

WWW.QUESTPOND.COM

MVC (Model View Controller) Interview questions with answers

www.questpond.com

Shivprasad Koirala

11/19/2013

Visit us for more details at www.questpond.com

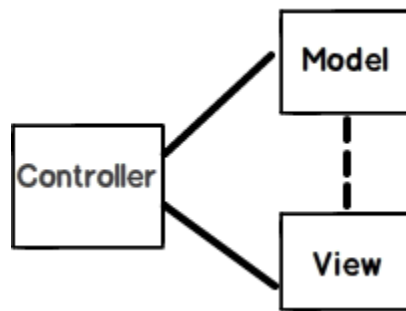
MVC Interview questions and answers

What is MVC?.....	2
Can you explain the complete flow of MVC?.....	2
Is MVC suitable for both windows and web application?	2
What are the benefits of using MVC?	2
Is MVC different from a 3 layered architecture?	3
What is the latest version of MVC?	4
What is the difference between each version of MVC?	4
What are routing in MVC?.....	4
Where is the route mapping code written?.....	4
Can we map multiple URL's to the same action?	5
How can we navigate from one view to other view using hyperlink?	5
How can we restrict MVC actions to be invoked only by GET or POST?	5
How can we maintain session in MVC?	5
What is the difference between TempData , ViewData and ViewBag?	5
What are partial views in MVC?	7
How did you create partial view and consume the same?.....	7
How can we do validations in MVC?	8
Can we display all errors in one go?	10
What are the other data annotation attributes for validation in MVC?	10
How can we enable data annotation validation on client side?	11
What is Razor in MVC?	11
Why Razor when we already had ASPX?	11
So which is a better fit Razor or ASPX?	12
How can you do authentication and authorization in MVC?	12
How to implement Windows authentication for MVC?	12
How do you implement forms authentication in MVC?	12
How to implement Ajax in MVC?	13
What kind of events can be tracked in AJAX ?	16
What is the difference between "ActionResult" and "ViewResult"?	16
What are the different types of results in MVC?.....	16
What are "ActionFilters" in MVC?.....	17
Can we create our custom view engine using MVC?	19
How to send result back in JSON format in MVC?	22
What is "WebAPI"?	23
But WCF SOAP also does the same thing, so how does "WebAPI" differ?	23
With WCF also you can implement REST, So why "WebAPI"?	23
How to implement "WebAPI" in MVC?	23
How can we detect that a MVC controller is called by POST or GET ?	25
What is Bundling and Minification in MVC?	26
How does bundling increase performance?.....	26
So how do we implement bundling in MVC ?.....	27
How can you test bundling in debug mode ?	28
Explain minification and how to implement the same ?	28
How to implement minification ?.....	29
What is the use of "AllowHTML" and "ValidateInput" attributes?	29

What is MVC?

MVC is an architectural pattern which separates the representation and the user interaction. It's divided in to three broader sections, "Model", "View" and "Controller". Below is how each one of them handles the task.

- The "View" is responsible for look and feel.
- "Model" represents the real world object and provides data to the "View".
- The "Controller" is responsible to take the end user request and load the appropriate "Model" and "View".



MVC (Model view controller)

Can you explain the complete flow of MVC?

Below are the steps how control flows in MVC (Model, view and controller) architecture:-

- All end user requests are first sent to the controller.
- The controller depending on the request decides which model to load. The controller loads the model and attaches the model with the appropriate view.
- The final view is then attached with the model data and sent as a response to the end user on the browser.

Is MVC suitable for both windows and web application?

MVC architecture is suited for web application than windows. For window application MVP i.e. "Model view presenter" is more applicable. If you are using WPF and SLMVVM is more suitable due to bindings.

What are the benefits of using MVC?

There are two big benefits of MVC:-

- Separation of concerns is achieved as we are moving the code behind to a separate class file. By moving the binding code to a separate class file we can reuse the code to a great extent.
- Automated UI testing is possible because now the behind code (UI interaction code) has moved to a simple.NET class. This gives us opportunity to write unit tests and automate manual testing.

Is MVC different from a 3 layered architecture?

MVC is an evolution of a 3 layered traditional architecture. Many components of 3 layered architecture are part of MVC. So below is how the mapping goes.

Functionality	3 layered / tiered architecture	Model view controller architecture
Look and Feel	User interface.	View.
UI logic	User interface.	Controller
Business logic /validations	Middle layer	Model.
Request is first sent to	User interface	Controller.
Accessing data	Data access layer.	Data access layer.

