MVC Interview Questions and Answers

What is ASP.NET MVC?

ASP.NET MVC is a web application Framework. It is light weight and highly testable Framework. MVC separates application into three components. Model, View and Controller.

Can you explain Model, View and Controller in MVC?

Model- Model is responsible for maintaining application data and business logic.

View- View is a user interface of the application, which displays the data.

Controller-Controller handles user's requests and renders appropriate View with Model data.

What are the new features added in MVC 5?

1. **One ASP.NET**: The Web MVC project templates integrate seamlessly with the One ASP.NET experience. You can customize your MVC project and configure authentication using the One ASP.NET project creation wizard.
2. **ASP.NET Identity**: The MVC project templates have been updated to use ASP.NET Identity for authentication and identity management.
3. **Bootstrap**: The MVC project template has been updated to use Bootstrap to provide a sleek and responsive look and feel that you can easily customize.
4. **Authentication filters**: Authentication filters are a new kind of filter in ASP.NET MVC that run prior to authorization filters in the ASP.NET MVC pipeline and allow you to specify authentication logic per-action, per-controller, or globally for all controllers. Authentication filters process credentials in the request and provide a corresponding principal.
5. **Filter overrides**: You can now override which filters apply to a given action method or controller by specifying an override filter. Override filters specify a set of filter types that should not be run for a given scope (action or controller).
6. **Scaffolding**: Scaffolding is a code generation framework for ASP.NET MVC applications.
7. **Attribute Routing:** Routing is how ASP.NET MVC matches a URI to an action. MVC 5 supports a new type of routing, called attribute routing.

What are new features added in MVC 6?

1. ASP.NET MVC and Web API has been merged into one.
2. Side by side deploy the runtime and framework with your application
3. No need to recompile for every change. Just hit save and refresh the browser.
4. Dependency injection is inbuilt and part of MVC.
5. Everything packaged with NuGet, Including the .NET runtime itself.
6. New JSON based project structure.
7. Compilation done with the new Roslyn real-time compiler.

Can you explain the page life cycle of MVC?

App initialization→ Routing→ Instantiate and execute controller→ Locate and invoke controller action→ Instantiate and render view.

What is Routing in MVC?

Routing is a matching mechanism of incoming requests to the URL patterns which are registered in route table. UrlRoutingModule class is used for the same process.

Type of Routing

1. Convention-Based Routing
2. Attribute Routing

```
 public class RouteConfig
    {
        public static void RegisterRoutes(RouteCollection routes)
        {
            routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

            routes.MapRoute(
                name: "Default",
                url: "{controller}/{action}/{id}",
                defaults: new { controller = "Home", action = "Index", id =
UrlParameter.Optional}
            );
        }
    }
```

What is Attribute Routing in MVC?

Attribute routing is introduced in MVC5. Attributes are being used to define the routes. This type of routing gives more control over classic URI Routing. Attribute Routing can be defined at controller level or at Action level. ASP.NET Web API supports attribute routing.

```
public static void RegisterRoutes(RouteCollection routes)
        {
            routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

            //Enable attribute routing
            routes.MapMvcAttributeRoutes();

            routes.MapRoute(
                name: "Default",
                url: "{controller}/{action}/{id}",
                defaults: new { controller = "Home", action = "Index", id =
UrlParameter.Optional }
            );
        }
```

Attribute routing at Action Level

```
[Route("{productId:int}/{productTitle}")]
public ActionResult Show(int productId) { … }
```

Attribute routing at Controller Level

```
[RouteArea("Admin")]
[RoutePrefix("menu")]
[Route("{action}")]
public class MenuController : Controller
{
    // eg: /admin/menu/login
    public ActionResult Login() { … }

    // eg: /admin/menu/show-options
    [Route("show-options")]
    public ActionResult Options() { … }

    // eg: /stats
    [Route("~/stats")]
    public ActionResult Stats() { … }
}
```

How route table has been created in ASP.NET MVC?

The method RegisterRoutes() is used for registering the routes which will be added in Application_Start() method of global.asax file, which is fired when the application is loaded or started.

What is Razor View Engine?

Razor is the first major update to render HTML in MVC 3. Razor view engine in asp.net MVC is syntax that allows you to write server side code on view. It helps to combine code and html in fluid manner.

