
- **O** - Open-closed Principle
- Objects or entities should be open for extension but closed for modification.
-
-

-
- **L - Liskov Substitution Principle**
- Let $q(x)$ be a property provable about objects of x of type T . Then $q(y)$ should be provable for objects y of type S where S is a subtype of T .
-
- **I - Interface Segregation Principle**
- A client should never be forced to implement an interface that it doesn't use, or clients shouldn't be forced to depend on methods they do not use.
-
- **D - Dependency Inversion Principle**
- Entities must depend on abstractions, not on concretions. It states that the high-level module must not depend on the low-level module, but they should depend on abstractions.

Unit Testing

UNIT TESTING is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object

Migration

Entity Framework introduced a migration tool that automatically updates the database schema when your model changes without losing any existing data or other database objects. It uses a new database initializer called ***MigrateDatabaseToLatestVersion***.

There are two kinds of Migration:

1. Automated Migration
2. Code-based Migration

Code First to an Existing Database

(entity framework- to connect to the database)

<https://www.tektutorialshub.com/entity-framework/ef-database-connection/>

in-memory database

An in-memory database is a type of purpose-built database that relies primarily on memory for data storage, in contrast to databases that store data on disk or SSDs. In-memory databases are designed to attain minimal response time by eliminating the need to access disks. Because all data is stored and managed exclusively in main memory, it is at risk of being lost upon a process or server failure. In-memory databases can persist data on disks by storing each operation in a log or by taking snapshots.

In-memory databases are ideal for applications that require microsecond response times and can have large spikes in traffic coming at any time such as gaming leaderboards, session stores, and real-time analytics.

sealed (C# Reference)

When applied to a class, the sealed modifier prevents other classes from inheriting from it. In the following example, class B inherits from class A, but no class can inherit from class B.

```
class A {}  
sealed class B : A {}
```

Access Modifiers in C#

Access Modifiers are keywords that define the accessibility of a member, class or datatype in a program. These are mainly used to restrict unwanted data manipulation by external programs or classes. There are **4** access modifiers (public, protected, internal, private) which defines the **6 accessibility levels** as follows:

public

protected

internal

protected internal

private

private protected

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

```
body {  
  
    background-color: lightblue;  
  
}
```

```
h1 {  
  
    color: navy;  
  
    margin-left: 20px;  
  
}
```

Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

The internal style is defined inside the <style> element, inside the head section.

Inline CSS

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

```
<h1 style="color:blue;text-align:center;">This is a heading</h1>
```

```
<p style="color:red;">This is a paragraph.</p>
```


JavaScript Variables

```
<script>

var price1 = 5;

var price2 = 6;

var total = price1 + price2;

document.getElementById("demo").innerHTML =

"The total is: " + total;

</script>
```

DDL is short name of **Data Definition Language**, which deals with database schemas and descriptions, of how the data should reside in the database.

- [CREATE](#) - to create a database and its objects like (table, index, views, store procedure, function, and triggers)
- ALTER - alters the structure of the existing database
- DROP - delete objects from the database
- TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed
- COMMENT - add comments to the data dictionary
- RENAME - rename an object

DML is short name of **Data Manipulation Language** which deals with data manipulation and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE, etc., and it is used to store, modify, retrieve, delete and update data in a database.

- [SELECT](#) - retrieve data from a database
- [INSERT](#) - insert data into a table
- [UPDATE](#) - updates existing data within a table
- [DELETE](#) - Delete all records from a database table
- MERGE - UPSERT operation (insert or update)
- CALL - call a PL/SQL or Java subprogram
- EXPLAIN PLAN - interpretation of the data access path
- LOCK TABLE - concurrency Control

Stored procedures and functions

Functions	Procedures
A function has a return type and returns a value.	A procedure does not have a return type. But it returns values using the OUT parameters.
You cannot use a function with Data Manipulation queries. Only Select queries are allowed in functions.	You can use DML queries such as insert, update, select etc... with procedures.
A function does not allow output parameters	A procedure allows both input and output parameters.
You cannot manage transactions inside a function.	You can manage transactions inside a function.
You cannot call stored procedures from a function	You can call a function from a stored procedure.
You can call a function using a select statement.	You cannot call a procedure using select statements.

Aggregate functions in SQL

In database management an aggregate function is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

Various Aggregate Functions

- 1) Count()
- 2) Sum()
- 3) Avg()
- 4) Min()
- 5) Max()

Debugging stored procedures

Debugging is one of the most important but painful parts of any software process. To find some errors you have to run the code step by step to see which section of the code is responsible for the error. This is called runtime debugging.

Luckily, SQL Server Management Studio (SSMS) comes with automated debugging capabilities to help developers debug their scripts.

