1. **What is model?**

**Ans:** A model is the representation of the data being posted to the Controller, the data being worked on in a view or the representation of the domain specific entities operating in the business tier. The model contains core application information. It includes data, validation rules, data access and aggregation logic

1. **What is routing in MVC?**

**Ans:** Routing is the process of directing an HTTP request to a controller and the functionality of this processing is implemented in System.Web.Routing. This assembly is not part of ASP.NET MVC. It is actually part of ASP.NET runtime.

1. **What is the difference between Temp data, View, and View Bag?**

**Ans:** ViewBag, ViewData and TempData all are objects in ASP.NET MVC and these are used to pass the data from controller to view.

* **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. And we have no need to typecast to read the data or for null checking. The scope of ViewBag is permitted to the current request and the value of ViewBag will become null while redirecting.
* **ViewData:** ViewData is a dictionary object to pass the data from Controller to View where data is passed in the form of key-value pair. And typecasting is required to read the data in View if the data is complex and we need to ensure null check to avoid null exception. The scope of ViewData is similar to ViewBag and it is restricted to the current request and the value of ViewData will become null while redirecting.
* **TempData:** TempData is a dictionary object to pass the data from one action to another action in the same Controller or different Controllers. Usually, TempData object will be stored in a session object. TempData is also requited to typecast and for null checking before reading data from it. TempData scope is limited to the next request and if we want TempData to be available further, we should use Keep and peek.

1. **What is difference between MVC and Web Forms?**

**Ans:**

|  |  |
| --- | --- |
| **ASP.NET MVC** | **Web Forms** |
| MVC focuses on separation of concern, i.e., there is no fixed code behind page for every view. A view can be called from multiple action. | Web form based on functions and page behind code, i.e., there is code behind page for each view. You have to write code in that class related to this view only. |
| First request comes to controller and action, then view gets called. | First request comes to Page (View) then it will go to code behind page. |
| MVC provides HTML Helpers to create form controls. This is optional. You can use simple HTML controls also. | For everything in webforms, you have a server control. |
| There is no ViewState for state management in View. | Viewstate is used to maintain state of form in view. This also makes page heavy. |
| We create partial views for reusable views. | We create user controls for reusable view or control. |

1. **What is session? What is the default time for session?**

**Ans:** Session is a State Management Technique. A Session can store the value on the Server. It can support any type of object to be stored along with our own custom objects. A session is one of the best techniques for State Management because it stores the data as client-based, in other words, the data is stored for every user separately and the data is secured also because it is on the server. Default timeout for each Session is 20 mins.

1. **What is Partial View in MVC? With example**

**Ans:** Partial view in ASP.NET MVC is special view which renders a portion of view content. It is just like a user control of a web form application. Partial can be reusable in multiple views. It helps us to reduce code duplication. In other word a partial view enables us to render a view within the parent view.

1. **What is the difference between View and Partial View?**

**Ans:**

|  |  |
| --- | --- |
| **View** | **Partial View** |
| View contains the layout page | Partial view does not contain the layout page |
| \_viewstart page is rendered before any view is rendered | Partial view does not check for a \_viewstart.cshtml. We cannot place any common code for a partial view within the \_viewStart.cshtml page. |
| View may have markup tags like html, body, head, title, meta etc. | The Partial view is specially designed to render within the view and as a result it does not contain any mark up. |

1. **Explain the concept of MVC Scaffolding?**

**Ans:** Scaffolding is used to define the code-generation framework used in web applications. It uses T4 templates to generate basic controllers and views for the models. It generates instances for the mapped domain model and code for all CRUD operations. It also reduces the amount of time for developing a standard data operation in the application.

Basically, it is an automated code generation framework, it generates code for CRUD operations based on the provided domain model classes and DB connections. You can add scaffolding to your project when you want to add code that interacts with the data model in the shortest amount of time.

1. **How to change time of session?**

**Ans:** To change the time of session we can use the inbuilt function Session.Timeout and assign the time in integer.

1. **What is query string? What are disadvantages of query string?**

**Ans:** A query string is a collection of characters input to a computer or web browser. A Query String is helpful when we want to transfer a value from one page to another.

Disadvantages: All the attributes and values are visible to the end user. Therefore, they are not secure. There is a limit to URL length of 255 characters.

1. **What is cookie? What are limitations for cookie?**

**Ans:** ASP.NET Cookie is a small bit of text that is used to store user-specific information. This information can be read by the web application whenever user visits the site.

When a user requests for a web page, web server sends not just a page, but also a cookie containing the date and time. This cookie stores in a folder on the user's hard disk.

When the user requests for the web page again, browser looks on the hard drive for the cookie associated with the web page. Browser stores separate cookie for each different sites user visited.

