

TagHelpers in ASP.NET Core MVC

-

What are Tag Helpers in ASP.NET Core?

- ▶ Tag Helpers in ASP.NET Core are the server-side components. They are basically used to perform defined transformations on HTML Elements. As they are server-side components, so they are going to be processed on the server to create and render HTML elements in the Razor files.
- ▶ If you are coming from ASP.NET MVC background, then you may be worked with the HTML helpers. The Tag Helpers are similar to the HTML helpers.

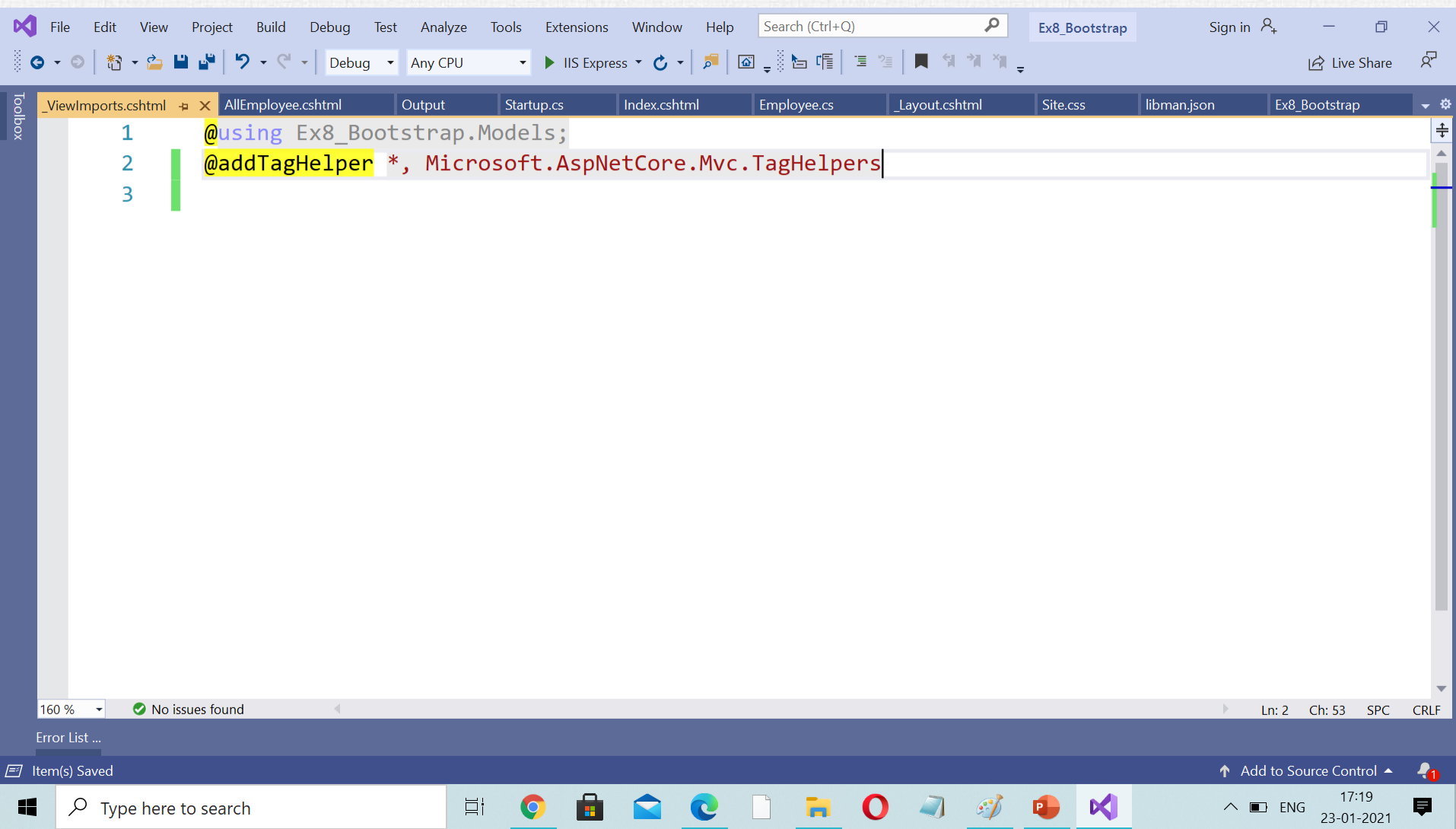
Types of Tag Helpers in ASP.NET Core:

- ▶ There are two types of Tag helpers in ASP.NET Core. They are as follows:
 - ▶ **Built-In Tag Helpers:** They come in-built in the ASP.NET Core Framework and can perform common tasks like generating links, creating forms, loading assets, showing validation messages, etc.
 - ▶ **Custom Tag Helper:** That can be created by us to perform our desired transformation on an HTML element.
- ▶ **Note:** In this video, I am going to discuss the Built-in Tag Helpers and in our upcoming articles, I will discuss the Custom Tag Helpers.

How to use built-in Tag Helpers in ASP.NET Core MVC?

- ▶ In order to make the built-in tag helpers available for all the views of our application, import the tag helpers using the **_ViewImports.cshtml** file.
- ▶ You need to import them using the **@addTagHelper** directive as shown below.
 - ▶ **@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers**
- ▶ The **@addTagHelper** makes the built-in tag helpers to available in the application which are defined in an assembly called **Microsoft.AspNetCore.Mvc.TagHelpers**.
- ▶ Here the wildcard **"*"** specifies that all the Tag Helpers are made available.

Add `@addTagHelper *,Microsoft.AspNetCore.Mvc.TagHelpers;` inside `_ViewImports` file means we can use any tag helper in any view page.



