Controllers and Action Methods

* The Asp .NET MVC framework maps URLs to classes that are referred to as controllers.
* Controllers process incoming requests, handle user input and interactions, and execute appropriate application logic.
* A controller class typically calls a separate view component to generate the HTML markup for the request.
* The Controller class is responsible for the following processing stages:
  + Locating the appropriate action method to call and validating that it can be called.
  + Getting the values to use as the action method’s arguments.
  + Handling all errors that might occur during the execution of the action method.
  + Providing the default class for rendering views.
* All controller classes must be named by using the “Controller” suffix.

Action Methods

* The user interaction with MVC applications is organized around controllers and action methods.
* The controller defines action methods. Controllers can include as many action methods as needed.
* Action methods typically have a one-to-one mapping with user interactions.
* Each of these user interactions causes a request to be sent to the server.
* The URL of the request includes information that the MVC framework uses to invoke an action method.

|  |
| --- |
| Notes:\*   * IIS is for Microsoft suitable only for static pages * KESTREL server of the container behind the web server for dynamic web pages with razor and c# * /Products/Details/5 (default stored in id variable) |

ActionResult Return Type

* Most action methods return an instance of a class that derives from ActionResult.
* The ActionResult class is the base for all action results.
* However, there are different action result types, depending on the task that the action method is performing.
* For Example, the most common action is to call the View() method.
* The View method returns an instance of the ViewResult class, which is derived from ActionResult.
* Action methods can return an object of any type, such as a string, an integer, or a Boolean value.
* These return types are wrapped in an appropriate ActionResult type before they are rendered to the response stream.
* By default, the MVC framework treats all public methods of a controller class as action methods.
* Preventing a Public Method from Being Invoked
  + [NonAction]

Private void DoSomething(){ // Method Logic. }

* + If you do not want a public method to be an action method, then mark that method with the NonActionAttribute.

Optional Parameters

* The MVC framework also supports optional arguments for action methods. Optional Parameters in the MVC framework are handled by using nullable-type arguments for controller action methods.
* If a method can take a date as part of the query string but you want to the default to be todays date if the query string parameter is missing.
* Public ActionResult ShowArticles (DateTime? Date)

{if(!date.HasValue) {date = DateTime.Now;} // … }

Adding an Action to a Controller

* In order to be exposed as an action, a method must :
  + Be public
  + Cannot be a static method
  + Cannot be an extension method
  + Cannot be a constructor, getter, or setter
  + Cannot have open generic types
  + The method is not a method of the base class.
  + The method cannot contain ref or out parameters.
  + Method on action overload is not supported in MVC
  + Cannot pass data to two controller using ViewData need to use TempData

\*Extension Methods

Return Type

* The ASP .NET MVC framework will convert any return type that is not an action result into a string and render the string to a browser.

Using the Default Route Table

* The ASP .NET Routing module is responsible for mapping incoming browser requests to particular MVC controller actions.
* When you create a new ASP .NET MVC ……see pic

RENDERING A FROM USING TAGHELPER

* They are inbuilt razor elements that get converted into html elements during runtime.
* They are not actual HTML elements but they get converted into html elements.

Available TagHelper

* ActionLink – Links to an action method.
* BeginForm – Marks the start of a form and links to the action method that renders the form.
* CheckBox – Renders a check box.
* DropDownList – Renders a drop-down list.
* Hidden – Embeds information in the form that is not rendered for the user to see.
* ListBox – Renders a list box.
* Password – Renders a text box for entering a password.
* RadioButton – Renders a text box for entering a password.
* TextArea – Renders a text area (multi-line text box).
* TextBox – Renders a text box.

|  |
| --- |
| State Management In Web Applications\* ISSUE to know of  Firewall  Web Server |

Differences between TCP and HTTP protocols

First choice always cookies.

URL encoding or as hidden fields if clients server is a cookie less environment is called as URL rewriting.

Session Tracking

Only for every successful logged in user session tracking is done

For guest it is not done

A sessiuon as a dictionary will be there in the server side

The life of session id is normally 20 mins

For banking payment it is 5 mins

Viewdata and tempdata are single users

Static values are multi user but not individual but shared.

WCF Core (no longer in use) for TCP applications alternate is DAPR not DAPPR

Partial Views

Invoking action from a view