**1) What is Output Caching in MVC ?**

->The main objective of making use of Output Caching is to dramatically improve the performance of an ASP.NET MVC Application. It enables us to cache the content returned by any controller method so that the same content does not need to be generated each time the same controller method is invoked. Output Caching has huge advantages, such as it reduces server round trips, reduces database server round trips, reduces network traffic, etc.

**2)What Are The Main Razor Syntax Rules**

* ->Razor code blocks are enclosed in @{ … }
* Inline expressions (variables and functions) start with @
* Code statements end with a semicolon
* Variables are declared with the var keyword
* Strings are enclosed with quotation marks
* C# code is case sensitive
* C# files have the extension .cshtml

**3) What Is Partial View In MVC ?**

->Partial view is a reusable view, which can be used as a child view in multiple other views. It eliminates duplicate coding by reusing same partial view in multiple places. You can use the partial view in the layout view, as well as other content views.

To start with, let's create a simple partial view for the following navigation bar for demo purposes. We will create a partial view for it, so that we can use the same navigation bar in multiple layout views without rewriting the same code everywhere.

Link- <https://www.tutorialsteacher.com/mvc/partial-view-in-asp.net-mvc>

**4) What is GET and POST Action Types?**

-> **GET Action Type:** GET is used to request data from a specified resource. With all the GET requests, we pass the URL, which is compulsory; however, it can take up the following overloads.

**POST Action Type**: The POST is used to submit data to be processed to a specified resource. With all the POST requests, we pass the URL, which is essential and the data. However, it can take up the following overloads.

**5) What are the filters and Types of filters in MVC ?**

-> There are situations in which we have an implementation that will be reused in many places that is not confined to a single place or method. This is fulfilled by the **Filters** in MVC. This is a very good concept introduced in MVC. The implementation that is said above is called cross-cutting concerns. Thus, in simple terms, this adds extra logic to be implemented into the request being processed. Some of the examples of cross-cutting conerns are Authorization & Output Caching.

a) Authorization Filters

b) Action Filter

c) Result Filter

d) Error Filter

e) Authentication Filter

link - <https://www.c-sharpcorner.com/UploadFile/0ef46a/filters-in-mvc/>

**6) Explain What is Database First Approach in MVC using Entity Framework?**

-> Database First Approach is an alternative or substitutes to the Code First and Model First approaches to the Entity Data Model. The Entity Data Model creates model codes (classes, properties, DbContext, etc.) from the database in the project and that class behaves as the link between database and controller.

There are the following approaches, which are used to connect the database with the application.

* Database First
* Model First
* Code First

**7) What are the Advantages of MVC?**

* -> **Multiple view support**: Due to the separation of the model from the view, the user interface can display multiple views of the same data at the same time.
* **Change Accommodation**: User interfaces tend to change more frequently than business rules (different colors, fonts, screen layouts, and levels of support for new devices such as cell phones or PDAs)
* **SoC**– Separation of Concerns: Separation of Concerns is one of the core advantages of ASP.NET MVC. The MVC framework provides a clean separation of the UI, Business Logic, Model or Data.
* **More Control**: The ASP.NET MVC framework provides more control over HTML, JavaScript, and CSS than the traditional Web Forms.
* **Testability**: ASP.NET MVC framework provides better testability of the Web Application and good support for the test-driven development too.
* **Lightweight**: ASP.NET MVC framework doesn’t use View State and thus reduces the bandwidth of the requests to an extent.

**8) What are Action Filters in MVC ?**

-> Action Filters are additional attributes that can be applied to either a controller section or the entire controller to modify the way in which action is executed. These attributes are special .NET classes derived from system attributes, which can be attached to classes, methods, properties, and fields.

**9) What is the difference between ViewData and ViewBag ?**

-> ViewBag is nothing but the collection of ViewData. ViewBag Simplifies the syntaxex of ViewData. In a ViewData syntax we need to put the square brackets and double quotes …etc.ViewBag uses the dynamic property of C# 4.0 . ViewData requires Typecasting ViewBag doesn’t requires Typecasting.