Also uses breakpoints.

SQL Server TRY CATCH overview

The TRY CATCH construct allows you to gracefully handle exceptions in SQL Server. To use the TRY CATCH construct, you first place a group of Transact-SQL statements that could cause an exception in a BEGIN TRY...END TRY block as follows:

```
BEGIN TRY
```

```
-- statements that may cause exceptions
```

```
END TRY
```

```
BEGIN CATCH
```

```
-- statements that handle exception
```

```
END CATCH
```

The CATCH block functions

Inside the CATCH block, you can use the following functions to get the detailed information on the error that occurred:

- `ERROR_LINE()` returns the line number on which the exception occurred.
- `ERROR_MESSAGE()` returns the complete text of the generated error message.
- `ERROR_PROCEDURE()` returns the name of the stored procedure or trigger where the error occurred.
- `ERROR_NUMBER()` returns the number of the error that occurred.
- `ERROR_SEVERITY()` returns the severity level of the error that occurred.
- `ERROR_STATE()` returns the state number of the error that occurred.

Note that you only use these functions in the CATCH block. If you use them outside of the CATCH block, all of these functions will return [NULL](#).

Example-

```
CREATE PROC usp_divide(
```

```
    @a decimal,
```

```
    @b decimal,
```

```
    @c decimal output
```

```
) AS
```

```
BEGIN
```

```
    BEGIN TRY
```

```
        SET @c = @a / @b;
```

```
    END TRY
```

```
    BEGIN CATCH
```

```
        SELECT
```

```
            ERROR_NUMBER() AS ErrorNumber
```

```
            ,ERROR_SEVERITY() AS ErrorSeverity
```

```
            ,ERROR_STATE() AS ErrorState
```

```
            ,ERROR_PROCEDURE() AS ErrorProcedure
```

```
            ,ERROR_LINE() AS ErrorLine
```

```
            ,ERROR_MESSAGE() AS ErrorMessage;
```

```
    END CATCH
```

```
END;
```

```
GO
```

Modifying an existing SQL Server stored procedure

To change the stored procedure and save the updated code you would use the ALTER PROCEDURE command as follows.

SQL Join vs Subquery

What are [Joins](#)?

A join is a query that combines records from two or more tables. A join will be performed whenever multiple tables appear in the FROM clause of the query. The select list of the query can select any columns from any of these tables. If join condition is omitted or invalid then a Cartesian product is formed. If any two of these tables have a column name in common, then must qualify these columns throughout the query with table or table alias names to avoid ambiguity. Most join queries contain at least one join condition, either in the FROM clause or in the WHERE clause.

what is [Subquery](#)?

A Subquery or Inner query or Nested query is a query within SQL query and embedded within the WHERE clause. A Subquery is a SELECT statement that is embedded in a clause of another SQL statement. They can be very useful to select rows from a table with a condition that depends on the data in the same or another table. A Subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved. The subquery can be placed in the following SQL clauses they are WHERE clause, HAVING clause, FROM clause.

SQL Cross Join

The SQL CROSS JOIN produces a result set which is the number of rows in the first table multiplied by the number of rows in the second table if no WHERE clause is used along with CROSS JOIN. This kind of result is called as Cartesian Product.

If WHERE clause is used with CROSS JOIN, it functions like an INNER JOIN.

Syntax:

```
SELECT *  
  
FROM table1  
  
CROSS JOIN table2;
```

Second max sal

```
SELECT name, MAX(salary) AS salary  
  
FROM employee  
  
WHERE salary < (SELECT MAX(salary)  
                FROM employee);
```

ADO.NET

ADO.NET provides a bridge between the front end controls and the back end database. The ADO.NET objects encapsulate all the data access operations and the controls interact with these objects to display data, thus hiding the details of movement of data.

Web API

In computer programming, an **application programming interface (API)** is a **set of subroutine definitions, protocols, and tools for building software and applications.**

To put it in simple terms, **API is some kind of interface which has a set of functions that allow programmers to access specific features or data of an application, operating system or other services.**

ASP.NET - Managing State

ASP.NET manages four types of states:

- View State
- Control State
- Session State
- Application State

Code First v/s Database First

Code First Approach

In code first approach we will first create entity classes with properties defined in it. Entity framework will create the database and tables based on the entity classes defined. So database is generated from the code. When the dot net code is run database will get created.

Advantages

1. You can create the database and tables from your business objects.
2. You can specify which related collections are to be eager loaded, or not be serialized at all.
3. Database version control.
4. Good for small applications.

Database First Approach

In this approach Database and tables are created first. Then you create entity Data Model using the created database.

