Microsoft®
.NET

UNIT 14:
FORMS AND HTML HELPERS

CONTENTS

- Using Forms
- HTML Helpers
- Rendering Helpers

Using Forms

- A form is a container for input elements: buttons, checkboxes, text inputs, and more.
- Two most important attributes of a form tag: the action and the method attributes.
- Using Html Helper

```
<form action="/Home/Search"
method="get">
   <input type="text" name="q" />
   <input type="submit" value="Search"
/>
</form>
```

```
@using (Html.BeginForm("Search",
"Home", FormMethod.Get))
{
   <input type="text" name="q" />
   <input type="submit" value="Search" />
}
```

HTML Helpers

- HtmlHelper class generates html elements using the model class object in razor view.
- It binds the model object to html elements to display value of model properties into html elements and also assigns the value of the html elements to the model properties while submitting web form.

HTML Helpers

- HTML helpers are methods you can invoke on the Html property of a view.

```
@model IEnumerable<MVC_BasicTutorials.Models.Student>

@{
    ViewBag.Title = "Index";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<h2>Index</h2>

<p>
    @Html.ActionLink("Create New", "Create")
</p>
<table class="table">
    <tr>
        <th>
            @Html.DisplayNameFor(model => model.StudentName)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.Age)
        </th>
        <th></th>
    </tr>
```

Extension method for HtmlHelper

HtmlHelper

HTML Helpers

- @Html is an object of HtmlHelper class .
- HtmlHelper class generates html elements.
- Ex:

@Html.ActionList("Create New", "Create") would generate anchor tag

Create New.

- The difference between calling the HtmlHelper methods and using an html tags is that the HtmlHelper method is designed to make it easy to bind to view data or model data.

HTML Helpers

- **Some Html Helpers**

HtmlHelper	Strogly Typed HtmlHelpers	Generates Html Control
Html.TextBox	Html.TextBoxFor	Textbox
Html.TextArea	Html.TextAreaFor	TextArea
Html.CheckBox	Html.CheckBoxFor	Checkbox
Html.RadioButton	Html.RadioButtonFor	Radio button
Html.DropDownList	Html.DropDownListFor	Dropdown, combobox
Html.ListBox	Html.ListBoxFor	multi-select list box
Html.Hidden	Html.HiddenFor	Hidden field
Password	Html.PasswordFor	Password textbox
Html.Display	Html.DisplayFor	Html text
Html.Label	Html.LabelFor	Label
Html.Editor	Html.EditorFor	Generates Html controls based on data type of specified model property e.g. textbox for string property, numeric field for int, double or other numeric type.

HTML Helpers

- Html Helper doesn't give you compile time error if you have specified wrong property name. It will throw run time exception.
- Strongly typed Html Helper is generic method so it will give you compile time error if you have specified wrong property name or property name changes. (Provided view is not compile at run time.)

TextBox

- **Use TextBox():**

MvcHtmlString Html.TextBox(string name, string value, object htmlAttributes)

- **The TextBox() method is a loosely typed method because name parameter is a string.**
- **The name parameter can be a property name of model object. It binds specified property with textbox. Ex:**

```
@model
Student @Html.TextBox("StudentName", null, new { @class
= "form-control" })
```

TextBox

- Use TextBoxFor()

MvcHtmlString TextBoxFor(Expression<Func<TModel,TValue>> expression, object htmlAttributes)

- TextBoxFor helper method is a strongly typed extension method. It generates a text input element for the model property specified using a lambda expression. TextBoxFor method binds a specified model object property to input text.
- Ex:

```
@model Student

@Html.TextBoxFor(m => m.StudentName, new { @class = "form-control" })
```

TextArea

- **Use TextArea():**

MvcHtmlString Html.TextArea(string name, string value, object htmlAttributes)

- **The TextBox() method is a loosely typed method because name parameter is a string.**
- **By default, it creates textarea with rows=2 and cols=20. Ex:**

```
@model Student

@Html.TextBox(“Description", null, new { @class
= "form-control" })
```

TextArea

- **Use TextAreaFor()**

MvcHtmlString TextAreaFor(Expression<Func<TModel,TValue>> expression, object htmlAttributes)

