

# Programming Microsoft ASP.NET MVC

## Chapter 01

### ASP.NET MVC controllers

1. What is Controller in ASP.NET MVC?
  - A controller is responsible for controlling the way that a user interacts with an MVC application.
  - Controller is responsible for
    - Processing request
    - Extracting data for view
    - Returning response to client
2. What is a route?
  - A route is a URL pattern from which controller, action and route data can be mapped
3. Name the available value provider for controller.
  - Child action values
  - Form data
  - Route data
  - Query string
  - Posted files
4. Write down the predefined action result types.
  - `ContentResult`
  - `EmptyResult`
  - `FileContentResult`
  - `FilePathResult`
  - `FileStreamResult`
  - `HttpNotFoundResult`
  - `HttpUnauthorizedResult`
  - `JavaScriptResult`
  - `JsonResult`
  - `PartialViewResult`
  - `RedirectResult`
  - `RedirectToRouteResult`
  - `ViewResult`

## Chapter 02

### ASP.NET MVC views

5. What is view engine?
  - The view engine is the component that physically builds the HTML output for the browser.
  - The view engine engages for each request that returns HTML, and it prepares its output by mixing together a template for the view and any data the controller passes in.
  - In ASP.NET MVC, a view engine is merely a class that implements a fixed interface—the `IViewEngine` interface. Each application can have one or more view engines
  - In ASP.NET MVC 5, each application comes by default with two view engines - ASPX or Razor. Default is Razor view engine.
6. What is action invoker?
  - The activity of both controllers and view engines is coordinated by an external manager object – the action invoker.
  - The action invoker is triggered directly by the HTTP handler in charge of the request. The action invoker does two key things.
    - First, it executes the controller's method and saves the action result.
    - Next, it processes the action result.
7. What is a View?
  - The view object is an instance of a class that implements the `IView` interface.
  - The only purpose of a view object is for writing some HTML response.
  - Each view is identified by name.
  - Each view has a view Template – contains razor code of HTML mixed with model data
  - Each view is associated with some physical file that defines the HTML layout to render
8. What are HTML Helpers?
  - The `HtmlHelper` class renders HTML controls in the razor view.
  - It binds the model object to HTML controls to display the value of model properties into those controls

9. Name 5 Basic Html Helpers and write what HTML each render.
- `ActionLink` – returns HTML string for a anchor (<a>) tag
  - `BeginForm` - renders the <form> tag
  - `CheckBoxFor` - Returns the HTML string for a check box input element
  - `PasswordFor` - Returns the HTML string for a password input element
  - `RadioButtonFor` - Returns the HTML string for a radio button input element
  - `Partial` - Returns the HTML string incorporated in the specified user
  - `ValidationMessageFor` - Returns the HTML string for a validation message
  - `ValidationSummary` - Returns the HTML string for a validation summary message

10. What is a Partial View?

- In ASP.NET MVC, a partial view is similar to a user control in Web Forms.
- The typical location for a partial view is the Shared folder in Views.
- However, you can also store a partial view in the controller-specific folder.
- You use either the `Partial` or `RenderPartial` helper method to insert a partial view.
- For example
  - `@Html.Partial("login")`
  - `@ {Html.RenderPartial("login");}`

11. What are templated helpers?

- Templated helpers take an instance of a C# class, read properties, and decide how to best render those values.
- Templated helpers have the special ability to process metadata (if any) and adjust their rendering accordingly;
- In ASP.NET MVC, there are two essential templated helpers: `Editor` and `Display`
- `Html.DisplayFor` - The `DisplayFor` helper accepts a lambda expression and requires that a view-model object be passed to the view: `@Html.DisplayFor(model => model.FirstName)`
- `Html.EditorFor` - The editor recognizes the type of the value it gets and picks up a made-to-measure template for editing. `@Html.EditorFor(person => person.FirstName)`

12. What are Custom Helpers?

- Custom helpers extend existing helpers to take control over the markup of basic HTML elements
- To create a custom helper you have to write an extension method for the `HtmlHelper` class or for the `AjaxHelper` class

13. Write down the common properties of Commonly used properties and methods of a Razor view object.

Property	Description
<b>Ajax</b>	Gets an instance of the <code>AjaxHelper</code> class used to reference Ajax HTML helpers around the template
<b>IsAjax</b>	Returns true if the current request was initiated by the browser's Ajax object.
<b>Layout</b>	Gets and sets the path to the file containing the master view template.
<b>Model</b>	Gets a reference to the view model object (if any) containing data for the view.
<b>ViewBag</b>	Gets a reference to the <code>ViewBag</code> dictionary that might contain data the controller needs to pass to the view object.
<b>ViewData</b>	Gets a reference to the <code>ViewData</code> dictionary that might contain data the controller needs to pass to the view object.

14. What is a Section in Layout template?

- In the layout template, you define an injection point by placing a call to `RenderSection` at the locations where you want those sections to appear.

15. What is ViewData?

- Using `ViewData` dictionary the controller can pass data to a view.
- The `ViewData` is a name-value dictionary object

16. What is ViewBag?

- Using `ViewBag` the controller can pass data to a view.
- `ViewBag` is dynamic object. You pass data as a property value of the `ViewBag` object

17. What is strongly-typed view?

- The view which binds to a specific type of ViewModel is called as Strongly Typed View.
- By specifying the model, the Visual studio provides the intellisense and compile time checking of type.

18. What is render action?

- A render action is a controller method that is specifically designed to be called from within a view.
- A render action is therefore a regular method on the controller class that you invoke from the view by using one of the following HTML helpers: Action or RenderAction.
  - `@Html.Action("Action Name", "Controller")`
  - `@{Html.RenderAction("Action Name", "Controller");}`

## Chapter 03

### The model-binding architecture

1. What is input model?

- The input model provides the representation of the data being posted to the controller.
- To process user input you must pass input data to controller action receive data
- To pass input data to a controller, you need to package data in some way.
- This is precisely where the input model comes into play.