**10) What is TempData in MVC ?**

-> ViewBag and ViewData does not persisit the data when we move from one action method to other action method. For that we have to use TempData.

**11) What is Session Variables in MVC ?**

-> Session Variables are used when we want to persist the data when moving from controller to controller, Controler to view & View to Controller. Means throughout the session.

**12) What is Validation Summary In MVC ?**

-> The Validation Summary helper method generates an unordered list (UL element) of validation messages that are in the Model State Dictionary object.

The Validation Summary can be used to display all the error messages for all the fields. It can also be used to display custom error messages.

**19) Explain What is Routing in MVC ?**

-> **Routing** is a mechanism in **MVC** that decides which action method of a controller class to execute. Without **routing** there's no way an action method can be mapped. to a request. **Routing** is a part of the**MVC** architecture so ASP.NET **MVC** supports**routing** by default.

Routing means mapping of incoming URL to Controller and Action Method

There are two types of Routing:

.**Conventonal Routing**:

In This we define the default route in startup.cs file

**Attribute based Routing**:

In this we can define the route above the controller or Action Method.

In MVC Application when there are more than one Action Methods with the Same name and parameters then the browser is getting confused which action method is to execute to solve this attribute routing is used In attribute routing we give URI t each action Method with the help of [Route] attribute so that each Action Method can be invoked without any confliction.

RouteTable is a collection of routes that is stored in **RouteConfig.** **cs file in App\_Start folder of the application**.

**Can we use Attribute routing and convention base routing at same time in application?**

We can use Convention based Routing and Attribute Routing in the same project. Be sure attribute routing should be defined first to convention-based routing.

**Can we define multiple routes for same action method?**

Yes, We can use multiple URLs to the same action with the use of a routing table.

**15) What does the MVC pattern define with 3 logical layers ?**

The MVC model defines web applications with 3 logic layers:

* The business layer (Model logic)
* The display layer (View logic)
* The input control (Controller logic)

The Model is the part of the application, which only handles the logic for the application data. Regularly, the model objects retrieve data (as well as store data) from a database. The View is the part of the application, which takes care of the display of the data.

Most often, the views are created from the model data, although there are other, more complicated methods of creating views.

The Controller, as the name implies, is the part of the application that handles user interaction.

**17) List The Various return types of the controller action method ?**

-> There are total nine return types we can use to return results from a controller to view.The base type of all these result types is ActionResult.

1. ViewResult (View): This return type is used to return a webpage from an action method.
2. PartialviewResult (Partialview): This return type is used to send a part of a view which will be rendered in another view.
3. RedirectResult (Redirect): This return type is used to redirect to any other controller and action method depending on the URL.
4. RedirectToRouteResult (RedirectToAction, RedirectToRoute): This return type is used when we want to redirect to any other action method.
5. ContentResult (Content): This return type is used to return HTTP content type like text/plain as the result of the action.
6. jsonResult (JSON): This return type is used when we want to return a JSON message.
7. javascriptResult (javascript): This return type is used to return JavaScript code that will run in the browser.
8. FileResult (File): This return type is used to send binary output in response.
9. EmptyResult: This return type is used to return nothing (void) in the result.

**18) Explain What is MVC application Life Cycle ?**

->  Any web application has two main execution steps, first understanding the request and depending on the type of the request sending out an appropriate response. MVC application life cycle is not different it has two main phases, first creating the request object and second sending our response to the browser.

**Step 1 Fill route: -**MVC requests are mapped to route tables which in turn specify which controller and action to be invoked. So if the request is the first request the first thing is to fill the route table with routes collection. This filling of route table happens in the global.asax file.

**Step 2 Fetch route:-**Depending on the URL sent "UrlRoutingModule" searches the route table to create "RouteData" object which has the details of which controller and action to invoke.