- **TextAreaFor helper method is a strongly typed extension method. It generates a multi line <textarea> element for the property in the model object specified using a lambda expression. TextAreaFor method binds a specified model object property to textarea element.**

```
@model Student

@Html.TextBox(m=>m.Description, null, new { @class = "form-control" })
```

Password

- HtmlHelper class includes two extension methods to generate a password field (<input type="password">) element in a razor view: Password() and PasswordFor().
- Ex:

```
@model Student
@Html.Password("OnlinePassword")
@Html.PasswordFor(m => m.Password)
```

Hidden Field

- HtmlHelper class includes two extension methods to generate a hidden field (<input type="hidden">) element in a razor view: Hidden() and HiddenFor().
- Ex:

```
@model Student

@Html.Hidden("StudentId")

@Html.HiddenFor(m => m.StudentId)
```

Label

- HtmlHelper class includes two extension methods to generate html label : Label() and LabelFor().
- Ex:
- @Html.Label("StudentName")

@Html.Label("StudentName","Student-Name")

- @model Student

@Html.LabelFor(m => m.StudentName)

CheckBox

- HtmlHelper class includes two extension methods to generate a <input type="checkbox"> element in razor view: CheckBox() and CheckBoxFor().
- Ex:

```
@Html.CheckBox("isNewlyEnrolled", true)

@model Student

@Html.CheckBoxFor(m => m.isNewlyEnrolled)
```

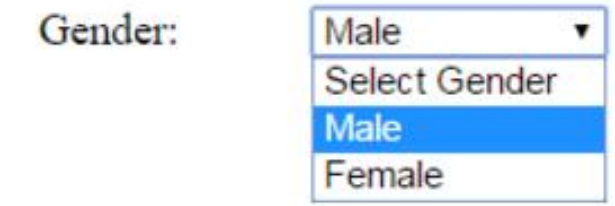
RadioButton

- HtmlHelper class include two extension methods to generate a <input type="radio"> element in a razor view: RadioButton() and RadioButtonFor().
- Ex:

```
Male: @Html.RadioButton("Gender","Male")

Female: @Html.RadioButton("Gender","Female")

@model Student

@Html.RadioButtonFor(m => m.Gender,"Male")

@Html.RadioButtonFor(m => m.Gender,"Female")
```

DropDownList and ListBox

- HtmlHelper class includes two extension methods to generate a <select> element in a razor view: DropDownList() and DropDownListFor().
- DropDownList allows single item selection.
- ListBox allows for multiple item selection
- A select element serves two purposes:
  - To show a list of possible options
  - To show the current value for a field



DropDownList and ListBox

- Ex: DropDownList

```
@model Student
@Html.DropDownList("StudentGender", new SelectList(Enum.GetValues(typeof(Gender))), "Select Gender", new { @class = "form-control" })
```

- Ex: DropDownListFor

```
@model Student
@Html.DropDownListFor(m => m.StudentGender, new SelectList(Enum.GetValues(typeof(Gender))), "Select Gender")
```

Display

- HtmlHelper class includes two extension methods to generate html string : Display() and DisplayFor().
- Ex:
- @Html.Display("StudentName")
- @model Student

  @Html.DisplayFor(m => m.StudentName)

Editor

- **ASP.NET MVC also includes a method that generates html input elements based on the datatype. Editor() or EditorFor() extension method generates html elements based on the data type of the model object's property.**

Property DataType	Html Element
string	<input type="text" >
int	<input type="number" >
decimal, float	<input type="text" >
boolean	<input type="checkbox" >
Enum	<input type="text" >
DateTime	<input type="datetime" >

Rendering Helpers

- The ActionLink method renders a hyperlink (anchor tag) to another controller action.

@Html.ActionLink("Link Text", "AnotherAction")

- When you need a link pointing to an action of a different controller, you can specify the controller name as a third argument to ActionLink.

@Html.ActionLink("Link Text", "AnotherAction“,”AnotherController”)

Rendering Helpers

- The RouteLink helper follows the same pattern as the ActionLink helper, but also accepts a route name and does not have arguments for controller name and action name.
- Ex:

```
@Html.RouteLink("Link Text", new {action="AnotherAction"})
```

- The URL helpers are similar to the HTML ActionLink and RouteLink helpers, but instead of returning HTML they build URLs and return the URLs as strings. There are three helpers:
  - Action
  - Content
  - RouteUrl