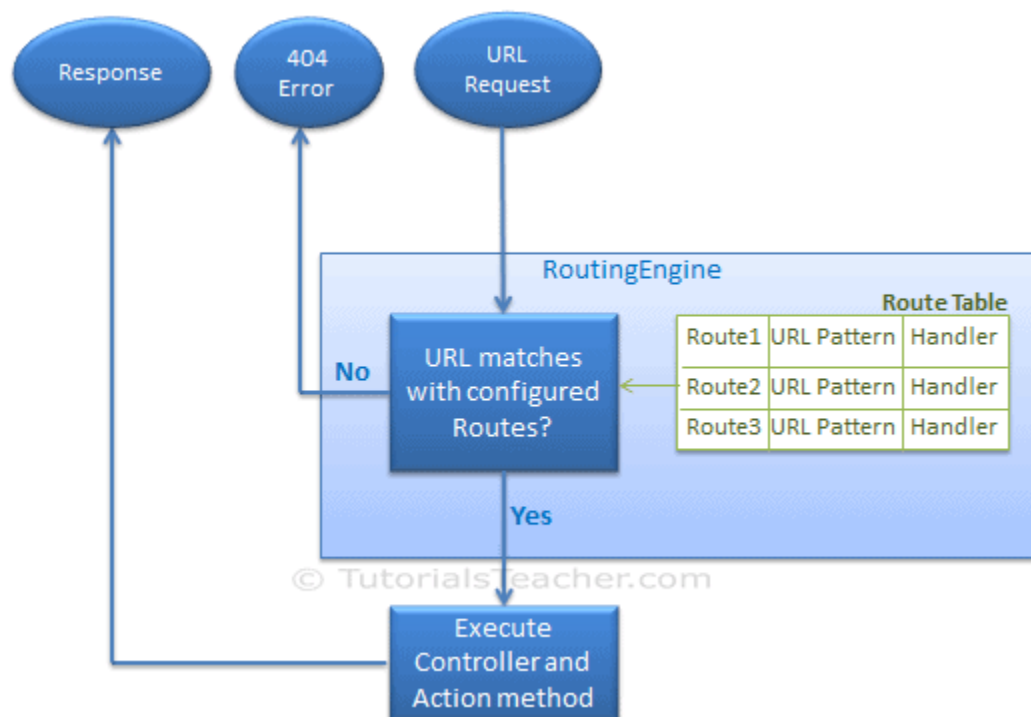
Advantages

1. Simple to create the data model

2. Graphical user interface.
3. Mapping and creation of keys and relationships are easy as you need not have to write any code.
4. Preferred for data intense and large applications

Routing in MVC

In the ASP.NET Web Forms application, every URL must match with a specific .aspx file. For example, a URL `http://domain/studentsinfo.aspx` must match with the file `studentsinfo.aspx` that contains code and markup for rendering a response to the browser.

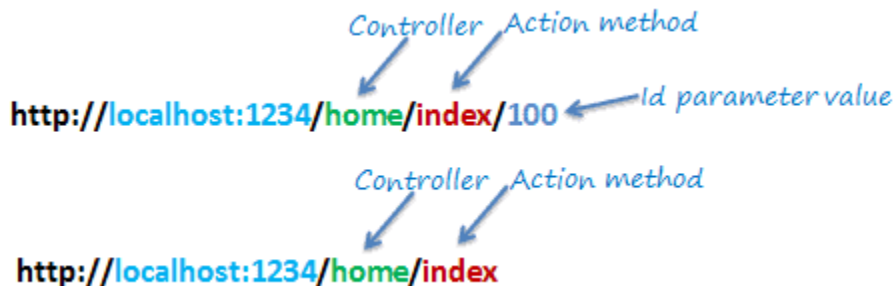


Configure a Route

Every MVC application must configure (register) at least one route configured by the MVC framework by default. You can register a route in `RouteConfig` class, which is in `RouteConfig.cs` under `App_Start` folder. The following figure illustrates how to configure a route in the `RouteConfig` class.

URL Pattern

The URL pattern is considered only after the domain name part in the URL. For example, the URL pattern "{controller}/{action}/{id}" would look like localhost:1234/{controller}/{action}/{id}. Anything after "localhost:1234/" would be considered as a controller name.



Why entity framework use over ado.net.

I suggest you to go with the Entity Framework(EF) because EF is an object-relational mapper that allows the developers to interact with the relational data using the domain-space objects. It works on the top on the ADO.NET by handling the ADO.NET provider.

The ADO.NET provides the dataset, datatables, command, and connection objects. It allows you to work on the higher level objects such as managers, workers, suppliers etc..

The ADO.NET provides the set of software components that the developer can use access data and various data services in the database.

