Creating Efficient Markup with Razor Templating

Alex Wolf

www.alexwolfthoughts.com

To-Do List

Understanding Templated Razor Helpers

Demo: Working with Templated HTML Helpers

Demo: Customizing Built-in Templates

Demo: Working with Type Templates

Demo: Building Custom Templates

Demo: Working with Model Templates

Demo: Using Templates with Generated Code

Understanding Templated Razor Helpers

The Templated Rendering Process

```
public class Client
{
    [Display(Name = “Last Name”)]
    public string LastName { get; set;}

    [DataType(DataType.Date)]
    public string Date { get; set;}

    [UIHint(“MultilineTextLarge”)]
    public string Notes { get; set;}

    // Other Properties
}
```

- Friendly display name
- Additional info about the data type
- Specific template to use

Templated HTML Helpers

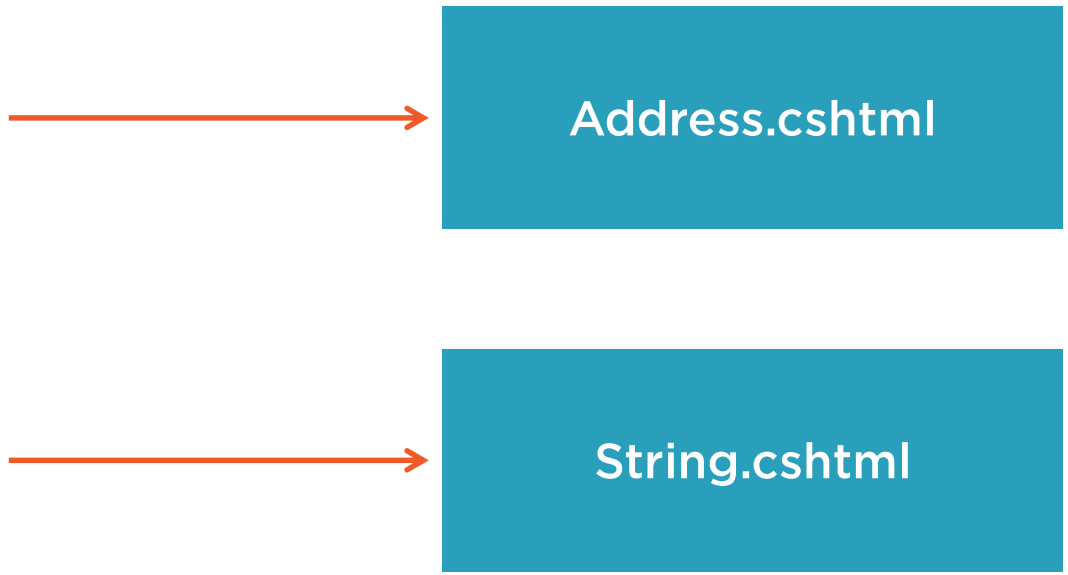
Helper	Output
Html.EditorFor	Renders properties using templates and Metadata

Benefits of Razor Templating

- Reduced Markup
- Configuration Through Metadata
- Powerful Reusability
- Alternative Template Options

Type Templates

```
<div class="widget">
        <div class="form-group">
        @Html.EditorFor(x => x.Address)
        </div>
        <div class="form-group">
        @Html.EditorFor(x => x.LastName)
        </div>
</div>
```



Demo

Working with templated HTML Helpers

Applying additional Model Metadata

Demo

Overriding the built-in Editor Templates

Demo

Working with Custom Type Templates

Demo

Creating Custom Editor Templates

Demo

Using templates and metadata with auto generated code

Summary

Model metadata can influence how properties are rendered using Templated Razor Helpers

Razor allows us to override various types of display templates using naming conventions and metadata

Razor supports custom templates to help with specific situations

Some Templated Helpers can even scaffold an entire model with a single method, drastically reducing markup

Thank You,
and Best of Luck!