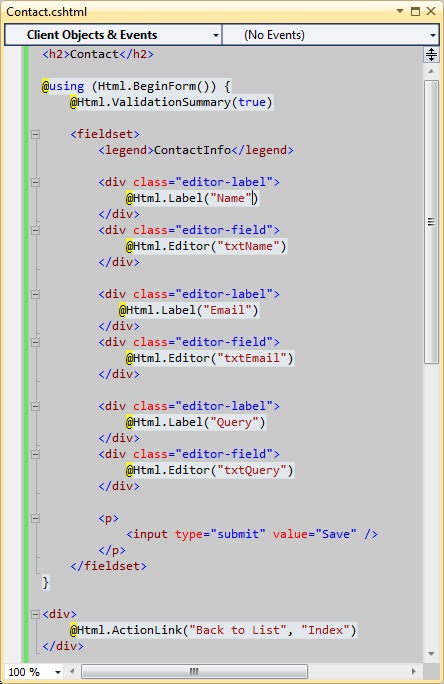
# HTML Helpers and Creating Custom HTML Helpers in ASP.NET MVC

People coming from the asp.net web forms background are used to putting the ASP.NET server control on the page using the toolbox. When we work with ASP.NET MVC application there is no toolbox available to us from where we can drag and drop HTML controls on the view. In MVC, if we want to create a view it should contain HTML code for specifying the mark up. MVC Beginners(specially with Web forms background) finds this a little troubling.

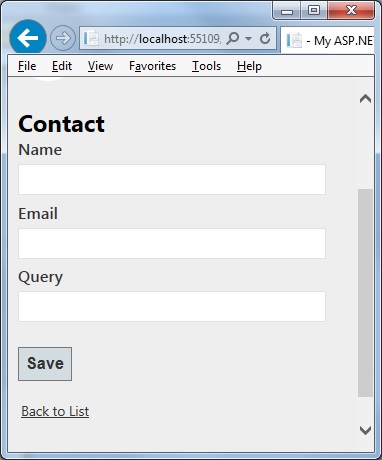
ASP.NET MVC team must have anticipated this problem and thus to ease this problem, the ASP.NET MVC framework comes with a set of HTML Helper methods. These helpers are simple functions that let the developer to specify the type of HTML needed on the view. This is done in C#. The final HTML will be generated by these functions at the runtime i.e. We don't have to worry about the correctness of generated HTML.

**Built in HTML Helpers**

Let’s look at how we can use various HTML helper methods. Lets try to create a simple contact us form that will ask the user for his name, email id and his query. we can design this form in simple HTML easily, let us see how we can utilize HTML helper to achieve the same.



In the above screenshot we can see that the HTML form is created using a HTML helper function, all the labels on the page are created using helper functions and all the textboxes are also created using helper functions. Now if we run the application and try to see the result:



If we want to control the HTML attributes of the generated HTML then we can use the overloaded version of the helper functions. Lets say in the above cove, we want to provide ID to all the labels.



If we run the application again and try to look at the generated HTML, we can see the ID values we have specified.



Following HTML helpers are built into the ASP.NET MVC framework:

* Html.BeginForm
* Html.EndForm
* Html.TextBox
* Html.TextArea
* Html.Password
* Html.Hidden
* Html.CheckBox
* Html.RadioButton
* Html.DropDownList
* Html.ListBox

**HTML Helpers for strongly typed views**

If we are creating a strongly typed view then it is also possible to use the HTML helpers methods with the model class. Let us create a model for the contact us page:

Hide   Copy Code

public class ContactInfo

{

public string Name { get; set; }

public string Email { get; set; }

public string Query { get; set; }

}

Now let is create a strongly typed view for contact us page using the Html helpers.