What are the advantages of ASP.NET MVC over ASP.NET?

1. ASP.NET MVC provides clean separation of concern among Model, View and controller.
2. ASP.NET MVC enables the full control over the rendering HTML.
3. It is easily integrate with JavaScript frameworks.
4. It follow stateless nature design of web application.
5. No ViewState and PostBack events
6. ASP.NET MVC improves reusability of model and views. We can have multiple views which can point to the same model and vice versa.
7. ASP.NET MVC improves structuring of the code.
8. It is easy to UNIT Test.

What is the different between ASP.NET Web Form and ASP.NET MVC?

ASP.NET WEB FORM	ASP.NET MVC
Rich server controls encapsulate flexible rendering	Separation of concerns with enhanced performance.
Event driven and stateful programming	Web stateless behavior with full control over HTML.
Productivity and rapid application development	Routing with rest and SEO friendly.
	Reusable components and test driven deployment

What is separation of concerns in ASP.NET MVC?

It is a process of breaking the program into various distinct features which overlaps in functionality as little as possible. MVC pattern concerns on separating the content from presentation and data-processing from content.

What is the use of ViewModel in MVC?

ViewModel is a plain class with properties, which is used to bind it to strongly typed view. ViewModel can have the validation rules defined for its properties using data annotations.

What are Actions in MVC?

Actions are the methods in controller class which is responsible for returning the view or json data. Action will mainly have return type "ActionResult" and it will be invoked from method InvokeAction () called by controller.

mvcaction4 and tab shortcut for MVC ActionResult

What is the importance of NonActionAttribute?

All public methods of a controller class are treated as the action method if you want to prevent this default method then you have to assign the public method with NonActionAttribute.

Which are the important namespaces used in MVC?

System.Web.Mvc
System.Web.Mvc.Ajax
System.Web.Mvc.Html
System.Web.Mvc.Async

What are areas in MVC?

Area allows us to partition large application into smaller units where each unit contains separate MVC folder structure, same as default MVC folder structure.

What is ViewData?

ViewData is a dictionary of objects that is derived from ViewDataDictionary class and accessible using strings as keys. It is useful in transferring data from Controller to View.

```
public ActionResult Index()
{
            List<string> employee = new List<string>();
            employee.Add("Afra Framin");
            employee.Add("Farhan Ahmed");
            employee.Add("Fayaz Ahmed");
            employee.Add("Abdul Vaheed");

            ViewData["EmloyeeList"] = employee;
            return View();
 }

<ul>
            @foreach (var emp in ViewData["EmloyeeList"] as List<string>)
            {
                <li>@emp</li>
            }
</ul>
```

What is ViewBag?

ViewBag is actually a wrapper around ViewData. ViewBag transfers data from the controller to the view, ideally temporary data which in not included in a model.

```
public ActionResult Index()
{
            List<string> employee = new List<string>();
            employee.Add("Afra Framin");
            employee.Add("Farhan Ahmed");
            employee.Add("Fayaz Ahmed");
            employee.Add("Abdul Vaheed");

            ViewBag.EmloyeeList=employee;
            return View();
 }

<ul>
   @foreach (var emp in ViewBag.EmloyeeList)
            {
                <li>@emp</li>
            }
</ul>
```

What is TempData?

TempData can be used to store data between two consecutive requests. TempData values will be retained during redirection.

```
public ActionResult Index()
{
            List<string> employee = new List<string>();
            employee.Add("Afra Framin");
            employee.Add("Farhan Ahmed");
            employee.Add("Fayaz Ahmed");
            employee.Add("Abdul Vaheed");

            TempData["EmloyeeList"] = employee;
            return View();
 }

<ul>
            @foreach (var emp in TempData["EmloyeeList"] as List<string>)
            {
                <li>@emp</li>
            }
</ul>
```

What is PartialView in MVC?

A partial view is a Razor markup file (*.cshtml*) that renders HTML output within another markup file's rendered output.

```
@Html.Partial("_Employee", item)

Html.RenderPartial("_Employee", item);
```

When to use PartialView?

- Break up large markup files into smaller components.
- Reduce the duplication of common markup content across markup files.
- Partial views shouldn't be used to maintain common layout elements. Common layout elements should be specified in _Layout.cshtml files.