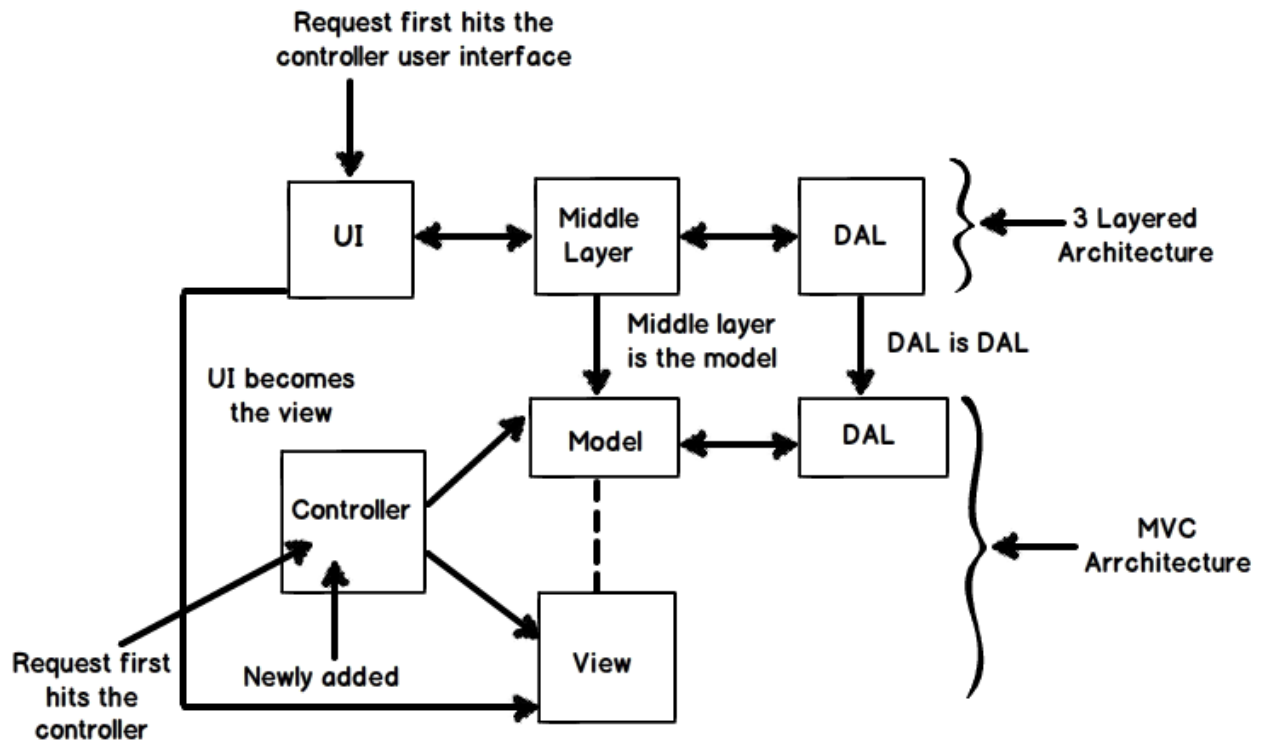


Figure 6.2:- 3 layered architecture

What is the latest version of MVC?

When this note was written, four versions were released of MVC. MVC 1, MVC 2, MVC 3 and MVC 4. So the latest is MVC 4.

What is the difference between each version of MVC?

Below is a detail table of differences. But during interview it's difficult to talk about all of them due to time limitation. So I have highlighted important differences which you can run through before the interviewer.

MVC 2	MVC 3	MVC 4
Client-Side Validation Templated Helpers Areas Asynchronous Controllers Html.ValidationSummary Helper Method DefaultValueAttribute in Action-Method Parameters Binding Binary Data with Model Binders DataAnnotations Attributes Model-Validator Providers New RequireHttpsAttribute Action Filter Templated Helpers Display Model-Level Errors	Razor Readymade project templates HTML 5 enabled templates Support for Multiple View Engines JavaScript and Ajax Model Validation Improvements	ASP.NET Web API Refreshed and modernized default project templates New mobile project template Many new features to support mobile apps Enhanced support for asynchronous methods

What are routing in MVC?

Routing helps you to define a URL structure and map the URL with the controller.

For instance let's say we want that when any user types "http://localhost/View/ViewCustomer/" , it goes to the "Customer" Controller and invokes "DisplayCustomer" action. This is defined by adding an entry in to the "routes" collection using the "maproute" function. Below is the under lined code which shows how the URL structure and mapping with controller and action is defined.

```
routes.MapRoute(  
    "View", // Route name  
    "View/ViewCustomer/{id}", // URL with parameters  
    new { controller = "Customer", action = "DisplayCustomer", id  
        = UrlParameter.Optional }); // Parameter defaults
```

Where is the route mapping code written?

The route mapping code is written in the "global.asax" file.

Can we map multiple URL's to the same action?

Yes , you can , you just need to make two entries with different key names and specify the same controller and action.

How can we navigate from one view to other view using hyperlink?

By using “ActionLink” method as shown in the below code. The below code will create a simple URL which help to navigate to the “Home” controller and invoke the “GotoHome” action.

```
<%= Html.ActionLink("Home", "Gotohome") %>
```

How can we restrict MVC actions to be invoked only by GET or POST?

We can decorate the MVC action by “HttpGet” or “HttpPost” attribute to restrict the type of HTTP calls. For instance you can see in the below code snippet the “DisplayCustomer” action can only be invoked by “HttpGet”. If we try to make Http post on “DisplayCustomer” it will throw an error.

```
[HttpGet]
public ActionResult DisplayCustomer(int id)
{
    Customer objCustomer = Customers[id];

    return View("DisplayCustomer", objCustomer);
}
```

How can we maintain session in MVC?

Sessions can be maintained in MVC by 3 ways TempData , ViewData and ViewBag.

What is the difference between TempData , ViewData and ViewBag?

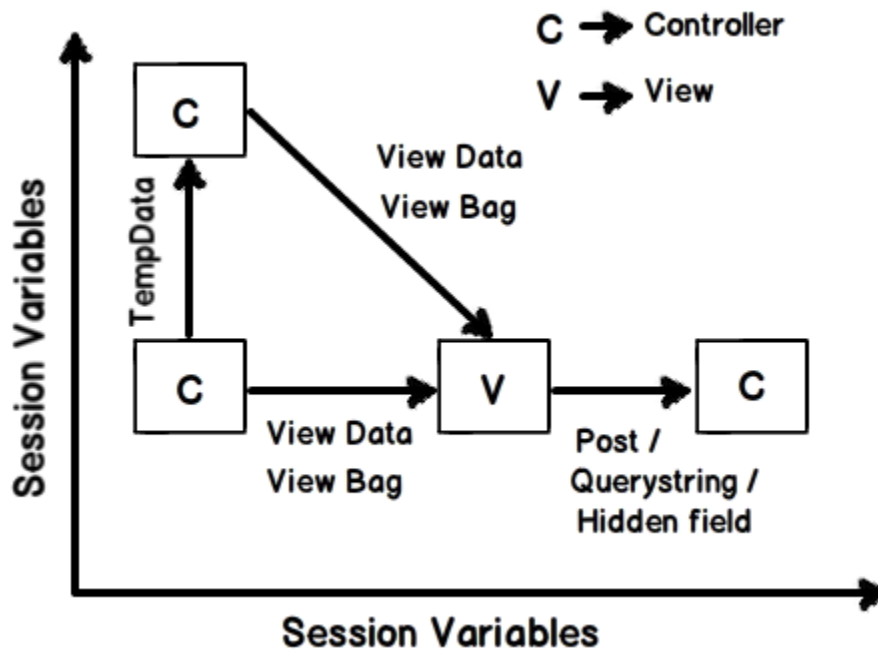


Figure 6.3:- difference between tempdata ,viewdata and viewbag

Temp data: -Helps to maintain data when you move from one controller to other controller or from one action to other action. In other words when you redirect,“tempdata” helps to maintain data between those redirects. It internally uses session variables.