Approaches of Entity Framework

1. model view
2. code first
3. database first

Difference between Primary Key and Foreign Key-

Primary Key:

A primary key is used to ensure data in the specific column is unique. It is a column cannot have NULL values. It is either an existing table column or a column that is specifically generated by the database according to a defined sequence.

Foreign Key:

A foreign key is a column or group of columns in a relational database table that provides a link between data in two tables. It is a column (or columns) that references a column (most often the primary key) of another table.

S.NO.	PRIMARY KEY	FOREIGN KEY
1	A primary key is used to ensure data in the specific column is unique.	A foreign key is a column or group of columns in a relational database table that provides a link between data in two tables.
2	It uniquely identifies a record in the relational database table.	It refers to the field in a table which is the primary key of another table.
3	Only one primary key is allowed in a table.	Whereas more than one foreign key are allowed in a table.
4	It is a combination of UNIQUE and Not Null constraints.	It can contain duplicate values and a table in a relational database.
5	It does not allow NULL values.	It can also contain NULL values.
6	Its value cannot be deleted from the parent table.	Its value can be deleted from the child table.
7	It constraint can be implicitly defined on the temporary tables.	It constraint cannot be defined on the local or global temporary tables.

ASP.NET - Validators

ASP.NET validation controls validate the user input data to ensure that useless, unauthenticated, or contradictory data don't get stored.

ASP.NET provides the following validation controls:

- RequiredFieldValidator
- RangeValidator
- CompareValidator
- RegularExpressionValidator
- CustomValidator
- ValidationSummary

Asp.net and testing

ASP.NET:

1. **What if I write code in wwwroot?**
wwwroot contains css and js files. Where the styling and js functions are defined for webApps. When we try to modify the existing code it will be accessed by all over the project.
2. **What is appsettings? Explain about .JSON file? What will happen if i make changes in Json file? Which is a plain text file, which stores configuration information as key-value pair in json format?**

There are mainly 6 configuration JSON files in ASP.net Core.

1. **global.json**: You can define the solution level settings in the global.json file.
projects – projects property defines the location of source code for your solution.
It specifies two location for projects in the solution: src and test.src contains actual application and test contains any test.
2. **launchsettings.json**: You can define project-specific settings associated with each profile
Visual Studio is configured to launch the application,
including any environment variables that should be used. You can define a framework for your project for compilation and debugging for specific profiles.
3. **appsettings.json**: ASP.NET Core uses AppSettings.json to store custom application settings, DB connection strings, Logging etc.
4. **bundleconfig.json**: You can define configuration for bundling and minification of project.
5. **project.json**: Asp.net Core uses Project.JSON file for storing all project-level configuration settings.
The Project.json file stores configuration information in JSON format.
6. **bower.json**: Bower is a package manager for the web. Bower manages components that contain HTML, CSS, JavaScript, fonts or even image files.
Bower installs the right versions of the packages you need and their dependencies.

3. **What is MVC? What is Asp.Net MVC?**

MVC:

The **Model-View-Controller (MVC)** is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller.
Each of these components are built to handle specific development aspects of an application.

1. **Model**
A set of classes that describes the data you are working with as well as the business logic.
2. **View**
The View component is used for all the UI logic of the application.
3. **Controller**
Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output.

4) **Asp.Net MVC:**

- ASP.NET MVC is an open source web development framework from Microsoft that provides a Model View Controller architecture.
- ASP.net MVC offers an alternative to ASP.net web forms for building web applications. It is a part of the .Net platform for building, deploying and running web apps.

5) Advantages of ASP.NET MVC Framework

1. It manages application complexity by dividing an application into the model, view and controller.
2. It does not use view state or server-based forms. This makes the MVC framework ideal for developers who want full control over the behavior of an application.
3. It provides better support for test-driven development.
4. It is suitable for large scale developer team and web applications.
5. It provides high degree of control to the developer over the application behavior.

6) What is Routing? What are the types?

Routing maps incoming requests to the route handler.

Types:

- **Convention Based Routing:**
In convention based routing, the route is determined based on the conventions defined in the route templates which will map the incoming request to controllers and their action methods.
In Asp.net core MVC application, the convention based routes are defined within the configure method of the startup.cs.
`app.UseMvcWithDefaultRoute()`
- **Attribute Based Routing:**
In attribute based routing, the route is determined based on the attribute which are configured either at the controller or at the action method.
`[Route("Default/GetRecordById/{id}")]`
- **Explain Filters and their types.**
Filter is a custom class where you can write custom logic to execute before or after an action method executes.
- **Authorization filter:**
Performs authentication and authorizes before executing an action method. This filter helps us to determine whether the user is authorized for the current request.
Interface: `IAuthorizationFilter`
Built-in filter: `[Authorize]`, `[RequireHttps]`
- **Resource filter:**
The Resource filters handle the request after authorization. It can run the code before and after the rest of the filter is executed.
This executes before the model binding happens. It can be used to implement caching.
- **Action Filters:**
Performs some operation before and after an action method executes.
Interface: `IActionFilter`
- **Result filters:**
Performs some operation before or after the execution of the view.