What is T4 template?

T4 (Text template transformation toolkit) is template based code generation engine. So you can go and write C# code in T4 templates (.tt is the extension) files and those c# codes execute to generate the file as per the written C# logic.

What is Layout in MVC?

Layout pages are similar to master pages in traditional web forms. This is used to set the common look across multiple pages.

What is bundling in MVC?

Bundling is a technique to improve performance by reducing the number of request to the server. To add a new bundle we use BundleConfig file. In this file we use BundleCollection class which is available in System.Web.Optimization namespace.

What is minification in MVC?

Minification is the process of removing unnecessary data without affecting functionality.it removes comments, extra spaces convert large variable name to small name.

What are the file extensions for razor views?

For razor views the file extensions are

- .cshtml: If C# is the programming language
- .vbhtml: If VB is the programming language

What is ViewStart?

Razor View Engine introduced a new layout named _ViewStart which is applied on all view automatically. Razor View Engine firstly executes the _ViewStart and then start rendering the other view and merges them.

What are HTML Helpers in MVC?

HTML Helpers are like controls in traditional web forms. But HTML helpers are more lightweight compared to web controls as it does not hold ViewState and events. HTML Helpers returns the HTML string which can be directly rendered to HTML page. Custom HTML Helpers also can be created by overriding "HtmlHelper" class.

There are two types of html helper in MVC

1. Standard HTML Helper
2. Strongly Type HTML Helper

Standard HTML Helper	Strongly Type HTML Helpers
@using(Html.BeginForm){}	
@Html.ActionLink	@Html.HiddenFor
@Html.Hidden	@Html.LabelFor
@Html.Label	@Html.TextBoxFor
@Html.TextBox	@Html.PasswordFor
@Html.Password	@Html.DropdownListFor
@Html.DropdownList	@Html.TextAreaFor
@Html.TextArea	@Html.ListBoxFor
@Html.ListBox	@Html.CheckBoxFor
@Html.CheckBox	@Html.RadioButtonFor
@Html.RadioButton	@Html.LabelFor

How to create custom helper method in MVC?

```
  public static class MyHeleprs
    {
        public static MvcHtmlString Image(this HtmlHelper htmlHelper,
             string source, string alternativeText)
        {
            //declare the html helper
            var builder = new TagBuilder("image");
            //hook the properties and add any required logic
            builder.MergeAttribute("src", source);
            builder.MergeAttribute("alt", alternativeText);
            //create the helper with a self closing capability
            return
MvcHtmlString.Create(builder.ToString(TagRenderMode.SelfClosing));
        }
    }
```

What are AJAX Helpers in MVC?

AJAX Helpers are used to create AJAX enabled elements like as Ajax enabled forms and links which performs the request asynchronously and these are extension methods of AJAXHelper class which exists in namespace System.Web.Mvc.

URL: Which controllers Action Method to be called on while making Ajax Request is set here.

Confirm: This property display confirmation box in which you will see ok and cancel button if you click on ok button after that Ajax request is made. This confirmation box is displayed before making AJAX request.

HttpMethod: This property is used while making Ajax request which are GET or POST by default this value is set to POST.

InsertionMode: There are 3 insertion modes ("InsertAfter", "InsertBefore", "Replace"). The default value is "Replace". This specifies the way in which the data response will be retrieved from server is inserted into the target DOM element.

LoadingElementId: While AJAX request is processing if we want to display Progress bar or Loader then we can use this Property.

LoadingElementDuration: By setting this property we can control duration of LoadingElementId (Progress bar or Loader duration) for how much time it must be displayed.

UpdateTargetId: By setting UpdateTargetId of HTML elements the data from server will be added to this HTML element.

OnBegin: This AjaxOptions is called before sending Ajax call.

OnComplete: This AjaxOptions is called when Ajax call completed.

OnFailure: This AjaxOptions is called when Ajax call fails.

OnSuccess: This AjaxOptions is Ajax request completes successfully and happens before OnComplete.

Explain what is the difference between View and Partial View?

