HTML Helpers

Types of HTML Helpers

- Inline HTML Helpers
- Built-In HTML Helpers
  - Standard
  - Strongly Typed
  - Templated
- Custom HTML Helpers

Inline HTML Helpers

- They are created using the @helper syntax in Razor

```
@helper ListingItems(string[] items)
{
<ol>
@foreach (string item in items)
{
<li>@item</li>
}
</ol>
}
```

Standard HTML Helpers

BeginForm

@using (Html.BeginForm("Manage", "Account")) {…

- Used to create a Form
- When used without parameters it represents the post action

ValidationSummary

@Html.ValidationSummary()

- Provides an unordered list of validation errors in the page

Standard HTML Helpers

TextBox

```
@Html.TextBox("Textbox1", "val")
Output: <input id="Textbox1" name="Textbox1" type="text" value="val" />
```

TextArea

```
@Html.TextArea("Textarea1", "val", 5, 15, null)
Output: <textarea cols="15" id="Textarea1" name="Textarea1"
rows="5">val</textarea>
```

Password

```
@Html.Password("Password1", "val")
Output: <input id="Password1" name="Password1" type="password"
value="val" />
```

Standard HTML Helpers

Hidden Field

```
@Html.Hidden("Hidden1", "val")
Output: <input id="Hidden1" name="Hidden1" type="hidden" value="val" />
```

CheckBox

```
@Html.CheckBox("Checkbox1", false)
Output:< input id="Checkbox1" name="Checkbox1" type="checkbox"
value="true" />< input name="myCheckbox" type="hidden" value="false" />
```

RadioButton

```
@Html.RadioButton("Radiobutton1", "val", true)
Output: <input checked="checked" id="Radiobutton1" name="Radiobutton1"
type="radio" value="val" />
```

Standard HTML Helpers

Drop-down list

@Html.DropDownList (“DropDownList1”, new SelectList(new [] {"Male", "Female"}))
Output:< select id="DropDownList1" name="DropDownList1">< option>M</option>< option>F</option>< /select>

Multiple-select

Html.ListBox(“ListBox1”, new MultiSelectList(new [] {"Cricket", "Chess"}))
Output:< select id="ListBox1" multiple="multiple" name="ListBox1">

< option>Cricket</option>< option>Chess</option>< /select>

Strongly Typed HTML Helpers

- These helpers are created based on Model properties
- They can work on lambda expressions
- The model object is passed as an object from which are selected fields or properties to be set to id, name and value attributes

Strongly Typed HTML Helpers

TextBox

```
@Html.TextBoxFor(m=>m.Name)
Output: <input id="Name" name="Name" type="text" value="Name-val" />
```

TextArea

```
@Html.TextArea(m=>m.Address , 5, 15, new{}))
Output: <textarea cols="15" id="Address" name=" Address "
rows="5">Addressvalue</textarea>
```

Password

```
@Html.PasswordFor(m=>m.Password)
Output: <input id="Password" name="Password" type="password"/>
```

Strongly Typed HTML Helpers

Hidden Field

@Html.HiddenFor(m=>m.UserId)
Output:

<input id=" UserId" name=" UserId" type="hidden" value="UserId-val" />

CheckBox

@Html.CheckBoxFor(m=>m.IsApproved)
Output:< input id="Checkbox1" name="Checkbox1" type="checkbox" value="true" />< input name="myCheckbox" type="hidden" value="false" />

Strongly Typed HTML Helpers

RadioButton

@Html.RadioButtonFor(m=>m.IsApproved, "val")

Output: <input checked="checked" id="Radiobutton1" name="Radiobutton1" type="radio" value="val" />

Drop-down list

@Html.DropDownListFor(m => m.Gender, new SelectList(new [] {"Male", "Female"}))
Output:< select id="Gender" name="Gender">< option>Male</option>< option>Female</option>< /select>

Strongly Typed HTML Helpers

Multiple-select

Html.ListBoxFor(m => m.Hobbies, new MultiSelectList(new [] {"Cricket", "Chess"}))
Output:< select id="Hobbies" multiple="multiple" name="Hobbies">< option>Cricket</option>< option>Chess</option>< /select>

Templated HTML Helpers

- These helpers figure out the HTML elements needed based on the properties of the model
- The model properties need to be set up with DataTypes and DataAnnotations for this helper to work effectively

Templated HTML Helpers

Display

Renders a read-only view of the specified model property and selects an appropriate HTML element based on property’s data type and metadata.
Html.Display("Name")

DisplayFor

Strongly typed version of the previous helper
Html.DisplayFor(m => m. Name)

Templated HTML Helpers

Editor

Renders an editor for the specified model property and selects an appropriate HTML element based on property’s data type and metadata.
Html.Editor("Name")

EditorFor

Strongly typed version of the previous helper
Html.EditorFor(m => m. Name)

Custom Html Helpers

- Create your own custom helper methods by creating an extension method to the HtmlHelper class