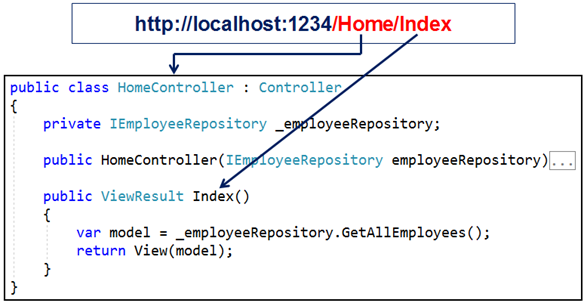
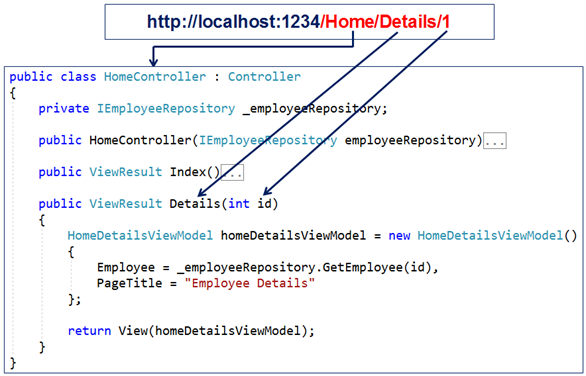
**Routing in ASP.NET Core MVC**

There are 2 routing techniques in ASP.NET Core MVC. **Conventional Routing**and **Attribute Routing**. In this video, we will discuss Conventional Routing and in our next video we will discuss Attribute Routing.   
  
  
When a request from the browser arrives at our application, it is the controller in the MVC design pattern, that handles the incoming http request and responds to the user action. The incoming request URL is mapped to a controller action method. This mapping is done by the routing rules defined in our application.   
  
For example, when a request is issued to /Home/Index, this URL is mapped to the Index() action method in the HomeController class.   
  
  
  
Similarly, when a request is issued to /Home/Details/1, this URL is mapped to the Details() action method in the HomeController class. The value 1 in the URL is automatically mapped to the "id" parameter of the Details(int id) action method.   
  
  
  
At the moment, we have not explicitly defined any routing rules in our ASP.NET Core MVC application. So the question that comes to our mind is, how is this URL (/Home/Index) mapped to the Index() action method in the HomeController.   
  
**Default Route in ASP.NET Core MVC**   
  
The following is the code in the Configure() method in Startup.cs file. The code in this method sets up the HTTP request processing pipeline. We discussed this method in our previous videos of this ASP.NET Core tutorial.

public void Configure(IApplicationBuilder app, IHostingEnvironment env)  
{  
    if (env.IsDevelopment())  
    {  
        app.UseDeveloperExceptionPage();  
    }  
  
    app.UseStaticFiles();  
  
    app.UseMvcWithDefaultRoute();  
}

In this method we have a call to UseMvcWithDefaultRoute() extension method. It is this method, that adds MVC with the following default route to our application's request processing pipeline.

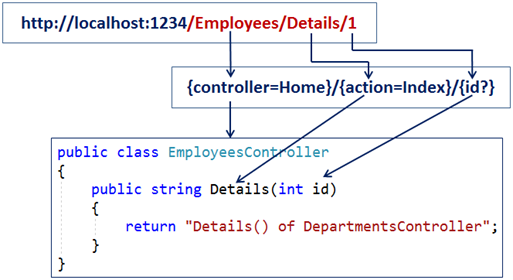
{controller=Home}/{action=Index}/{id?}

You can see the default route from the intellisense, just by hovering the mouse over UseMvcWithDefaultRoute() in visual studio. Also, ASP.NET Core is opensource. So you can also see the code of UseMvcWithDefaultRoute() method on their GITHUB page at the following link.   
  
<https://github.com/aspnet/Mvc/blob/release/2.2/src/Microsoft.AspNetCore.Mvc.Core/Builder/MvcApplicationBuilderExtensions.cs>   
  
**Code in UseMvcWithDefaultRoute()**   
  
For your quick reference, the following is the code from the above GITHUB page. Notice, UseMvcWithDefaultRoute() method internally calls UseMvc() method and it sets up the default route.

public static IApplicationBuilder UseMvcWithDefaultRoute(this IApplicationBuilder app)  
{  
    if (app == null)  
    {  
        throw new ArgumentNullException(nameof(app));  
    }  
  
    return app.UseMvc(routes =>  
    {  
        routes.MapRoute(  
            name: "default",  
            template: "{controller=Home}/{action=Index}/{id?}");  
    });  
}

**Understanding the default route**  
  
The default route template "{controller=Home}/{action=Index}/{id?}" maps most URL's that have the following pattern   
  
http://localhost:1234/Employees/Details/1   
  
In the above example

|  |  |
| --- | --- |
| **Path Segment** | **Maps to** |
| /Employees | EmployeesController class |
| /Details | Details(int id) action method |
| /1 | id parameter of the Details(int id) action method |

   
  
The first URL path segment "/Employees" is mapped to the "EmployeesController". In the URL we do not have to have the word Controller. When MVC finds the word /Employee in the first URL path segment, it appends the word Controller and looks for a class with name EmployeesController.   
  
The second URL path segment "/Details" is mapped to the Details(int id) action method in the EmployeesController.   
  
The third path segment **"1"** is mapped to the id parameter of the Details(int id) action method. This is done by a process called **model binding**. We will discuss model binding in detail in our upcoming videos.   
  
Notice in the following default route template, we have a question mark after the id parameter.   
The default route template **"{controller=Home}/{action=Index}/{id?}"**   
  
The question mark, makes the id parameter in the URL optional. This means both the following URLs will be mapped to the Details() action method of the EmployeesController class.   
  
/Employees/Details/1  
/Employees/Details   
  
The value "Home" in {controller=Home} is the default value for the Controller. Similarly the value "Index" in {action=Index} is the default value for the action method.    
  
So with these default values in place, if we navigate to the application root URL like the following. Notice in the URL we do not have the Controller and Action method path segments. So the default values will be used and the request will be mapped to the Index() action method of the HomeController class.  
http://localhost:1234   
  
Along the same lines both the following request URL's will also be mapped to the Index() action method of the HomeController class.  
http://localhost:1234/Home  
http://localhost:1234/Home/Index   
  
For most applications the default route works just fine. For example, if you have the following DepartmentsController.

public class DepartmentsController : Controller  
{  
    public string List()  
    {  
        return "List() of DepartmentsController";  
    }  
  
    public string Details()  
    {  
        return "Details() of DepartmentsController";  
    }  
}

The URL "/departments/list" is mapped to the List() action method of the DepartmentsController and the URL "/departments/details" is mapped to the Details()action method of the DepartmentsController.   
  
**UseMvc or UseMvcWithDefaultRoute**   
  
If you want to define your own route templates and want to have more control over the routes, use UseMvc() method, instead of UseMvcWithDefaultRoute() method.

app.UseMvc(routes =>  
{  
    routes.MapRoute("default", "{controller=Home}/{action=Index}/{id?}");  
});