**Step 3 Request context created: -**The "RouteData" object is used to create the "RequestContext" object.

**Step 4 Controller instance created: -**This request object is sent to "MvcHandler" instance to create the controller class instance. Once the controller class object is created it calls the "Execute" method of the controller class.

**Creating Response object: -**This phase has two steps executing the action and finally sending the response as a result to the view.

**Step 5 Execute Action: -**The "ControllerActionInvoker" determines which action to executed and executes the action.

**Step 6 Result sent: -**The action method executes and creates the type of result which can be a view result , file result , JSON result etc.

**21) Whats New in the latest version of MVC ?**

-> In MVC 6 Microsoft removed the dependency of System.Web.Dll from MVC6 because it is extremely expensive, as it typically used to consume 30K of memory per request and response, whereas now MVC 6 only requires 2K of memory per request and the response is really small memory consumption.

**22) What are html helpers in MVC ?**

-> An html helper is a method that is used to render html content in the view.

For eg- To produce the html for textbox we write as

<input type=”text” name=”firstname” id=”firstname” />

Instead of writing all this code we can simply use “Textbox” html helper as

@Html.TextBox(“firstname”)

We use html helpers like chechbox,dropdownlist,gender etc

**24) What are the Benefits of area in MVC ?**

1. -> Allows us to organize models, views, and controllers into separate functional sections of the application, such as administration, billing, customer support and much more.
2. Easy to integrate with other Areas created by another.
3. Easy for unit testing.

**27) What is MVC (Model View Controller) ?**

-> Model–view–controller (MVC) is a software architectural pattern for implementing user interfaces. It divides a given software application into three interconnected parts, to separate the internal representation of information from the way that information is presented to or accepted by the user.

**28) What is Bundling and minification in MVC ?**

-> Bundling and minification are two new techniques introduced to improve request load time. It improves load time by reducing the number of requests to the server and reducing the size of requested assets (such as CSS and JavaScript).

**Bundling**: It lets us combine multiple JavaScript (.js) files or multiple cascading style sheet (.css) files so that they can be downloaded as a unit, rather than making individual HTTP requests.

**Minification**: It extracts the whitespace and performs other types of compression to make the downloaded files as small as possible. At runtime, the process recognizes the agent of the user, for example, IE, Mozilla, etc. and then removes whatever is specific to Mozilla when the request comes from IE.

**29) How do you implement forms authentication in MVC ?**

-> Authentication is giving access to the user for a specific service by verifying his/her identity using his/her credentials like username and password or email and password. It assures that the correct user is authenticated or logged in for a specific service and the right service has been provided to the specific user based on their role that is nothing but authorization.

**30) Explain the concept of MVC Scafolding ?**

-> ASP.NET Scaffolding is a code generation framework for ASP.NET Web applications. Visual Studio 2013 includes pre-installed code generators for MVC and Web API projects. 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.  
Scaffolding consists of page templates, entity page templates, field page templates, and filter templates. These templates are called Scaffold templates and allow you to quickly build a functional data-driven Website.

**31) What is Interface**

Class : It’s a user defined data type

Interface : This is also an user defined data type

Class : containes Non Abstract Methods (Methods with Method body)

Abstract Class : Contains Abstract Methods(Methods without method body) as well as Non Abstract Methods(Methods with method body)

Interface : Contains only Abstract Methods(Methods without method body)

Generally a class inherits from another class to consume the members of its parent, where as if class is inheriting from an interface it is to implement the members of its parent.

Note : A class can inherit from a class and interface at a time.

The default scope the members of an interface is public where as its private in case of class.

By default every member of an interface is abstract so we don’t require to use abstract modifier on it again just like we do in a case of abstract class.

We cannot declare any variables under an interface

If required an interface can inherit from another interface

Every member of an interface should be implemented under the child class of the interface without fail, but while implementing we don’t require to use override modifier just like we have done in case of abstract class.