View data: - Helps to maintain data when you move from controller to view.

View Bag: - It’s a dynamic wrapper around view data. When you use “Viewbag” type casting is not required. It uses the dynamic keyword internally.

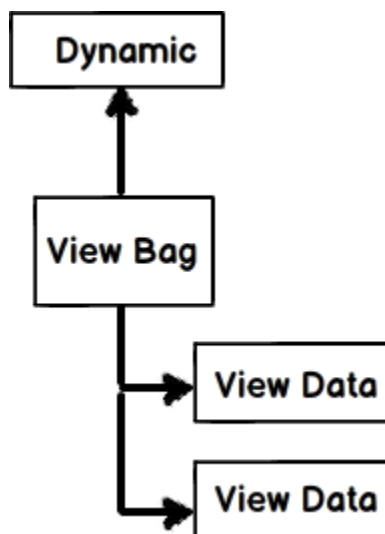


Figure 6.4:-dynamic keyword

Session variables: - By using session variables we can maintain data from any entity to any entity.

Hidden fields and HTML controls: - Helps to maintain data from UI to controller only. So you can send data from HTML controls or hidden fields to the controller using POST or GET HTTP methods.

Below is a summary table which shows different mechanism of persistence.

Maintains data between	ViewData/ViewBag	TempData	Hidden fields	Session
Controller to Controller	No	Yes	No	Yes
Controller to View	Yes	No	No	Yes
View to Controller	No	No	Yes	Yes

What are partial views in MVC?

Partial view is a reusable view (like a user control) which can be embedded inside other view. For example let's say all your pages of your site have a standard structure with left menu, header and footer as shown in the image below.

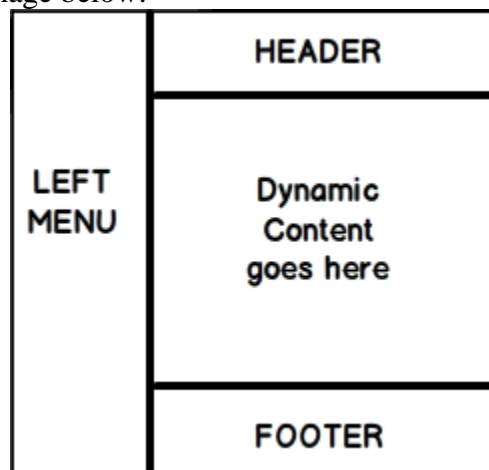


Figure 6.5:- partial views in MVC

For every page you would like to reuse the left menu, header and footer controls. So you can go and create partial views for each of these items and then you call that partial view in the main view.

How did you create partial view and consume the same?

When you add a view to your project you need to check the "Create partial view" check box.

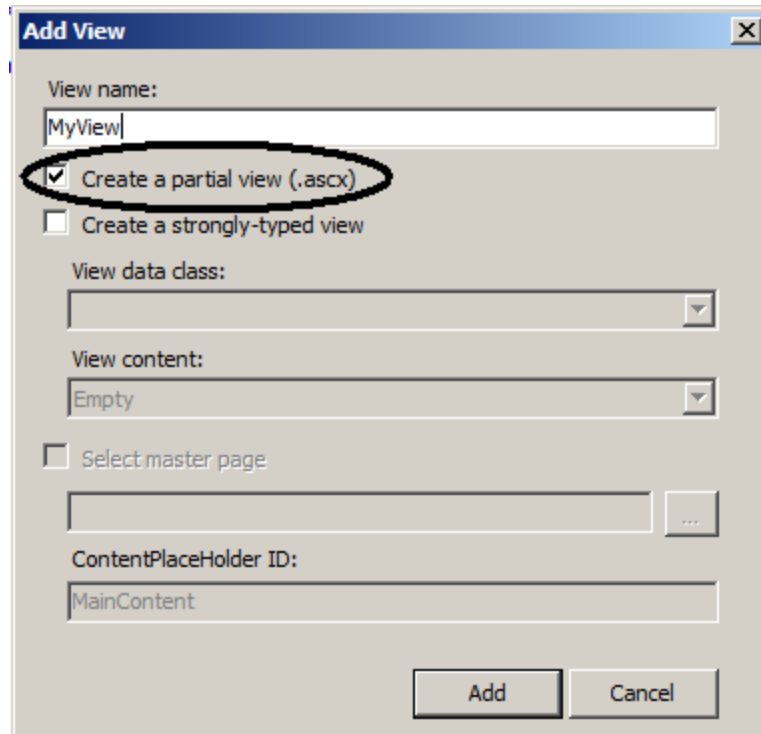


Figure6.6:-createpartialview

Once the partial view is created you can then call the partial view in the main view using “Html.RenderPartial” method as shown in the below code snippet.

```
<body>
<div>
<% Html.RenderPartial("MyView"); %>
</div>
</body>
```

How can we do validations in MVC?

One of the easy ways of doing validation in MVC is by using data annotations. Data annotations are nothing but attributes which you can be applied on the model properties. For example in the below code snippet we have a simple “customer” class with a property “customercode”.

This “CustomerCode” property is tagged with a “Required” data annotation attribute. In other words if this model is not provided customer code it will not accept the same.

```
public class Customer
{
    [Required(ErrorMessage="Customer code is required")]
    public string CustomerCode
```

```

        {
            set;
            get;
        }
    }
}

```

In order to display the validation error message we need to use “ValidateMessageFor” method which belongs to the “Html” helper class.

```

<% using (Html.BeginForm("PostCustomer", "Home", FormMethod.Post))
{ %>
<%=Html.TextBoxFor(m => m.CustomerCode)%>
<%=Html.ValidationMessageFor(m => m.CustomerCode)%>
<input type="submit" value="Submit customer data" />
<%%>

```

Later in the controller we can check if the model is proper or not by using “ModelState.IsValid” property and accordingly we can take actions.

```

public ActionResult PostCustomer(Customer obj)
{
    if (ModelState.IsValid)
    {
        obj.Save();
        return View("Thanks");
    }
    else
    {
        return View("Customer");
    }
}

```

Below is a simple view of how the error message is displayed on the view.