These helper methods create the output HTML elements based on model properties. The property to be used to create the HTML is passed to the method as a lambda expression. It could also be possible to specify id, name and various other HTML attributes using these helper methods. Following HTML helpers are available to be used with strongly typed views:

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

**Creating Custom HTML Helpers**

If we want to create our own HTML helpers than that can also be done very easily. There are quite a few ways of creating custom helpers. lets look at all the possible methods.

1. Creating a static method
2. Writing an extension method
3. Using the @helper(razor only)

Let us try to create a simple HTML helper method which will create a marked HTML label. There is a new mark tag in HTML5 specifications. Let us try to create a label which create a label then mark it using the HTML5 mark tag.

Let’s try to achieve this using all the above mentioned methods.

**Creating a static method**

In this approach we can simply create a static class with static method which will return the HTML string to be used at the time of rendering.

namespace HTMLHelpersDemo.Helpers

{

public static class MyHTMLHelpers

{

public static IHtmlString LabelWithMark(string content)

{

string htmlString = String.Format("<label><mark>{0}</mark></label>", content);

return new HtmlString(htmlString);

}

}

}

**Writing an extension method**

In this approach we will write a simple extension method for the built in html helper class. this will enable us to use the same Html object to use our custom helper method.

public static class MyExtensionMethods

{

public static IHtmlString LabelWithMark(this HtmlHelper helper, string content)

{

string htmlString = String.Format("<label><mark>{0}</mark></label>", content);

return new HtmlString(htmlString);

}

}

**Using the @helper(razor only)**

This method is pretty specific to razor view engine. Let us see how this can be done. We need to write this method in the view itself.

Hide   Copy Code

@helper LabelWithMarkRazor(string content)

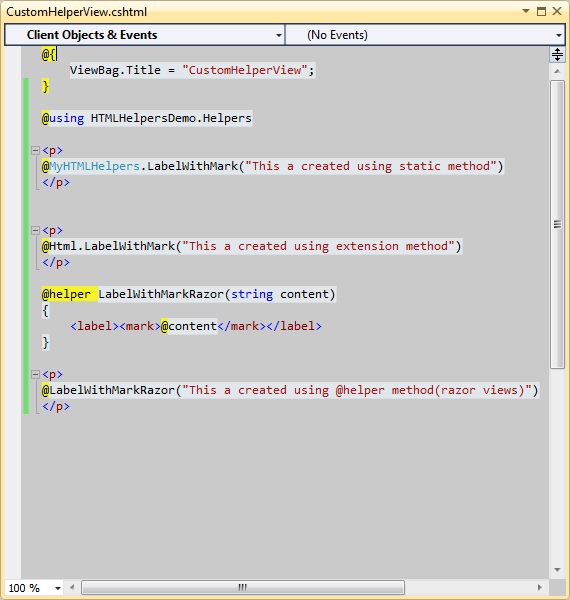
{

<label><mark>@content</mark></label>

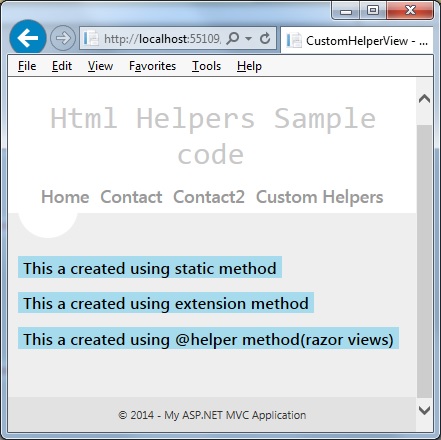
}

**Note:** By default these helpers will be available in the view they are defined in. If we want to use the razor created helpers in multiple views then we have to put all of them in one view and put that view file in App\_Code directory. This way they will be usable from all the views.

Now we have the same html helper method written in 3 different ways, Let us create a simple view and try to use these helper methods.



Let’s run the application and see the result:



In this article we looked at HTML helpers. We have also looked at how we can create custom HTML helpers.

**Written by** [**Rahul Rajat Singh**](http://www.codeproject.com/Members/rahul-rajat-singh)