**Creating Custom HTML Helpers**

이 튜토리얼의 목적은 MVC views에서 사용할 수 있는 사용자 정의 HTML Helpers을 어떻게 만드는 지 설명하는 것입니다. HTML Helpers의 장점으로, HTML 태그의 코드 양을 감소할 수 있습니다.

이 튜토리얼의 첫 파트에서, ASP.NET MVC 프레임워크에 내장된 HTML Helpers에 대해 알아 볼 것이고, 다음으로 사용자 정의 HTML Helpers을 만드는 두 가지 메소드를 설명할 것입니다.

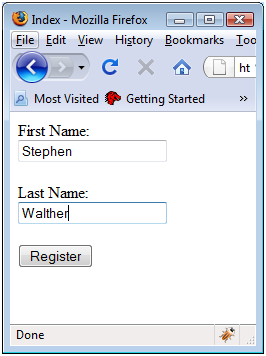
**Understanding HTML Helpers**

HTML Helper은 단순한 string을 리턴 하는 메소드 입니다. String은 사용자가 원하는 여러 타입의 콘텐트를 나타냅니다. 예를 들어, HTML <input>과 <img> 태그와 같이 표준 HTML 태그를 렌더링 할 때 HTML Helpers을 사용할 수 있습니다. 또, 데이터베이스 데이터로부터 HTML table을 만드는 경우와 같이 복잡한 콘텐트를 렌더링 하기 위해서도 사용할 수 있습니다.

ASP.NET MVC 프레임워크는 다음 표준 HTML Helpers을 포함하고 있습니다.

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

예를 들어, 아래의 폼을 생각해 봅시다. 이 폼은 두 개의 표준 HTML Helpers로 렌더링 되었습니다.

[](http://static.asp.net/asp.net/images/mvc/09/Images/mvc_tutorial09_vb_figure01.png)

|  |
| --- |
| <%@ Page Language="C#" AutoEventWireup="true"  CodeBehind="Index.aspx.cs" Inherits="MvcApplication1.Views.Home.Index"%>  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  <html xmlns= "http://www.w3.org/1999/xhtml ">  <head id="Head1" runat="server">  <title>Index</title>  </head>  <body>  <div>  <% using (Html.BeginForm())  { %>  <label for="firstName">First Name:</label>  <br />  <%= Html.TextBox("firstName")%>  <br /><br />  <label for="lastName">Last Name:</label>  <br />  <%= Html.TextBox("lastName")%>  <br /><br />  <input type="submit" value="Register" />  <% } %>  1 </div>  </body>  </html> |

Html.BeginForm() Helper 메소드는 HTML <form> 태그의 시작과 끝을 만드는데 사용된다. Html.BeginForm() 메소드는 using 문안에서 호출 되어 진다는 것을 명심해야 합니다. Using 문은 <form> 태그가 using 블록의 끝에서 닫히게 되는 것을 보장해줍니다.

Using 블록을 만드는 것 대신에, <form> 태그를 닫기 위해 Html.EndForm() Helper 메소드를 호출 할 수도 있습니다. 사용자가 편한 방법으로 <form> 태그를 닫고 여는 방법을 선택할 수 있습니다.

위 코드에서 Html.TextBox() Helper 메소드는 Html <input> 태그를 렌더링 합니다. 웹 브라우저에서 소스 보기를 한다면 아래와 같은 코드를 볼 것입니다.

중요! Html.TextBox() Html Helper은 <$ $> 태그가 아니라 <$= $> 태그 내에서 렌더링 됩니다. ‘=’ 기호가 포함되어 있지 않다면, 브라우저에서 렌더링 되지 않을 것입니다.

|  |
| --- |
| <%@ Page Language="C#" AutoEventWireup="false" CodeBehind="Index.aspx.cs"  Inherits="MvcApplication1.Index" %>  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  <html xmlns="http://www.w3.org/1999/xhtml">  <head>  <title>Index</title>  </head>  <body>  <div>  <form action="http://localhost:33102/" method="post">  <label for="firstName">First Name:</label>  <br />  <input id="firstName" name="firstName" type="text" value="" />  <br /><br />  <label for="lastName">Last Name:</label>  <br />  <input id="lastName" name="lastName" type="text" value="" />  <br /><br />  <input id="btnRegister" name="btnRegister" type="submit" value="Register" />  </form>  </div>  </body>  </html> |

ASP.NET MVC 프레임워크의 helper은 소규모 이기에, 대부분 사용자들은 사용자 정의 HTML Helpers 로 확장하기를 원할 것입니다.

