HTML HELPER CLASSES IN ASP.NET MVC

We can build an ASP.NET MVC application without using them, but HTML Helpers helps in the rapid development of a view. HTML Helpers are more lightweight as compared to ASP.NET Web Form controls as they do not use ViewState and do not have event models.

In ASP.Net web forms, developers are using the toolbox for adding controls on any particular page. However, in ASP.NET MVC application there is no toolbox available to drag and drop HTML controls on the view. In ASP.NET MVC application, if you want to create a view it should contain HTML code. So those developers who are new to MVC especially with web forms background finds this a little hard.

To overcome this problem, ASP.NET MVC provides HtmlHelper class which contains different methods that help you create HTML controls programmatically

HTML Helpers are categorized into three types:

1. Inline HTML Helpers
2. Built-in HTML Helpers
3. Custom HTML Helpers

There are different types of helper methods.

* **Createinputs** − Creates inputs for text boxes and buttons.
* **Createlinks** − Creates links that are based on information from the routing tables.
* **Createforms** − Create form tags that can post back to our action, or to post back to an action on a different controller.

The following is the list of Html Helper controls.

* Html.Beginform
* Html.EndForm
* Html.Label
* Html.TextBox
* Html.TextArea
* Html.Password
* Html.DropDownList
* Html.CheckBox
* Html.RedioButton
* Html.ListBox
* Html.Hidden

Below are Strongly Type Html Helper methods, this will allow us to check compile time errors. We get Model's Property intelligence at Runtime.

* Html.LabelFor
* Html.TextBoxFor
* Html.TextAreaFor
* Html.DropDownListFor
* Html.CheckBoxFor
* Html.RadioButtonFor
* Html.ListBoxFor
* Html.HiddenFor

Example of Using HTML HELPER

@using (Html.BeginForm("Index", "Home", FormMethod.Post))

{

<br />

@Html.Label("Enter Emp Name")

@Html.TextBox("EmpName")

<br />

@Html.Label("Enter Emp Designation")

@Html.TextBox("EmpDesig")

<br />

@Html.Label("Enter Employee Email Address")

@Html.TextBox("EmpEmail")

<br />

}

@Html.Label("Enter Name")

@Html.TextBox("txt1")

<br />

@Html.TextBox("txt2")

<br />

ColdDrink

@Html.CheckBox("Colddrink", false)

HTML HELPER WITH CSS and other attribute and property

Example No 1

@Html.TextBox("txtbox","",new{

style="font-size:24px; background-color:lightgreen"

})

* Here new is behaving like the anonymous function

Example No 2

@Html.TextBox("txtbox","",new{

style="font-size:24px; background-color:lightgreen",

title="Enter Company information",

placeholder="Fill these field"

})

* The short hint is displayed in the input field before the user enters a value.
* Title is the hint which display when the user hold the curser

Example 3 Third How to call the class in HTML HELPER

@Html.TextBox("txt3","",new{

@class="format"

})

TextArea Using HTML HELPER

@Html.TextArea("info","",10,20,null)

HTML Element

Example

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" />

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" />

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>