Interface: IResultFilter

- **Exception filter:**

Performs some operation if there is an unhandled exception thrown during the execution of Asp.net pipeline.

Built-in filter: [HandleError]

Interface: IExceptionHandler

7) **What is scaffolding?**

Scaffolding is a code generation framework for ASP.NET Web applications. You add scaffolding to your project when you want to quickly add code that interacts with data models. Using scaffolding can reduce the amount of time to develop standard data operations in your project.

8) **What is ADO.NET?**

ADO.NET is the data access component for the .NET Framework. ADO.NET provides a bridge between the front end controls and the back end database.

The ADO.NET objects encapsulate all the data access operations and the controls interact with these objects to display data.

9) **What is Entity Framework? Explain their different approaches**

Entity Framework is an object-relational mapper (O/RM) that enables .NET developers to work with a database using .NET objects.

It eliminates the need for most of the data-access code that developers usually need to write.

1. **Database-first approach:**

In the database-first approach, we generate the context and entities for the existing database using the EDM wizard integrated with Visual Studio.

2. **Code-First approach:**

We use this approach when we do not have an existing database for our application. In the Code-First approach, we start writing the entities and context class and then create the database from these classes using the migration command.

3. **Model-First approach:**

In Model-First Approach, we create entities, relationships, and the inheritance hierarchy.

10) **What is ORM?**

Object-relational mapping is a programming technique for converting data between incompatible type systems in object-oriented programming languages.

It tries to abstract the database access.

11) **Explain ViewBag, ViewData and TempData**

View data ,view bag and temp data are used for same purpose- to transfer data.

ViewData : view data is a dictionary object to pass the data from Controller to View where data is passed in the form of key-value pair.

Typecasting is required to read the data in View if the data is complex. ViewData is faster than ViewBag.

Eg: Controller:

```
Public ActionResult Index()
```

```
{
```

```
ViewData["Title"] = "Welcome";
```

```
return View();
```

```

}
View
<h2>@ViewData["Title"]</h2>

```

ViewBag: ViewBag is a dynamic object to pass the data from Controller to View. And, this will pass the data as a property of object ViewBag.

we have no need to typecast to read the data or for null checking.

Eg: Controller

```

Public ActionResult Index()
{
    ViewBag.Title = "Welcome";
    return View();
}

```

View

```

<h2>@ViewBag.Title</h2>

```

TempData: TempData is a dictionary object to pass the data from one action to other action in the same Controller or different Controllers.

Tempdata is also required to typecast and for null checking before reading data from it.

Eg: Controller

```

Public ActionResult Index()
{
    TempData["Data"] = "I am from Index action";
    return View();
}

```

Public string Get()

```

{
    return TempData["Data"] ;
}

```

12) What is Asp.Net core?

Asp.net core is a open-source, cross platform and high performance platform that allows you to build modern, internet connected and cloud enabled applications.

13) Difference between Asp.net core and Asp.Net

1. Asp.net core is open-source, whereas only some components of asp.net are open source.
2. Asp.net is cross-platform, it is compatible with windows, linux and Mac Os, while asp.net is compatible with windows operating system.
3. .Net Core supports the development and implementation of micro-services and the user has to create a REST API for its implementation.
.Net Framework does not support the development and implementation of microservices but it supports the REST API services.
4. .NET Core offers high performance and scalability..Net Framework is less effective in comparison to .Net Core.
5. .NET Core is compatible with open-source mobile application platforms,.NET Framework does not support any framework for mobile application development.
6. .NET Core provides light-weight editors and command-line tools for all supported platforms..Net Framework is heavy for Command Line Interface.

14) Difference between ADO.NET and Entity Framework.

1. ADO.NET is faster, EF is slower.
2. ADO.NET creates several data layer codes, whereas EF doesn't create data layer codes.
3. ADO.NET doesn't create the codes for data access layer, intermediate layer and mapping codes by itself, EF automatically creates codes for these layers.
4. No scaffolding is used in ADO.NET, Scaffolding is used in EF.