**Creating HTML Helpers with Static Methods**

새로운 HTML Helper을 만드는 가장 쉬운 방법은 문자를 리턴 하는 static 메소드를 만드는 것입니다. 예를 들어, HTML <label> 태그를 렌더링 하는 새로운 HTML Helper을 만든다고 생각해 봅시다. 사용자는 <label>을 렌더링 하기 위한 아래 코드의 클래스를 사용할 수 있습니다.

|  |
| --- |
| using System;  namespace MvcApplication1.Helpers  {  public class LabelHelper  {  public static string Label(string target, string text)  {  return String.Format("<label for='{0}'>{1}</label>", target, text);  }  }  } |

위 코드에서 클래스에 관해 특별한 것이 없습니다. Label() 메소드는 단지 문자를 리턴 합니다.

아래 코드에서 수정된 Index view은 HTML <label> 태그를 렌더링 하기 위해 LabelHelper을 사용합니다.

주의! <%@ Import Namespace="MvcApplication1.Helpers" %>

|  |
| --- |
| <%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Index2.aspx.cs"  Inherits="MvcApplication1.Views.Home.Index2"%>  <%@ Import Namespace="MvcApplication1.Helpers" %>  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  <html xmlns="http://www.w3.org/1999/xhtml" >  <head id="Head1" runat="server">  <title>Index2</title>  </head>  <body>  <div>  <% using (Html.BeginForm())  { %>  <%= LabelHelper.Label("firstName", "First Name:") %>  <br />  <%= Html.TextBox("firstName")%>  <br /><br />  <%= LabelHelper.Label("lastName", "Last Name:") %>  <br />  <%= Html.TextBox("lastName")%>  <br /><br />  <input type="submit" value="Register" />  <% } %>  </div>  </body>  </html> |

**Creating HTML Helpers with Extension Methods**

사용자가 확장 메소드를 만들 필요가 생길 때, ASP.NET MVC 프레임워크에 포함된 표준 HTML Helpers 처럼 동작하는 HTML Helpers을 만드 -----------

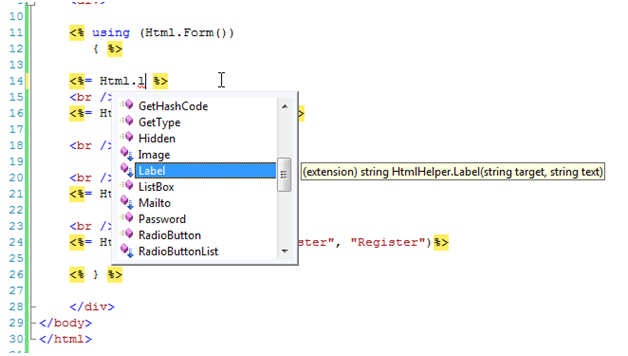
확장 메소드는 존재하는 클래스에 새로운 메소드를 추가하여 활성화 할 수 있습니다. HTML Helper 메소드를 만들 때, view의 Html 프로퍼티를 통해 HtmlHelper 클래스에 새로운 메소드를 추가합니다.

아래 코드의 클래스는 Label() 이라는 이름의 HtmlHelper 클래스에 확장 메소드를 추가합니다. 이 클래스에 관해 주의해야 할 사항이 있습니다. 첫 번째는 이 클래스는 static 클래스이기에 사용자는 static 클래스 내에서 확장 메소드를 정의 해야 합니다.

두 번째로 Label() 메소드의 첫 번째 파라메터는 this 키워드에 의해 선행되어 짐을 알아야 합니다. 확장 메소드의 첫 번째 파라메터는 새로운 확장 메소드가 들어가는 클래스를 의미 합니다.

|  |
| --- |
| using System;  using System.Web.Mvc;  namespace MvcApplication1.Helpers  {  public static class LabelExtensions  {  public static string Label(this HtmlHelper helper, string target, string text)  {  return String.Format("<label for='{0}'>{1}</label>", target, text);  }  }  } |

사용자는 확장 메소드를 만들고 성공적으로 어플리케이션을 빌드 했으면, 확장 메소드는 아래와 같이 Visual Studio Intellisense 에 나타날 것입니다.



아래 코드에서 수정된 Index view은 <label> 태그를 렌더링 할 Html.Label() 확장 메소드를 사용합니다.

|  |
| --- |
| <%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Index3.aspx.cs" Inherits="MvcApplication1.Views.Home.Index3" %>  <%@ Import Namespace="MvcApplication1.Helpers" %>  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  <html xmlns="http://www.w3.org/1999/xhtml" >  <head id="Head1" runat="server">  <title>Index3</title>  </head>  <body>  <div>  <% using (Html.BeginForm())  { %>  <%= Html.Label("firstName", "First Name:") %>  <br />  <%= Html.TextBox("firstName")%>  <br /><br />  <%= Html.Label("lastName", "Last Name:") %>  <br />  <%= Html.TextBox("lastName")%>  <br /><br />  <input type="submit" value="Register" />  <% } %>  </div>  </body>  </html> |