By Using Classes we cant achieve multiple inheritance because it creates ambiguity but by using interfaces we can achieve multiple inheritance

**What is Abstract Class**

A method without method body is an Abstract Method

We use astract classes for code reusability. Common methods we can declare in astract classes as an Abstract methods and we can implement in our child classes as we want by using Override keyword.

**What is Delegates**

A delegate holds the reference of a method and then calls the method for execution.

A case when we need to pass method as a parameter then we can use delegete.

Eg :- to print the current number of for loop.

To call a method using delegate we have three steps:

* 1. Define a delegate
  2. Instantiating a delegate
  3. Call the delegate

**What Is Middleware in MVC?**

Middleware handles the incoming http request and outgoing http response

Middlewares are defined in Configure part in startup class.

**What is the difference between app.use and app.run and app.map in middleware?**

Run will end the request, and app. Use will pass the request to next middleware.  The Map extension method is used to match request delegates based on a request's path.

**What is Razor View in MVC?**

In Razor pages we can write C# code in between the html code

We can switch to C# code by using @symbol

**What is CLR in ASP.NET Core**

CLR is a run time environment provided in asp.net core for program execution

It contains JIT compiler, Garbage collector, Thread Management, Exception Handling

**JIT Compiler** : Just in time compiler normal C# compiler converts source code to byte code and then JIT Compiler converts this byte code to machine code

**Threading** : CLR provides threading functionality, it creates multiple threads for execution.

In multithreading if a single task is in a process and the user has to wait then at that time it executes the background threads.

**What is Kestral in ASP.NET Core?**

Kestrel is a cross platform web server for running the web applications.

**What is Web-Listener?**

WebListener is also a web server for ASP.NET Core that runs only on Windows.

**What is implementation and interface inheritance?**

When a class (type) is derived from another class(type) such that it inherits all the members of the base type it is Implementation Inheritance.  
When a type (class or a struct) inherits only the signatures of the functions from another type it is Interface Inheritance.  
In general Classes can be derived from another class, hence support Implementation inheritance. At the same time Classes can also be derived from one or more interfaces. Hence they support Interface inheritance.

**Define Overriding?**

Overriding is a concept where a method in a derived class uses the same name, return type, and arguments as a method in its base class. In other words, if the derived class contains its own implementation of the method rather than using the method in the base class, the process is called overriding.

**What is an Interface?**

An interface is a standard or contract that contains only the signatures of methods or events. The implementation is done in the class that inherits from this interface. Interfaces are primarily used to set a common standard or contract.

**What is Namespace?**

namespaces are used to logically arrange classes, structs, interfaces, enums and delegates.

**What is enum?**

***Enumeration (or enum)*** is a [value data type](https://www.geeksforgeeks.org/c-data-types-2/) in C#. It is mainly used to assign the names or string values to integral constants, that make a program easy to read and maintain.

Here enum with name **month** is created and its data members are the name of months like jan, feb, mar, apr, may.

namespace ConsoleApplication1 {

// making an enumerator 'month'

enum month

{

    // following are the data members

    jan,

    feb,

    mar,

    apr,

    may

}

**What is Constructor?**

 A **constructor** is a special type of member function of a class which initializes objects of a class.

In the following example, a class named Taxi is defined by using a simple constructor. This class is then instantiated with the [new](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/new-operator) operator. The Taxi constructor is invoked by the new operator immediately after memory is allocated for the new object.

public class Taxi

{

public bool IsInitialized;

public Taxi()

{

IsInitialized = true;

}

}

class TestTaxi

{

static void Main()

{

Taxi t = new Taxi();

Console.WriteLine(t.IsInitialized);

}

}

**What is the base class for the controller class?**

System. Web. Mvc. Controller

**What is the base class for all Return Results?**

ActionResult is the base class for all return results

**What is difference between WEB API Controller and MVC Controller?**

Web Api controller derives from SYSTEM.WEB.HTTP.APIController while MVC Controller derives from SYSTEM.WEB.MVC.CONTROLLER

Web API Controller does not give view support while MVC Controller gives View support.