View	Partial View
MVC view contains the layout page.	Partial does not contain the layout page.
Before any view is rendered, ViewStart page is rendered.	It does not verify for a ViewStart.cshtml. We cannot put common code for a partial view within the viewStart.cshtml.page.
View might have markup tags like body, html, head, title, meta etc.	In MVC Partial view is designed specially to render within the view and just because of that it does not consist any mark up.
View is not lightweight as compare to Partial View.	We can pass a regular view to the RenderPartial method.

What are the ActionResult in MVC?

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 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.

What are Validation Annotations?

1. **Required**: Indicates that the property is a required field
2. **StringLength**: Defines a maximum length for string field
3. **Range**: Defines a maximum and minimum value for a numeric field
4. **RegularExpression**: Specifies that the field value must match with specified Regular Expression
5. **CreditCard**: Specifies that the specified field is a credit card number
6. **CustomValidation**: Specified custom validation method to validate the field
7. **EmailAddress**: Validates with email address format
8. **Phone**: Specifies that the field is a phone number using regular expression for phone numbers
9. **FileExtension**: Validates with file extension
10. **MaxLength**: Specifies maximum length for a string field
11. **MinLength**: Specifies minimum length for a string field

What is filter and type of Filters in MVC?

In MVC, many times we would like to perform some action before or after a particular operation. For achieving this functionality, ASP.NET MVC provides feature to add pre and post action behaviors on controller's action methods.

The ASP.NET MVC framework supports four different types of filters

1. **Authorization filters**: Performs authentication and authorizes before executing action method. It implements the IAuthorizationFilter attribute.
2. **Action filters**: Performs some operation before and after an action method executes. It implements the IActionFilter attribute.
3. **Result filters**: Performs some operation before or after the execution of view result. It implements the IResultFilter attribute.
4. **Exception filters**: Performs some operation if there is an unhandled exception thrown during the execution of the ASP.NET MVC pipeline. It implements the IExceptionFilter attribute.

If we have multiple filters, what's the sequence for execution?

1. Authorization filters
2. Action filters
3. Response filters
4. Exception filters

What are the level filter can be apply?

- Global level
- Controller level
- Action method level

Where do you apply global filter

```
// MvcApplication class contains in Global.asax.cs file
public class MvcApplication : System.Web.HttpApplication
{
    protected void Application_Start()
    {
        FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
    }
}

// FilterConfig.cs located in App_Start folder
public class FilterConfig
{
```

```
    public static void RegisterGlobalFilters(GlobalFilterCollection filters)
    {
        filters.Add(new HandleErrorAttribute());
    }
}
```

What are the options can be configured in AJAX helpers?

Url: This is the request URL.

Confirm:This is used to specify the message which is to be displayed in confirm box.

OnBegin:JavaScript method name to be given here and this will be called before the AJAX request.

OnComplete: JavaScript method name to be given here and this will be called at the end of AJAX request.

OnSuccess:JavaScript method name to be given here and this will be called when AJAX request is successful.

OnFailure: JavaScript method name to be given here and this will be called when AJAX request is failed.

UpdateTargetId: Target element which is populated from the action returning HTML.

What are difference between Textbox and TextBoxFor?

- @Html.TextBox() is loosely typed method whereas @Html.TextBoxFor() is a strongly typed (generic) extension method.
- TextBox() requires property name as string parameter where as TextBoxFor() requires lambda expression as a parameter.
- TextBox doesn't give you compile time error if you have specified wrong property name. It will throw run time exception.
- TextBoxFor is generic method so it will give you compile time error if you have specified wrong property name or property name changes. (Provided view is not compile at run time.)

Explain Dependency Resolution?

Dependency Resolver again has been introduced in MVC3 and it is greatly simplified the use of dependency injection in your applications. This turn to be easier and useful for decoupling the application components and making them easier to test and more configurable.

What are scaffolding templates in MVC?

Create: It creates a View that helps in creating a new record for the Model. It automatically generates a label and input field for each property in the Model.

Delete: It creates a list of records from the model collection along with the delete link with delete record.

Details: It generates a view that displays the label and an input field of the each property of the Model in the MVC framework.

Edit: It creates a View with a form that helps in editing the current Model. It also generates a form with label and field for each property of the model.

List: It generally creates a View with the help of a HTML table that lists the Models from the Model Collection. It also generates a HTML table column for each property of the Model.

What is the difference between ActionResult and ViewResult?