2. What is view model?

- The view model provides the representation of the data being worked on in the view.

3. What is domain model?

- The domain model is the representation of the domain-specific entities operating in the middle tier.

4. What is Model Binder?

- ASP.NET MVC model binder allows you to map Http Request data with the model. Http Request data means when a user makes a request with form data from the browser to a Controller.
- Model binder works as a middleman to map the incoming HTTP request with Controller action method parameters.
- Model Binder does the following
  - Retrieves data from various sources such as route data, form fields, and query strings.
  - Provides the data to controllers in action method parameters and public properties.
  - Converts string data to .NET types.
  - Updates properties of complex types.

5. Write down the properties of the Bind attribute?

- Prefix - String property. It indicates the prefix that must be found in the name of the posted value for the binder to resolve it. The default value is the empty string.
- Include - Gets or sets a comma-delimited list of property names for which binding is permitted
- Exclude - Gets or sets a comma-delimited list of property names for which binding is not allowed.

## Chapter 04

### Input forms

1. What is TempData?

- TempData dictionary stores any data you provide in the session state for as long as two requests. the container clears the entry.

2. Write down the data Annotation that affects the rendering of data?

- DataType - Indicates the data type of the member
- DisplayFormat - Use this to indicate a format for the value
- DisplayName - Indicates the text to use for the label that presents the value.
- HiddenInput - Indicates whether a hidden input field should be displayed instead of a visible one
- UIHint - Indicates the name of the custom HTML template to use

3. What are data annotations?

- Data annotations are a set of attributes used on public properties of any .NET class so that that any interested client code can read and consume.
- Attributes fall into two main categories: display and validation.

4. Write down the Data annotation attributes for validation?

- Compare - Checks whether two specified properties in the model have the same value
- CustomValidation - Checks the value against the specified custom function.

- EnumDataType - Checks whether the value can be matched to any of the values in the specified enumerated type.
- Range - Checks whether the value falls in the specified range.
- RegularExpression - Checks whether the value matches the specified expression
- Remote - Makes an Ajax call to the server and checks whether the value is acceptable.
- Required - Checks whether a non-null value is assigned to the property.
- StringLength - Checks whether the string is longer than the specified value

## Chapter 05

### Aspects of ASP.NET MVC applications

1. How is the session state maintained in MVC?
  - HttpSessionState class provides a dictionary-based model of storing and retrieving session-state values
  - Session state can only be accessed in controllers.
  - The session dictionary is a name/value collection, so it requires plain strings to identify entries.
2. What is caching?
  - Caching indicates the application's ability to save frequently used data to an intermediate storage medium.
  - In ASP.NET, built-in caching capabilities come through the Cache object.

## Chapter 06

### Securing your application

1. Why is the authorize attribute?
  - You use the Authorize attribute when you want to restrict access to an action method and ensure that only authenticated users can execute
2. What is ASP.NET identity?
  - An identity is credentials associated with the current user.
  - User credentials are retrieved from database
3. What is an identity user?
  - An IdentityUser inherits from TUser represents current user to be managed.
  - It contains the data of current user
4. What are User Manager and Store Manager?
  - The user manager is an instance of User Manager<TUser>
  - It basically takes the roles of signing users in and out.
  - The store manager is an instance of UserStore<TUser>
  - It is responsible for creating, updating and deleting user

## Chapter 08

### Customizing ASP.NET MVC controllers

1. What is an action filter?
  - An action filter is a piece of code that runs around the execution of an action method and can be used to modify and extend the behavior of the method.
  - Action filters implemented in attributes to be applied to action methods.
  - There are a number Built-in action filters : Authorize, ValidateAntiforgeryToken for example
2. Write down the types of action filters and their purposes.
  - IActionFilter - Defines two methods, one that executes before and one that executes after the controller action
  - IAuthenticationFilter - Defines a method that executes early in the action pipeline, giving you a chance to customize the process of authentication
  - IAuthorizationFilter - Defines a method that executes after authentication
  - IExceptionHandler - Defines a method that runs whenever an exception is thrown
  - IResultFilter - Defines two methods, one that executes before and one that executes after the processing of the action result
3. Write the Predefined filters in ASP.NET MVC?
  - AsyncTimeout - Marks an action method as one that will execute asynchronously and terminate in the specified number of milliseconds
  - Authorize - Marks an action method as one that can be accessed only by authenticated users
  - ChildActionOnly - Marks an action method as one that can be executed only as a child action
  - HandleError - Marks an action method as one that requires automatic handling of any exceptions

- OutputCache - Marks an action method as one whose output needs to be cached
  - RequireHttps - Marks an action method as one that requires a secure request
  - ValidateAntiForgeryToken - Marks an action method as one that requires validation against the antiforgery token
  - ValidateInput - Marks an action method as one whose posted input data might (or might not) need validation.
4. Why is the ActionName filter used?
- The purpose of the selector is simple: checking whether the specified action name is a valid action name for the method.

## Chapter 10

### An executive guide to Web API

1. What is Web API?
  - ASP.NET Web API is a framework expressly designed to support the building of REST HTTP services that can be consumed by a variety of clients; in particular, HTML pages and mobile
2. HTTP verbs and actions in Web API.
  - GET - used to return .net objects serialized in Jason or xml format
  - POST - used to add new resources to some back-end stores.
  - PUT - used to expected to update an existing resource in some back-end store
  - DELETE - Used to delete an existing resource in some back-end store.
3. What is attribute routing?
  - Attribute routing allows to define route without registering a route template.