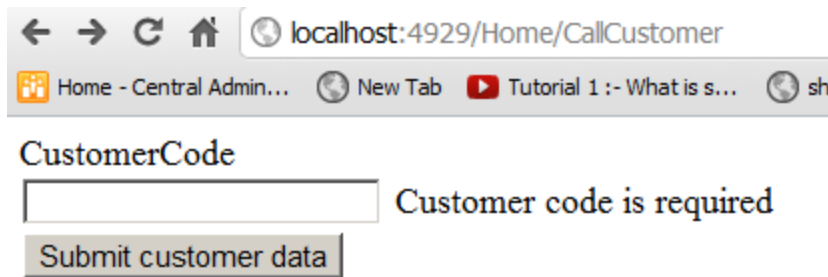


Figure 6.7:- validations in MVC

Can we display all errors in one go?

Yes we can, use “ValidationSummary” method from HTML helper class.

```
<%= Html.ValidationSummary() %>
```

What are the other data annotation attributes for validation in MVC?

If you want to check string length, you can use “StringLength”.

```
[StringLength(160)]
public string FirstName { get; set; }
```

In case you want to use regular expression, you can use “RegularExpression” attribute.

```
[RegularExpression(@"[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}")]
public string Email { get; set; }
```

If you want to check whether the numbers are in range, you can use the “Range” attribute.

```
[Range(10,25)]
public int Age { get; set; }
```

Some time you would like to compare value of one field with other field, we can use the “Compare” attribute.

```
public string Password { get; set; }
[Compare("Password")]
public string ConfirmPass { get; set; }
```

In case you want to get a particular error message , you can use the “Errors” collection.

```
var ErrMessage = ModelState["Email"].Errors[0].ErrorMessage;
```

If you have created the model object yourself you can explicitly call “TryUpdateModel” in your controller to check if the object is valid or not.

```
TryUpdateModel(NewCustomer);
```

In case you want add errors in the controller you can use “AddModelError” function.

```
ModelState.AddModelError("FirstName", "This is my server-side error.");
```

How can we enable data annotation validation on client side?

It’s a two-step process first reference the necessary jquery files.

```
<script src="<%= Url.Content("~/Scripts/jquery-1.5.1.js") %>"
type="text/javascript"></script>
<script src="<%= Url.Content("~/Scripts/jquery.validate.js") %>"
type="text/javascript"></script>
<script src="<%= Url.Content("~/Scripts/jquery.validate.unobtrusive.js") %>"
type="text/javascript"></script>
```

Second step is to call “EnableClientValidation” method.

```
<% Html.EnableClientValidation(); %>
```

What is razor inMVC?

It’s a light weight view engine. Till MVC we had only one view type i.e.ASPX, Razor was introduced in MVC 3.

Why razor when we already had ASPX?

Razor is clean, lightweight and syntaxes are easy as compared to ASPX. For example in ASPX to display simple time we need to write.

```
<%=DateTime.Now%>
```

In Razor it’s just one line of code.

```
@DateTime.Now
```

So which is a better fit Razor or ASPX?

As per Microsoft razor is more preferred because it's light weight and has simple syntaxes. But my suggestion would be to answer more realtime....

How can you do authentication and authorization in MVC?

You can use windows or forms authentication for MVC.

How to implement windows authentication for MVC?

For windows authentication you need to go and modify the “web.config” file and set authentication mode to windows.

```
<authentication mode="Windows"/>
<authorization>
    <deny users="?" />
</authorization>
```

Then in the controller or on the action you can use the “Authorize” attribute which specifies which users have access to these controllers and actions. Below is the code snippet for the same. Now only the users specified in the controller and action can access the same.

```
[Authorize(Users= @"WIN-3LI600MWLQN\Administrator")]
public class StartController : Controller
{
    //
    // GET: /Start/
    [Authorize(Users = @"WIN-3LI600MWLQN\Administrator")]
    public ActionResult Index()
    {
        return View("MyView");
    }
}
```

How do you implement forms authentication in MVC?

Forms authentication is implemented the same way as we do in ASP.NET. So the first step is to set authentication mode equal to forms. The “loginUrl” points to a controller here rather than page.

```
<authentication mode="Forms">
<forms loginUrl="~/Home/Login" timeout="2880"/>
</authentication>
```

We also need to create a controller where we will check the user is proper or not. If the user is proper we will set the cookie value.

```
public ActionResult Login()
{
    if ((Request.Form["txtUserName"] == "Shiv") && (Request.Form["txtPassword"]
    == "Shiv@123"))
    {
        FormsAuthentication.SetAuthCookie("Shiv",true);
        return View("About");
    }
    else
    {
        return View("Index");
    }
}
```

All the other actions need to be attributed with “Authorize” attribute so that any unauthorized user if he makes a call to these controllers it will redirect to the controller (in this case the controller is “Login”) which will do authentication.

```
[Authorize]
public ActionResult Default()
{
    return View();
}

[Authorize]
public ActionResult About()
{
    return View();
}
```

How to implement Ajax in MVC?