15) Difference between Asp.net web forms and asp.net mvc

1. Asp.Net Web Form follow a traditional event-driven development model. Asp.Net MVC is a lightweight and follows MVC.
2. Asp.Net Web Form has server controls. Asp.Net MVC has HTML helpers.
3. Asp.Net Web Form has file-based URLs means file name exist in the URLs must have its physical existence. Asp.Net MVC has route-based URLs means URLs are divided into controllers and actions and moreover it is based on controller not on physical file.
4. Asp.Net Web Form follows Web Forms Syntax. Asp.Net MVC follow customizable syntax (Razor as default).
5. In Asp.Net Web Form, Web Forms(view) are tightly coupled to Code behind(logic). In Asp.Net MVC, Views and logic are kept separately.
6. Asp.Net Web Form has Master Pages for a consistent look and feels. Asp.Net MVC has Layouts for a consistent look and feels.

16) Action Methods

All the public methods of the Controller class are called Action methods.

They are like any other normal methods with the following restrictions:

1. Action method must be public. It cannot be private or protected
2. Action method cannot be overloaded
3. Action method cannot be a static method.

C# ADO.NET SqlDataAdapter

- **SqlDataAdapter** Class is a part of the C# ADO.NET Data Provider and it resides in the **System.Data.SqlClient** namespace.
- SqlDataAdapter provides the communication between the Dataset and the SQL database. We can use SqlDataAdapter Object in combination with DataSet Object.
- DataAdapter provides this combination by mapping Fill method, which changes the data in the DataSet to match the data in the data source, and Update, which changes the data in the data source to match the data in the DataSet.

First() vs FirstOrDefault()

- The major difference between First and FirstOrDefault is that First() will throw an exception if there is no result data for the supplied criteria whereas FirstOrDefault() will return the default

value (null) if there is no result data.

- First() will throw an exception if there is no result data, as you can see below.
- FirstOrDefault() returns the default value (null) if there is no result data and you can see the result in the following screen.

WCF

WCF stands for Windows Communication Foundation. It is basically used to create a distributed and interoperable Application. WCF Applications came into the picture in .Net 3.0 Framework. This is a framework, which is used for creating Service oriented Applications. You can send the data asynchronously from one end point to another.

Fundamentals of WCFMessage

Message is the communication unit, as it is in the form of an envelop. The transmission of the data from the client to Service and Service to client is being done by envelop. The envelop or message has the sections, given below-

- Header
- Body

By default Header and fault are disabled but Body is responsible for the data transmission or data exchanging.

Header is useful to send some data from client to Server. Suppose we want to send user name from each request but don't want to send it by an argument, we can easily add it into message header.

Endpoint

Endpoint is a very essential part of WCF Application, as it describes the address of Web Service from where a user can receive and send the message. It also specifies the communication mechanism of how the message will be sent or received.

End point consists of three things, which are A,B,C and each of them have a question mark.

- Address (Where?)
- Binding (how?)
- Contract (What?)

Endpoint = A + B + C Address is the address of WCF Service, where the Service is hosted? It gives the exact URL of Web Service, where the Service hosts the pattern of URL, which is- *Scheme://domain/[:port]/path* net.tcp://localhost:1234/MyService http://localhost:1234/MyService

Binding

It describes the way or mechanism by which the user will communicate with Web Service. It constitutes some binding element, which creates the structure of communication such as some transport protocols like HTTP, TCP etc. Message format or security techniques etc.

Contract

Contract is the third important question. What functionality and operation is being provided by the service is called contract. It specifies what functionality and operations are need to be exposed to the client. It is the interface name which has all operation that need to be exposed.

Hosting

Hosting is the important thing in WCF Application and it makes WCF Application different from other distributed Applications. WCF supports following types of hosting-

- IIS Hosting
- Self hosting
- WAS hosting

SOAP

SOAP stands for Simple Object Access Protocol. It is not a transport protocol but an XML based message protocol.

Advantages of WCF

- WCF provides better reliability and security compared to ASMX Web services.
- In WCF, there is no need to make much of a change to code to use the security model and alter the binding.
- Small changes in the configuration file will match your requirements.
- WCF provides interoperability between services.



ViewModel



In ASP.NET MVC, **ViewModel** is a class that contains the fields which are represented in the strongly-typed view. It is used to pass data from controller to strongly-typed view

Key Points about ViewModel

1. ViewModel contain fields that are represented in the view (for LabelFor,EditorFor,DisplayFor helpers)
2. ViewModel can have specific validation rules using data annotations or IDataErrorInfo.
3. ViewModel can have multiple entities or objects from different data models or data source.