- ActionResult is an abstract class while ViewResult derives from the ActionResult class.
- ActionResult has several derived classes like ViewResult, JsonResult, FileStreamResult, and so on.
- ActionResult can be used to exploit polymorphism and dynamism. So if you are returning different types of views dynamically, ActionResult is the best thing. For example in the below code snippet, you can see we have a simple action called DynamicView. Depending on the flag (IsHtmlView) it will either return a ViewResult or JsonResult.

What is the different between Html.Partial and Html.RenderPartial?

- **Html.Partial()** - This method actually returns an MvcHtmlString *object (*basically stringified HTML content*)* in the location where it was specified*.*

```
@Html.Partial("_Employee", item)
```

- **Html.RenderPartial()** - This method will not actually return any values or strings and instead will write the Partial View that is requested to the Response Stream (through Response*.*Write) internally.

```
Html.RenderPartial("_Employee", item);
```

What is Output Caching in MVC?

The main purpose of using 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.

OutputCache label has a "Location" attribute and it is fully controllable. Its default value is "Any", however there are the following locations available; as of now, we can use any one.

- Any
- Client
- Downstream
- Server
- None
- ServerAndClient

What are the Folders in MVC application solutions?

When you create a project a folder structure gets created by default under the name of your project which can be seen in solution explorer. Below I will give you a brief explanation of what these folders are for.

Model: This folder contains classes that is used to provide data. These classes can contain data that is retrieved from the database or data inserted in the form by the user to update the database.

Controllers: These are the classes which will perform the action invoked by the user. These classes contains methods known as "Actions" which responds to the user action accordingly.

Views: These are simple pages which uses the model class data to populate the HTML controls and renders it to the client browser.

App_Start: Contains Classes such as FilterConfig, RoutesConfig, WebApiConfig. As of now we need to understand the RouteConfig class. This class contains the default format of the URL that should be supplied in the browser to navigate to a specified page.

Explain RenderSection in MVC?

RenderSection() is a method of the WebPageBase class. Scott wrote at one point, The first parameter to the "RenderSection()" helper method specifies the name of the section we want to render at that location in the layout template. The second parameter is optional, and allows us to define whether the section we are rendering is required or not. If a section is "required", then Razor will throw an error at runtime if that section is not implemented within a view template that is based on the layout file (that can make it easier to track down content errors). It returns the HTML content to render.

What is the use of remote validation in MVC?

Remote validation is the process where we validate specific data posting data to a server without posting the entire form data to the server. Let's see an actual scenario, in one of my projects I had a requirement to validate an email address, whether it already exists in the database. Remote validation was useful for that; without posting all the data we can validate only the email address supplied by the user.

Let's create a MVC project and name it accordingly, for me its "TestingRemoteValidation". Once the project is created let's create a model named UserModel that will look like

Explain Bundle.Config in MVC?

"BundleConfig.cs" in MVC4 is used to register the bundles by the bundling and minification system. Many bundles are added by default including jQuery libraries like - jquery.validate, Modernizr, and default CSS references.

What is the meaning of Unobtrusive JavaScript?

This is a general term that conveys a general philosophy, similar to the term REST (Representational State Transfer). Unobtrusive JavaScript doesn't intermix JavaScript code in your page markup.

Eg : Instead of using events like onclick and onsubmit, the unobtrusive JavaScript attaches to elements by their ID or class based on the HTML5 data- attributes.

What is Dependency Injection?

- Dependency Injection (DI) is a design pattern that takes away the responsibility of creating dependencies from a class thus resulting in a loosely coupled system
- The core features of the DI container have been abstracted out to the `IServiceProvider`interface and are available throughout the stack. Because the `IServiceProvider` is the same across all components of the ASP.NET framework a single dependency can be resolved from any part of the application.
- The DI container supports just 4 modes of operation:
    - **Instance** – a specific instance is given all the time. You are responsible for its initial creation.
    - **Transient** – a new instance is created every time.
    - **Singleton** – a single instance is created and it acts like a singleton.
    - **Scoped** – a single instance is created inside the current scope. It is equivalent to Singleton in the current scope.

Advantages of Dependency Injection:

- Reduces class coupling
- Increases code reusing
- Improves code maintainability
- Improves application testing