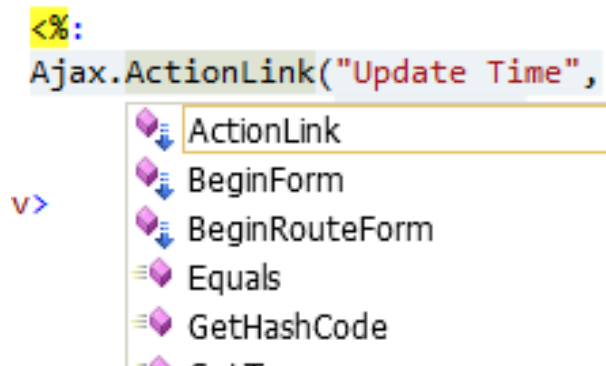
You can implement Ajax in two ways in MVC:-

- Ajax libraries
- JQuery

Below is a simple sample of how to implement Ajax by using “Ajax” helper library. In the below code you can see we have a simple form which is created by using “Ajax.BeginForm” syntax. This form calls a controller action called as “getCustomer”. So now the submit action click will be an asynchronous ajax call.

```
<script language="javascript">
function OnSuccess(data1)
{
// Do something here
}
</script>
<div>
<%
    var AjaxOpt = new AjaxOptions{OnSuccess="OnSuccess"};
    %>
<% using (Ajax.BeginForm("getCustomer","MyAjax",AjaxOpt)) { %>
<input id="txtCustomerCode" type="text" /><br />
<input id="txtCustomerName" type="text" /><br />
<input id="Submit2" type="submit" value="submit"/></div>
<%} %>
```

In case you want to make ajax calls on hyperlink clicks you can use “Ajax.ActionLink” function as shown in the below code.



The screenshot shows a code editor with the following text:

<%:

Ajax.ActionLink("Update Time",

 Below this text, an IntelliSense dropdown menu is open, displaying several methods:

- ActionLink (highlighted with a yellow border)
- BeginForm
- BeginRouteForm
- Equals
- GetHashCode

 The dropdown items are preceded by small purple icons.

Figure 6.8:- implement Ajax in MVC

So if you want to create Ajax asynchronous hyperlink by name “GetDate” which calls the “GetDate” function on the controller, below is the code for the same. Once the controller responds this data is displayed in the HTML DIV tag by name “DateDiv”.

```

<span id="DateDiv" />
<%:
Ajax.ActionLink("Get Date","GetDate",
new AjaxOptions {UpdateTargetId = "DateDiv" })
%>

```

Below is the controller code. You can see how “GetDate” function has a pause of 10 seconds.

```

public class Default1Controller : Controller
{

    public string GetDate()
    {
        Thread.Sleep(10000);
        return DateTime.Now.ToString();
    }
}

```

The second way of making Ajax call in MVC is by using JQuery. In the below code you can see we are making an ajax POST call to a URL “/MyAjax/getCustomer”. This is done by using “\$.post”. All this logic is put in to a function called as “GetData” and you can make a call to the “GetData” function on a button or a hyper link click event as you want.

```

function GetData()
{
    var url = "/MyAjax/getCustomer";
    $.post(url, function (data)
    {

        $("#txtCustomerCode").val(data.CustomerCode);
        $("#txtCustomerName").val(data.CustomerName);
    }
    )
}

```


What kind of events can be tracked in AJAX ?

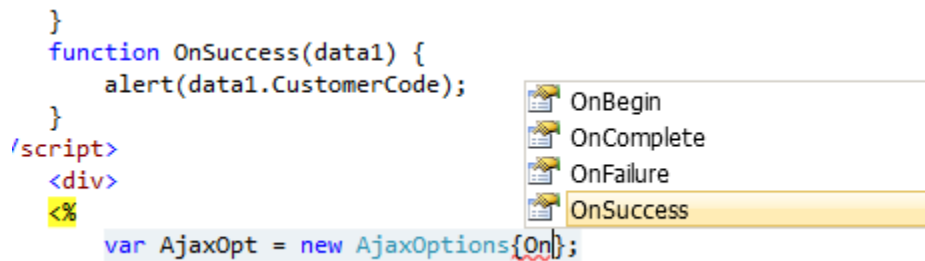


Figure 6.9:- tracked in AJAX

What is the difference between “ActionResult” and “ViewResult”?

“ActionResult” is an abstract class while “ViewResult” derives from “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 view dynamically “ActionResult” is the best thing. For example in the below code snippet you can see we have a simple action called as “DynamicView”. Depending on the flag (“IsHtmlView”) it will either return “ViewResult” or “JsonResult”.

```
public ActionResult DynamicView()
{
    if (IsHtmlView)
        return View(); // returns simple ViewResult
    else
        return Json(); // returns JsonResult view
}
```

What are the different types of results in MVC?

Note:-It’s difficult to remember all the 12 types. But some important ones you can remember for the interview are “ActionResult”, “ViewResult” and “JsonResult”. Below is a detailed list for your interest.

There 12 kinds of results in MVC, at the top is “ActionResult” class which is a base class that can have 11 subtypes as listed below: -

1. ViewResult - Renders a specified view to the response stream

2. PartialViewResult - Renders a specified partial view to the response stream
3. EmptyResult - An empty response is returned
4. RedirectResult - Performs an HTTP redirection to a specified URL
5. RedirectToRouteResult - Performs an HTTP redirection to a URL that is determined by the routing engine, based on given route data
6. JsonResult - Serializes a given ViewData object to JSON format
7. JavaScriptResult - Returns a piece of JavaScript code that can be executed on the client
8. ContentResult - Writes content to the response stream without requiring a view
9. FileContentResult - Returns a file to the client
10. FileStreamResult - Returns a file to the client, which is provided by a Stream
11. FilePathResult - Returns a file to the client

What are “ActionFilters”in MVC?

“ActionFilters” helps you to perform logic while MVC action is executing or after a MVC action has executed.

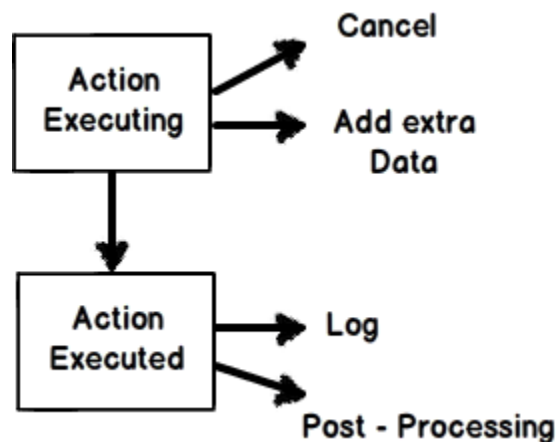


Figure 6.10:- “ActionFilters”in MVC

Action filters are useful in the following scenarios:-

1. Implement post-processing logic before the action happens.
2. Cancel a current execution.
3. Inspect the returned value.
4. Provide extra data to the action.