The screenshot shows the Visual Studio IDE with the `_ViewImports.cshtml` file open. The file contains the following code:

```
1 @using Ex8_Bootstrap.Models;  
2 @addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers  
3
```

The `@addTagHelper` directive is highlighted in yellow. The Visual Studio interface includes the menu bar, toolbar, and a taskbar at the bottom with various application icons.

Anchor TagHelper in ASP.NET Core MVC

Generating Links using Tag Helpers in ASP.NET Core MVC Application:

- ▶ We have many different options to generate a link in ASP.NET Core MVC Application. Let us discuss all the possible options to generate a link and then we will discuss why we should use Tag Helper over others.
- ▶ Manually generating the links:
- ▶ Using HTML helpers
- ▶ Using Tag Helpers:

Using Anchor Tag Helper:

- ▶ In order to use Tag Helpers first import the **@addTagHelper** directive in the **_ViewImport.cshtml** file. Along with the **@addTagHelper** directive, we also add the model namespace using the **@using** directive.
 - ▶ `@using FirstCoreMVCApplication.Models;`
 - ▶ `@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers`
- ▶ Then modify the view as shown below.
 - ▶ `<a asp-controller="Home" asp-action="Details" asp-route-id="@student.StudentId">View`

Understanding the Anchor Tag Helper in ASP.NET Core:

- ▶ The Anchor Tag Helper in ASP.NET Core creates the standard HTML anchor (`<a ... >`) tag by adding new attributes such as:
 - ▶ **asp-controller**: It is used to specify the controller to target based on the routing system. If you omitted this, then the controller of the current view is used by default.
 - ▶ **asp-action**: It is used to specify the Action method to target based on the routing system. If you omitted this attribute then the action rendering the current view is used by default.
 - ▶ **asp-route-{value}**: It is used for specifying the additional segment value for the URL. For example, `asp-route-id` is used to provide value for the 'id' segment.
- ▶ The rendered anchor element's "href" attribute value is determined by the values of these "asp-" attributes.
 - ▶ As the name says, **asp-controller** specifies the name of the controller
 - ▶ whereas **asp-action** specifies the name of the action name.
 - ▶ Similarly, the **asp-route-{value}** attribute is used to include route data in the generated href attribute value. **{value}** can be replaced with route parameters such as id, name, etc.


```
<a asp-controller="Emp" asp-action="Search" asp-route-id="3">View</a>
```

Example:

```
<a asp-controller="Home" asp-action="Details"  
  asp-route-id="@student.StudentId">View</a>
```

Generates the following HTML

```
<a href="/Home/details/101">View</a>
```

3 way Hyperlink is created

The screenshot displays the Visual Studio IDE with the file `AllEmployee.cshtml` open. The code is in Razor syntax, showing a table structure. A 3-way hyperlink is being created in the table body. The code is as follows:

```
19 </tr>
20 </thead>
21 <tbody>
22     @foreach (var i in Model)
23     {
24         <tr>
25             <td>@i.Id</td>
26             <td>@i.Name</td>
27             <td>@i.Salary</td>
28             <td></td>
29             <td><a href="/Employee/Search/@i.Id">View</a></td>
30             <td>@Html.ActionLink("View", "Search", "Employee", new {id=i.Id })</td>
31             <td><a asp-controller="Employee" asp-action="Search" asp-route-id="@i.Id">View</a>
32         </tr>
33     }
34
35
36 </tbody>
37
```

The third column of the table (line 29) contains a 3-way hyperlink created using the `asp-controller`, `asp-action`, and `asp-route-id` attributes. The first column (line 25) shows the ID, the second column (line 26) shows the Name, and the fourth column (line 28) is empty.

The Visual Studio interface includes the following elements:

- Menu Bar:** File, Edit, View, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q).
- Toolbox:** Visible on the left side.
- Output Window:** Shows "No issues found".
- Taskbar:** Includes icons for Windows, Search, and various applications like Chrome, File Explorer, and Visual Studio.
- System Tray:** Shows the time as 17:20 on 23-01-2021.

Action using TagHelper

- `<td><a asp-controller="Employee" asp-action="Search" asp-route-id="@i.Id">View </td>`
- `<td><a asp-route-id="@i.Id">View </td>`
- `<td><a asp-action="Search" asp-route-id="@i.Id">View </td>`
- Output: html

```
<td><a asp-controller="Employee"
href="/Employee/Search/3">View</a>
</td>
```

```
<td><a
href="/Employee/AllEmployee/3">View
</a> </td>
```

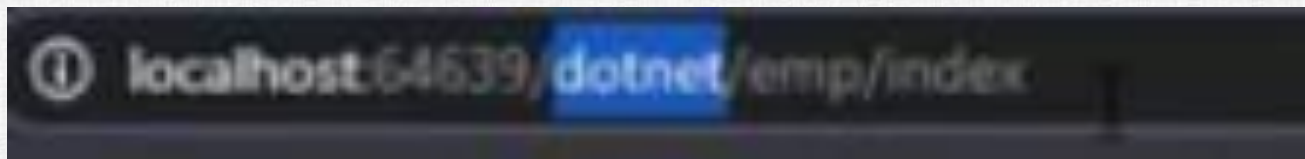
```
<td><a
href="/Employee/Search/3">View</a>
</td>
```


If we change the Routing Templates.
Before controller we put dotnet

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

```
s.MapRoute("default", "dotnet/{controller=Home}/{action=index}/{id?}")
```

- Run:-before controller put dotnet prefix :-
hyperlist work file using tag helper but not work
using hyperlink.



```
<td><a href="/emp/details/@emp.EmpId">View 1</a> | &nt  
<a asp-action="Details" asp-route-id="@emp.EmpId" -  
/tr>
```


Thanks