ASP.NET Life Cycle Events

 Page Event	 Typical Use
<u>PreInit</u>	This event is raised after the start stage is complete and before the initialization stage.
<u>Init</u>	This event occurs after all controls have been initialized.We can use this event to read or initialize control properties.
<u>InitComplete</u>	This event occurs at the end of the page's initialization stage.We can use this event to make changes to view state that we want to make sure are persisted after the next postback.
<u>PreLoad</u>	This event is occurs before the post back data is loaded in the controls.
<u>Load</u>	This event is raised for the page first time and then recursively for all child controls.
<u>Control events</u>	This event is used to handle specific control events such as Button control' Click event.


 Page Event	 Typical Use
<u>LoadComplete</u>	This event occurs at the end of the event-handling stage. We can use this event for tasks that require all other controls on the page be loaded.
<u>PreRender</u>	This event occurs after the page object has created all controls that are required in order to render the page.
<u>PreRenderComplete</u>	This event occurs after each data bound control whose DataSourceID property is set calls its DataBind method.
<u>SaveStateComplete</u>	It is raised after view state and control state have been saved for the page and for all controls.
<u>Render</u>	This is not an event; instead, at this stage of processing, the Page object calls this method on each control.
<u>Unload</u>	This event raised for each control and then for the page.

View State

- ViewState stores data on single page
- ViewState is client side state management technique
- Session stores data on whole website pages
- Session is a server side state management technique

State management technique


ASP.NET State Management Techniques

Hello Friends this is my first article and I am trying to provide as much knowledge as I can through this article. In today's world everything is becoming Internet / Web based. Whatever work it may be, whether related to Banking, Education,  <https://www.c-sharpcorner.com/UploadFile/78d182/Asp-Net-state-manageme-nt-techniques/>



ADO Event Handlers


ASP.NET - Event Handling

An event is an action or occurrence such as a mouse click, a key press, mouse movements, or any system-generated notification. A process communicates through events. For example, interrupts are system-generated events. When  https://www.tutorialspoint.com/asp.net/asp.net_event_handling.htm

Page prerender event handled.
Page load event handled.
Page post back event handled.
Button click event handled.
Page prerender event handled.

Annotations

DataAnnotations In Depth

The namespace System.ComponentModel.DataAnnotations, has a group of classes, attributes and methods, to make validations in our .NET applications. In the Microsoft world, there are technologies such as WPF, Silverlight, ASP MVC,  <https://www.c-sharpcorner.com/article/dataannotations-in-depth/>



DbContext vs DbSet

a DbContext corresponds to your database (or a collection of tables and views in your database) whereas a DbSet corresponds to a table or view in your database. So it makes perfect sense that you will get a combination of both!

You will be using a DbContext object to get access to your tables and views (which will be represented by DbSet's) and you will be using your DbSet's to get access, create, update, delete and modify your table data.

If you have 10 tables in your database and your application works with 5 of them (let us call them Table1 - Table 5) it would make sense to access it using a MyAppContext object where the MyAppContext class is defined thus:

```
public class MyAppContext : DbContext
{
    public MyAppContext () : ;



    public DbSet<Table1> Table1 { get; set; }
    public DbSet<Table2> Table2 { get; set; }
    public DbSet<Table3> Table3 { get; set; }
    public DbSet<Table4> Table4 { get; set; }
    public DbSet<Table5> Table5 { get; set; }
}
```

Note that, for instance, the identifier Table1 is used both as the name of a type and as a name of a property in the defined context type. What you see above is quite typical. An example of a class that corresponds to a table schema is given below:

```
public class Table1
{
    public int Id {get; set;}
    public string AStringField {get; set;}
    //etc.
}
```

Return Type of MVC Controller action method

MVC framework includes various **Result** classes, which can be returned from an action method. The result classes represent different types of responses, such as HTML, file, string, JSON, javascript, etc. The following table lists all the result classes available in ASP.NET MVC.

 Result Class	 Description
<u>ViewResult</u>	Represents HTML and markup.
<u>EmptyResult</u>	Represents No response.
<u>ContentResult</u>	Represents string literal.
<u>FileContentResult/ FilePathResult/ FileStreamResult</u>	Represents the content of a file.
<u>JavaScriptResult</u>	Represent a JavaScript script.
<u>JsonResult</u>	Represent JSON that can be used in AJAX.
<u>RedirectResult</u>	Represents a redirection to a new URL.

 Result Class	 Description
<u>RedirectToRouteResult</u>	Represent another action of same or other controller.
<u>PartialViewResult</u>	Returns HTML from Partial view.
<u>HttpUnauthorizedResult</u>	Returns HTTP 403 status.

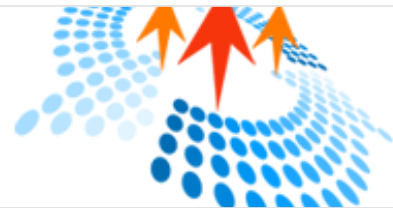
The `ActionResult` class is a base class of all the above result classes, so it can be the return type of action method that returns any result listed above. However, you can specify the appropriate result class as a return type of action method.