You can create action filters by two ways:-

- Inline action filter.

- Creating an “ActionFilter” attribute.

To create a inline action attribute we need to implement “IActionFilter” interface. The “IActionFilter” interface has two methods “OnActionExecuted” and “OnActionExecuting”. We can implement pre-processing logic or cancellation logic in these methods.

```
public class Default1Controller : Controller , IActionFilter
{
    public ActionResult Index(Customer obj)
    {
        return View(obj);
    }
    void IActionFilter.OnActionExecuted(ActionExecutedContext
filterContext)
    {
        Trace.WriteLine("Action Executed");
    }
    void IActionFilter.OnActionExecuting(ActionExecutingContext
filterContext)
    {
        Trace.WriteLine("Action is executing");
    }
}
```

The problem with inline action attribute is that it cannot be reused across controllers. So we can convert the inline action filter to an action filter attribute. To create an action filter attribute we need to inherit from “ActionFilterAttribute” and implement “IActionFilter” interface as shown in the below code.

```
public class MyActionAttribute : ActionFilterAttribute , IActionFilter
{
    void IActionFilter.OnActionExecuted(ActionExecutedContext filterContext)
    {
        Trace.WriteLine("Action Executed");
    }
    void IActionFilter.OnActionExecuting(ActionExecutingContext filterContext)
    {
        Trace.WriteLine("Action executing");
    }
}
```

```
}  
}
```

Later we can decorate the controllers on which we want the action attribute to execute. You can see in the below code I have decorated the “Default1Controller” with “MyActionAttribute” class which was created in the previous code.

```
[MyActionAttribute]  
public class Default1Controller : Controller  
{  
    public ActionResult Index(Customer obj)  
    {  
        return View(obj);  
    }  
}
```

Can we create our custom view engine using MVC?

Yes, we can create our own custom view engine in MVC. To create our own custom view engine we need to follow 3 steps:-

Let’ say we want to create a custom view engine where in the user can type a command like “<DateTime>” and it should display the current date and time.

Step 1:- We need to create a class which implements “IView” interface. In this class we should write the logic of how the view will be rendered in the “render” function. Below is a simple code snippet for the same.

```
public class MyCustomView : IView  
{  
  
    private string _FolderPath; // Define where our views are stored  
  
    public string FolderPath  
    {  
        get { return _FolderPath; }  
        set { _FolderPath = value; }  
    }  
}
```

```

        public void Render(ViewContext viewContext, System.IO.TextWriter
writer)
        {
            // Parsing logic <dateTime>
            // read the view file
            string strFileData = File.ReadAllText(_FolderPath);
            // we need to and replace <datetime> datetime.now value
            string strFinal = strFileData.Replace("<DateTime>",
DateTime.Now.ToString());
            // this replaced data has to sent for display
            writer.Write(strFinal);
        }
    }
}

```

Step 2 :-We need to create a class which inherits from “VirtualPathProviderViewEngine” and in this class we need to provide the folder path and the extension of the view name. For instance for razor the extension is “cshtml”, for aspx the view extension is “.aspx”, so in the same way for our custom view we need to provide an extension. Below is how the code looks like. You can see the “ViewLocationFormats” is set to the “Views” folder and the extension is “.myview”.

```

public class MyViewEngineProvider : VirtualPathProviderViewEngine
{
    // We will create the object of Mycustome view
    public MyViewEngineProvider() // constructor
    {
        // Define the location of the View file
        this.ViewLocationFormats = new string[] {
"~/Views/{1}/{0}.myview", "~/Views/Shared/{0}.myview" }; //location and
extension of our views
    }

    protected override IView CreateView(ControllerContext
controllerContext, string viewPath, string masterPath)
    {
        var physicalpath =
controllerContext.HttpContext.Server.MapPath(viewPath);
    }
}

```

```

        MyCustomView obj = new MyCustomView(); // Custom view engine
class
        obj.FolderPath = physicalpath; // set the path where the views
will be stored
        return obj; // returned this view parsing logic so that it can
be registered in the view engine collection
    }
    protected override IView CreatePartialView(ControllerContext
controllerContext, string partialPath)
    {
        var physicalpath =
controllerContext.HttpContext.Server.MapPath(partialPath);
        MyCustomView obj = new MyCustomView(); // Custom view engine
class
        obj.FolderPath = physicalpath; // set the path where the views
will be stored
        return obj; // returned this view parsing logic so that it can
be registered in the view engine collection
    }
}

```

Step 3:- We need to register the view in the custom view collection. The best place to register the custom view engine in the “ViewEngines” collection is the “global.asax” file. Below is the code snippet for the same.

```

protected void Application_Start()
{
    // Step3 :- register this object in the view engine collection
    ViewEngines.Engines.Add(new MyViewEngineProvider());
    ....
}

```

Below is a simple output of the custom view written using the commands defined at the top.

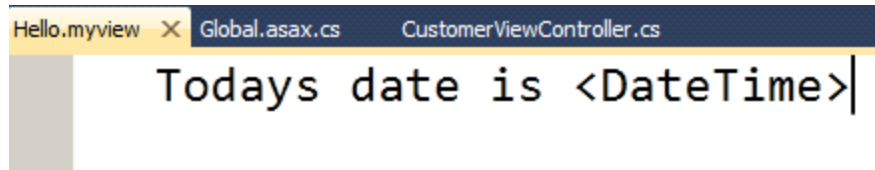


Figure 6.11:- custom view engine using MVC

If you invoke this view you should see the following output.

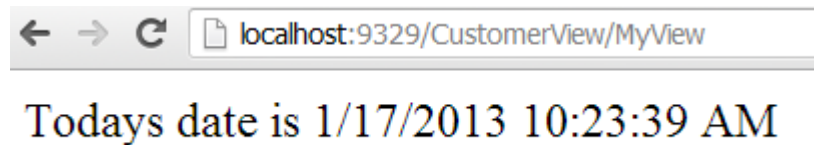


Figure 6.12:- output

How to send result back in JSON format in MVC?

