**Understanding URL Rewriting and URL Attribute Routing in ASP.NET MVC (MVC5) with Examples**

## How to Enabling Attribute Routing in MVC5

To enable attribute routing, call MapMvcAttributeRoutes during configuration. Following are the code snipped.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | public class RouteConfig  {      public static void RegisterRoutes(RouteCollection routes)      {          routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");          routes.MapMvcAttributeRoutes();      }  } |

In MVC5, we can combine attribute routing with convention-based routing. Following are the code snipped.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | public static void RegisterRoutes(RouteCollection routes)  {      routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");      routes.MapMvcAttributeRoutes();      routes.MapRoute(          name: "Default",          url: "{controller}/{action}/{id}",          defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }      );  } |

## Optional URI Parameters and Default Values in Attribute Routing in MVC5

It is very easy to make a URI parameter optional by adding a question mark to the route parameter. We can also specify a default value by using the form parameter=value. Following are a code sample shows, how it can be done.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | public class StudentController : Controller  {      // eg: /students      // eg: /students/1      [Route("students/{id?}")]      public ActionResult View(string id)      {          if (!String.IsNullOrEmpty(id))          {              return View("ViewStudent", GetStudent(id));          }          return View("AllStudents", GetStudents());      }        // eg: /students/category      // eg: /students/category/general      // eg: /students/category/obc      [Route("students/category/{category=general}")]      public ActionResult ViewByCategory(string category)      {          return View("ViewStudent", GetStudentByCategory(category));      }  } |

## Route Prefixes (Prefix in Url in MVC)

N ASP.NET MVC5 the routes in a controller all start with the same prefix. For example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | public class StudentsController : Controller  {      // eg: /students      [Route("students")]      public ActionResult Index() { ... }      // eg: /students/5      [Route("students/{studentId}")]      public ActionResult Show(int studentId) { ... }      // eg: /students/5/edit      [Route("students/{studentId}/edit")]      public ActionResult Edit(int studentId) { ... }  } |

To achieve same features, we can set a common prefix for an entire controller by using the [RoutePrefix] attribute. Following are the code sample.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | [RoutePrefix("students ")]  public class StudentsController : Controller  {      // eg.: /students      [Route]      public ActionResult Index() { ... }      // eg.: /students/5      [Route("{studentId}")]      public ActionResult Show(int studentId) { ... }      // eg.: /students/5/edit      [Route("{studentId}/edit")]      public ActionResult Edit(int studentId) { ... }  } |

To change this Url Prefix, we must use a tilde (~) on the method attribute to override the route prefix. Following code sample shows how it can be done.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | [RoutePrefix("students")]  public class StudentsController : Controller  {      // eg.: /college-students      [Route("~/college-students")]      public ActionResult ShowStudents() { ... }      //TODO: any logic      ...  } |

## Default Route in MVC5

In some context we may want to apply the [Route] attribute on the controller level, and trying to capture the action as a parameter. That route would then be applied on all actions in the controller, unless a specific [Route] has been defined on a specific action, overriding the default set on the controller. Following are a code sample to do same.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | [RoutePrefix("students")]  [Route("{action=index}")]  public class StudentsController : Controller  {      // eg.: /students      public ActionResult Index() { ... }        // eg.: /students/archive      public ActionResult Archive() { ... }        // eg.: /students/new      public ActionResult New() { ... }        // eg.: /students/edit/5      [Route("edit/{studentId:int}")]      public ActionResult Edit(int studentId) { ... }  }  Route Constraints in MVC5  Route constraints allow us to restrict how the parameters in the route template are matched. The general syntax is {parameter:constraint}. Following code sample shows, how it could be achieved:  // eg: /students/5  [Route("students/{id:int}"]  public ActionResult GetStudentById(int id) { ... }    // eg: students/ramesh  [Route("students/{name}"]  public ActionResult GetStudentByName(string name) { ... } |

## Route Names

We can specify a name for a route, this name will help us to easily allow URI generation for it in our application. For example, for the following route:

|  |  |
| --- | --- |
| 1  2 | [Route("dashboard", Name = "admindashboard")]  public ActionResult AdminDashboard() { ... } |

We could generate a link using Url.RouteUrl:

|  |  |
| --- | --- |
| 1 | <a href="@Url.RouteUrl(" admindashboard")">Main Dashboard</a> |

## Areas

Areas in MVC is one of the important features to be used in categories code files in large application. We can define that a controller belongs to an area by using the [RouteArea] attribute. When doing so, you can safely remove the AreaRegistration class for that area.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | [RouteArea("Admin")]  [RoutePrefix("dashboard")]  [Route("{action}")]  public class AdminController : Controller  {      // eg: /admin/dashboard/login      public ActionResult Login() { ... }        // eg: /admin/dashboard/show-options      [Route("show-options")]      public ActionResult Options() { ... }        // eg: /stats      [Route("~/stats")]      public ActionResult Stats() { ... }  } |

With the help of above controller, the following link generation call will result with the string "/Admin/dashboard/show-options"

|  |  |
| --- | --- |
| 1 | Url.Action("Options", "Dashboard", new { Area = "Admin" }) |

MVC5 also allow us to set up our custom prefix for the area that defer from the area name, We need to use AreaPrefix named parameter, for example:

|  |  |
| --- | --- |
| 1 | [RouteArea("BackOffice", AreaPrefix = "back-office")] |

In some cases, there may be requirement to use both Areas and Route Attributes and may be Areas with convention based routes, Which is set by an AreaRegistration class, then you need to make sure that area registration happen after MVC attribute routes are configured, however before the default convention-based route is set. The reason behind this is that route registration should be ordered from the most specific (attributes) through more general (area registration) to the mist generic (the default route) to avoid generic routes from “hiding” more specific routes by matching incoming requests too early in the pipeline. Following are code snipped to do the same:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | public static void RegisterRoutes(RouteCollection routes)  {      routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");        routes.MapMvcAttributeRoutes();        AreaRegistration.RegisterAllAreas();        routes.MapRoute(          name: "Default",          url: "{controller}/{action}/{id}",          defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }      );  } |