**Limitations:** Most browsers support cookies of up to 4096 bytes. Because of this small limit, cookies are best used to store small amounts of data, or better yet, an identifier such as a user ID. The user ID can then be used to identify the user and read user information from a database or other data store

1. **Create one example to store data in session and show on other view.**

**Ans:**

1. **Explain MVC application life cycle.**

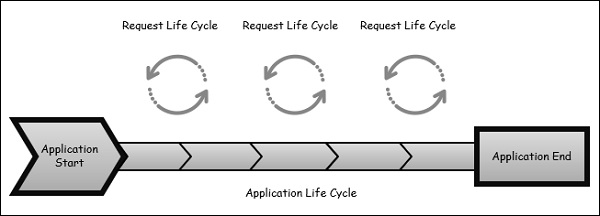
**Ans:** A life cycle is simply a series of steps or events used to handle some type of request or to change an application state.

MVC has two life cycles:

* The application life cycle
* The request life cycle

**The Application life cycle:**

The application life cycle refers to the time at which the application process actually begins running IIS until the time it stops. This is marked by the application start and end events in the startup file of your application.



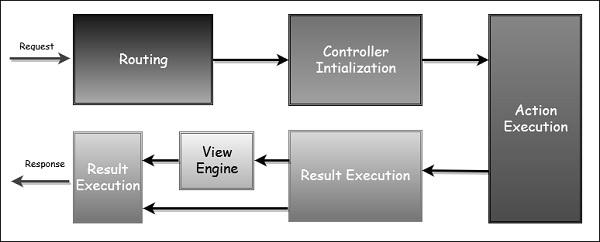
**The Request Life Cycle:**

It is the sequence of events that happen every time an HTTP request is handled by our application.

The entry point for every MVC application begins with routing. After the ASP.NET platform has received a request, it figures out how it should be handled through the URL Routing Module.

Modules are .NET components that can hook into the application life cycle and add functionality. The routing module is responsible for matching the incoming URL to routes that we define in our application.

All routes have an associated route handler with them and this is the entry point to the MVC framework.



The MVC framework handles converting the route data into a concrete controller that can handle requests. After the controller has been created, the next major step is **Action Execution**. A component called the **action invoker** finds and selects an appropriate Action method to invoke the controller.

After our action result has been prepared, the next stage triggers, which is **Result Execution**. MVC separates declaring the result from executing the result. If the result is a view type, the View Engine will be called and it's responsible for finding and rending our view.

If the result is not a view, the action result will execute on its own. This Result Execution is what generates an actual response to the original HTTP request.

1. **List out different return types of a controller action method**

**Ans:**

* Return View
* Return Partial View
* Redirect
* Redirect to Action
* Return Content
* Return Jason
* Return JavaScript
* Return File

1. **What are filters in MVC?**

**Ans:** The ASP.NET MVC framework supports four different types of filters: Authorization filters – Implements the AuthorizationFilter attribute. Action filters – Implements the ActionFilter attribute. Result filters – Implements the ResultFilter attribute. Exception filters – Implements the ExceptionFilter attribute.

1. **What are HTML helpers in MVC?**

**Ans:**

* HTML Helpers are methods that return a string.
* Helper class can create HTML controls programmatically. HTML Helpers are used in View to render HTML content.
* It is not mandatory to use HTML Helper classes for building an ASP.NET MVC application.
* We can build an ASP.NET MVC application without using them, but HTML Helpers helps in the rapid development of a view.
* HTML Helpers are more lightweight as compared to ASP.NET Web Form controls as they do not use ViewState and do not have event models.
* MVC has built-in Helpers methods
* We can create custom HTML helpers.

1. **Differences between Razor and ASPX View Engine in MVC?**

**Ans:**

|  |  |
| --- | --- |
| **Razor View Engine** | **ASPX View Engine** |
| Razor View Engine is an advanced view engine and introduced with MVC3. This is not a language but it is a markup syntax. | ASPX View Engine is the default view engine for the ASP.NET MVC that is included with ASP.NET MVC from the beginning. |
| Razor View Engine supports System.Web.Razor. | ASPX View Engine supports System.Web.Mvc.WebFormViewEngine |
| In Razor View Engine we use Layouts. | In ASPX View Engine we use masterPages. |
| In Razor View Engine we use PartialPage. | In ASPX View Engine we use WebUserControls. |
| Razor View Engine has .cshtml (with C#) and .vbhtml (with VB) extension for views, Layout and Partial views. | ASPX View Engine has a similar extension as in a simple web application like .aspx for the views, .acsx for UserControls and .master for Master Pages. |
| Razor Engine is a little slow compared to Aspx Engine. | Aspx Engine is faster compared to Razor Engine. |
| ‘@’ symbol uses in Razor Engine to write the code. @Html.ActionLink("Login", "LoginView") | ‘<%:’ delimiters use as starting point and ‘ %>’ use as ending point. You can write the code between them in ASPX Engine. |
| Razor Engine prevents Cross-Site Scripting Attacks, in other words it encodes the script or HTML tags like <,> before rendering to view. | ASPX Engine does not prevent Cross-Site Scripting Attacks, in other words any script saved in the database will be fired while rendering the page. |

1. **Pass one values from one view to another view using query string.**

**Ans:**