In MVC we have “JsonResult” class by which we can return back data in JSON format. Below is a simple sample code which returns back “Customer” object in JSON format using “JsonResult”.

```
public JsonResult getCustomer()
{
    Customer obj = new Customer();
    obj.CustomerCode = "1001";
    obj.CustomerName = "Shiv";
    return Json(obj, JsonRequestBehavior.AllowGet);
}
```

Below is the JSON output of the above code if you invoke the action via the browser.

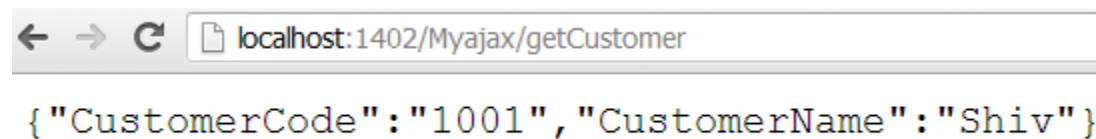


Figure 6.13:- JSON format in MVC

What is “WebAPI”?

HTTP is the most used protocol. For past many years browser was the most preferred client by which we can consume data exposed over HTTP. But as years passed by client variety started spreading out. We had demand to consume data on HTTP from clients like mobile, javascripts, windows application etc.

For satisfying the broad range of client “REST” was the proposed approach. You can read more about “REST” from WCF chapter.

“WebAPI” is the technology by which you can expose data over HTTP following REST principles.

But WCF SOAP also does the same thing, so how does “WebAPI” differ?

	SOAP	WEB API
Size	Heavy weight because of complicated WSDL structure.	Light weight, only the necessary information is transferred.
Protocol	Independent of protocols.	Only for HTTP protocol
Formats	To parse SOAP message, the client needs to understand WSDL format. Writing custom code for parsing WSDL is a heavy duty task. If your client is smart enough to create proxy objects like how we have in .NET (add reference) then SOAP is easier to consume and call.	Output of “WebAPI” are simple string message, JSON, Simple XML format etc. So writing parsing logic for the same is very easy.
Principles	SOAP follows WS-* specification.	WEB API follows REST principles. (Please refer about REST in WCF chapter).

With WCF also you can implement REST, So why “WebAPI”?

WCF was brought in to implement SOA, never the intention was to implement REST. “WebAPI” is built from scratch and the only goal is to create HTTP services using REST. Due to the one point focus for creating “REST” service “WebAPI” is more preferred.

How to implement “WebAPI” in MVC?

Below are the steps to implement “webAPI” :-

Step1:-Create the project using the “WebAPI” template.

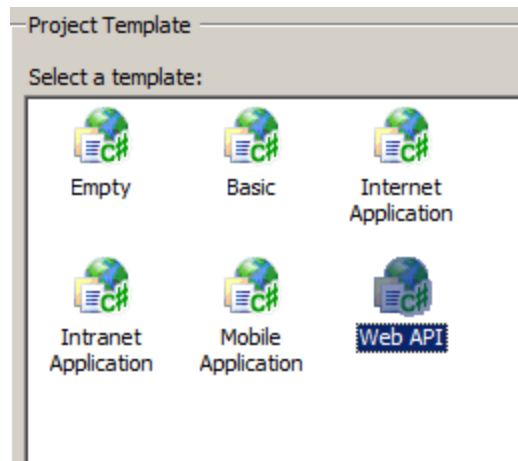


Figure 6.14:- implement “WebAPI” in MVC

Step 2:- Once you have created the project you will notice that the controller now inherits from "ApiController" and you can now implement "post", "get", "put" and "delete" methods of HTTP protocol.

```
public class ValuesController : ApiController
{
    // GET api/values
    public IEnumerable<string> Get()
    {
        return new string[] { "value1", "value2" };
    }

    // GET api/values/5
    public string Get(int id)
    {
        return "value";
    }

    // POST api/values
    public void Post([FromBody]string value)
    {
    }

    // PUT api/values/5
    public void Put(int id, [FromBody]string value)
    {
    }
}
```

```
// DELETE api/values/5
public void Delete(int id)
{
}
}
```

Step 3:- If you make a HTTP GET call you should get the below results.

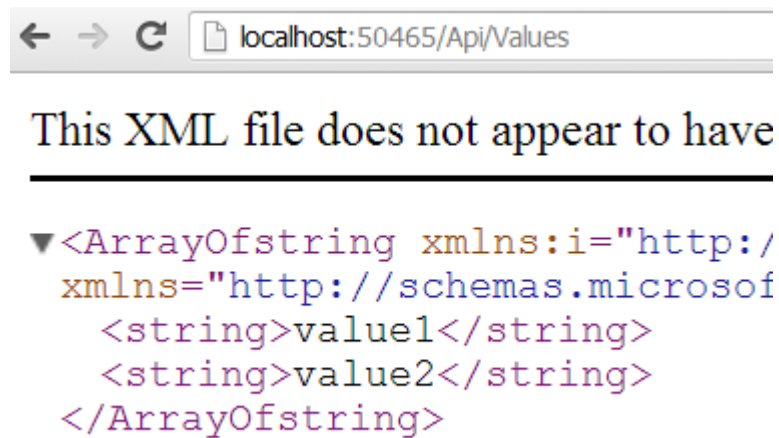


Figure 6.15:- HTTP

How can we detect that a MVC controller is called by POST or GET ?

To detect if the call on the controller is “POST” action or a “GET” action we can use “Request.HttpMethod” property as shown in the below code snippet.

```
public ActionResult SomeAction()
{
    if (Request.HttpMethod == "POST")
    {
        return View("SomePage");
    }
    else
    {
        return View("SomeOtherPage");
    }
}
```

```
}  
  
}
```

What is Bundling and minification in MVC?

Bundling and minification helps us to improve request load time of a page thus increasing performance.