Event handling

Event Handling In ASP.NET

Introduction In ASP.NET Web forms, we are using an event-based model. We can create every server control event. For example, in the case of the button control, we can create an event handler to handle the click event. Even the ASPX page has

 <https://www.c-sharpcorner.com/article/event-handling-in-asp-net/>



What will controller do?

- A controller is responsible for controlling the way that a user interacts with an MVC application.
- A controller contains the flow control logic for an ASP.NET MVC application.
- A controller determines what response to send back to a user when a user makes a browser request.

Entry point of ASP.net Core apps

Startup.cs file is entry point, and it will be called after Program.cs file is executed at application level. It handles the request pipeline. Startup class triggers the second the application launches.

What is Program.cs ?

Program.cs is where the application starts. Program.cs class file is entry point of our application and creates an instance of IWebHost which hosts a web application.

What is Startup file?

startup.cs is mandatory, it can be decorated with any access modifier like public, private, internal. Multiple Startup classes are allowed in a single application. ASP.NET Core will select the appropriate class based on its environment.

Dependency injection (DI) and implementation

Dependency injection (DI) is a technique for achieving loose coupling between objects and their collaborators, or dependencies. Most often, classes will declare their dependencies via their constructor, allowing them to follow the Explicit Dependencies Principle. This approach is known as "constructor injection".

To implement dependency injection, we need to configure a DI container with classes that is participating in DI. DI Container has to decide whether to return a new instance of the service or

provide an existing instance. In startup class, we perform this activity on ConfigureServices method.

The lifetime of the service depends on when the dependency is instantiated and how long it lives. And lifetime depends on how we have registered those services.

The below three methods define the lifetime of the services,

ASP.net core has built-in support for Dependency Injection. We can configure services to DI container using this method. Following ways are to configure method in startup class.

AddTransient

Transient objects are always different; a new instance is provided to every controller and every service.

Scoped

Scoped objects are the same within a request, but different across different requests.

Singleton

Singleton objects are the same for every object and every request.

☰ Service Type	Aa In the scope of a given http request	☰ Across different http requests
Transient	<u>New Instance</u>	New Instance
Scoped	<u>Same Instance</u>	New Instance
Singleton	<u>Same Instance</u>	Same Instance

Microservice Workflow

A microservice architecture splits a system into individual services, each focused on a single business capability

We can take file as input in microservice

Request PipeLine

The Request Pipeline is the mechanism by which requests are processed beginning with a Request and ending with a Response.

The pipeline specifies how the application should respond to the HTTP request. The Request arriving from the browser goes through the pipeline and back

The individual components that make up the pipeline are called Middleware.

What is Middleware

Middleware is a software component that hooks into the request pipeline to handle web requests and generate responses.

Each middleware Process and manipulates the request as it is received from the previous middleware.

It may decide to call the next middleware in the pipeline or send the response back to the previous middleware (terminating the pipeline)

What is Kestrel?

The Kestrel is open-source, cross-platform, event-driven, asynchronous I/O based HTTP server. It is developed to host ASP.NET Core applications on any platform. It is included by default in the ASP.NET Core applications.

Testing:

1. What is NUnit?

NUnit is a unit-testing framework for .NET applications in which the entire application is isolated into diverse modules.

Each module is tested independently to ensure that the objective is met.

2. What is Moq? Why do we use mocking?

Moq is used to mock/fake the objects that simulate or mimic a real object. Moq can be used to mock both classes and interfaces.

In mock we replace external dependencies with pseudo-dependencies to test whether the units of code function as expected.

```
Mock<TypeToMock> mockObjectType=new Mock<TypeToMock>();
```

Mocking is to avoid database interaction.

3. What is TDD?

TDD stands for Test-driven development. It is a process of modifying the code in order to pass a test designed previously.

TDD is to write and correct the failed tests before writing new code. This helps to avoid duplication of code as

we write a small amount of code at a time in order to pass tests.

Test-Driven development is a process of developing and running automated test before actual development of the application.

Hence, TDD sometimes also called as Test First Development.

4. What is DDT?

Data Driven Testing is a software testing method in which test data is stored in table or spreadsheet format.

DDT allows testers to input a single test script that can execute tests for all test data from a table and expect the test output in the same table.

It is also called table-driven testing or parameterized testing.

Data Driven Testing is a Test design and execution strategy where the test scripts read test data from data sources (file or database)

such as ADO objects, ODBC sources, CSV files, etc. rather than using hard-coded values.