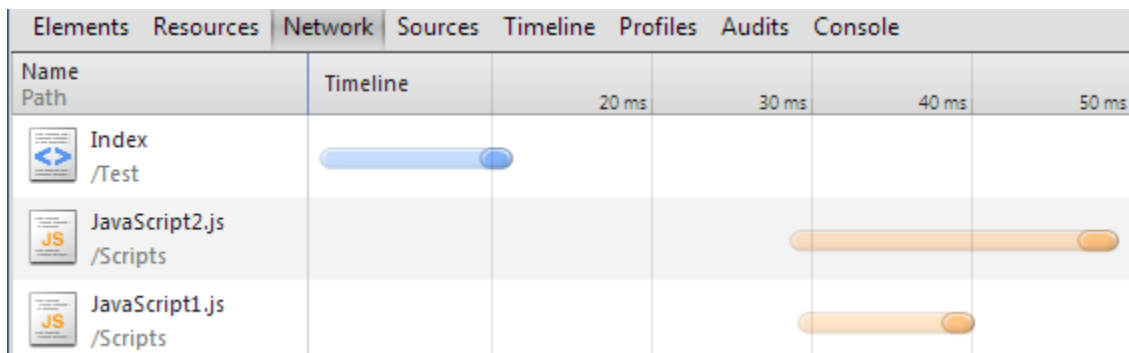
How does bundling increase performance?





Web projects always need CSS and script files. Bundling helps us to combine multiple javascript and CSS file into a single entity thus minimizing multiple requests into a single request.

For example consider the below web request to a page. This page consumes two JavaScript files “Javascript1.js” and “Javascript2.js”. So when this page is requested it makes three request calls:-

- One for the “Index” page.
- Two requests for the other two JavaScript files “Javascript1.js” and “Javascript2.js”.

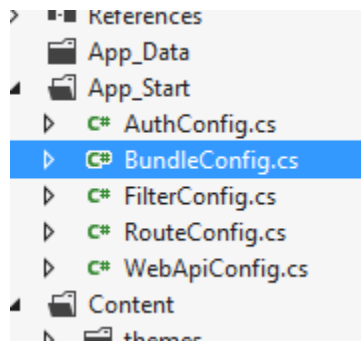
Below scenario can become worst if we have lot of javascript files resulting in many multiple requests thus decreasing performance. If we can somehow combine all the JS files into a single bundle and request them as a single unit that would result in increased performance (See the next figure which has a single request).



Elements Resources Network Sources Timeline >>	
Name Path	Timeline
 Index /Test	
 MyScripts?v=E... /Scripts	

So how do we implement bundling in MVC ?

Open the “BundleConfig.cs” from the “App_Start” folder.



In the “BundleConfig.cs” add the JS files which you want bundle in to single entity in to the bundles collection. In the below code we are combining all the javascript JS files which are existing in the “Scripts” folder as a single unit in to the bundle collection.

```
bundles.Add(new ScriptBundle("~/Scripts/MyScripts").Include(
    "~/Scripts/*.js"));
```

Below is how your “BundleConfig.cs” file will look like.

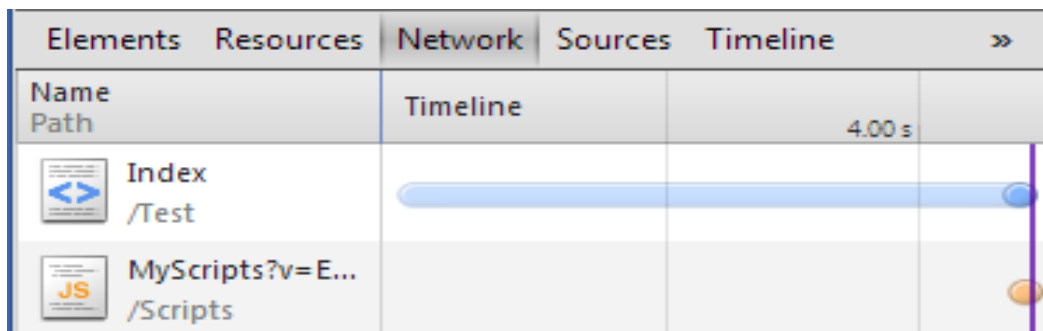
```
public class BundleConfig
{
    public static void RegisterBundles(BundleCollection bundles)
    {
        bundles.Add(new ScriptBundle("~/Scripts/MyScripts").Include(
            "~/Scripts/*.js"));
        BundleTable.EnableOptimizations = true;
    }
}
```

```
}  
}
```

Once you have combined your scripts in to one single unit we then to include all the JS files in to the view using the below code. The below code needs to put in the ASPX or RAZOR view.

```
<%= Scripts.Render("~/Scripts/MyScripts") %>
```

If you now see your page requests you would see that script request is combined in to a one request.



How can you test bundling in debug mode ?

If you are in a debug mode you need to set “EnableOptimizations” to true in the “bundleconfig.cs” file or else you will not see the bundling effect in the page requests.

```
BundleTable.EnableOptimizations = true;
```

Explain minification and how to implement the same ?

Minification reduces the size of script and CSS files by removing blank spaces , comments etc. For example below is a simple javascript code with comments.

```
// This is test  
var x = 0;  
x = x + 1;  
x = x * 2;
```

After implementing minification the javascript code looks something as below. You can see how whitespaces and comments are removed to minimize file size and thus increasing performance.

```
var x=0;x=x+1;x=x*2;
```

How to implement minification ?

When you implement bundling minification is implemented by itself. In other words steps to implement bundling and minification are same.

What is the use of “AllowHTML” and “ValidateInput” attributes?

While working with MVC you may have noticed that MVC actions doesn't allow posting HTML values . This is done to avoid cross site scripting security issues. Look at the following example where we are trying to post HTML to the MVC action.

ArticleTitle

ArticleContent



Server Error in '/' Application.

A potentially dangerous Request.Form value was detected from the client (ArticleContent="Welcome</E

Description: ASP.NET has detected data in the request that is

But you can bypass this restriction by using one of the following 2 attributes

- 1) ValidateInput attribute at controller level or action level

```
[HttpPost()]
[ValidateInput(false)]
public ActionResult Index(ArticleData o)
{
    return View(o);
}
```

- 2) AllowHtml attribute at Mode Level

```
public class ArticleData
{
    [AllowHtml()]
    public string ArticleContent { get; set; }
    public string ArticleTitle { get; set; }
}
```