ASP.NET Core MVC HTML Helper

Ingo Köster

Diplom Informatiker (FH)

HTML Helper

- Hilfsmethoden zum Erzeugen von HTML aus Modell, ViewBag, ViewData, etc.
- Realisiert durch eine Klasse im Namespace `System.Web.Mvc`
- Klasse mit statischen Methoden
- Kann über statische Erweiterungsmethoden (Extension Methods) beliebig ergänzt werden

Einige Helper

- `Html.ActionLink()`
- `Html.BeginForm()`
- `Html.CheckBox()`
- `Html.DropDownList()`
- `Html.EndForm()`
- `Html.Hidden()`
- `Html.ListBox()`
- `Html.Password()`
- `Html.RadioButton()`
- `Html.TextArea()`
- `Html.TextBox()`
- `Url.Action()`

Links

- Html.ActionLink()
- Aus

```
@Html.ActionLink("Click here to view photo 1", "Display", new { id = 1 })
```

- Wird

```
<a href="/photo/display/1"> Click here to view photo 1</a>
```

- Url.Action()
- Aus

```
<img alt="This image came from an action" src="@Url.Action("GetImage",
  new { id = 1 })" />
```

- Wird

```
<img alt="This image came from an action" src="/photo/getimage/1" />
```

Links

- `Html.ActionLink`
  - Erzeugt einen Link mit einer Beschriftung und verweist auf eine ActionMethode (z.B. Display) des zuletzt verwendeten Controllers
  - Andere Controller können angegeben werden
  - Der ActionLink aus dem Beispiel

    ```
    @Html.ActionLink("…", "Display", new { id = 1 })
    ```
  - Eignet sich für folgende ActionResult Methode

    ```
    public IActionResult Display(int id) { … }
    ```

- `Url.Action`
  - Generiert einen URL, jedoch ohne das Anchor Tag (<a>)

Model-Eigenschaften

- Html.Display**Name**For()
- Aus

```
@Html.DisplayNameFor(model => model.CreatedDate)
```

- Wird

```
CreatedDate
```

- Html.DisplayFor()
- Aus

```
@Html.DisplayFor(model => model.CreatedDate)
```

- Wird

```
03/12/2012
```

Model-Eigenschaften

- `Html.DisplayNameFor`
  - Ermittelt den Namen einer Eigenschaft des Models
  - Verwendet einen Lambda-Ausdruck
  - Durch Annotierung mittels Display kann der angezeigte Wert geändert werden
    - `[Display(Name = "Create Date")]`
    - `public DateTime CreatedDate { get; set; }`

- `Html.DisplayFor`
  - Zeigt den Inhalt aus dem Model an

Label und Eingabefelder

› Html.LabelFor()

› Aus

```
@Html.LabelFor(model => model.ContactMe)
```

› Wird

```
<label for="ContactMe">ContactMe</label>
```

› Html.EditorFor()

› Aus

```
@Html.EditorFor(model => model.ContactMe)
```

› Wird

```
<input type="checkbox" name="ContactMe">
```

Label und Eingabefelder

- `Html.LabelFor`
  - Erzeugt Beschriftungsfelder aus Model-Eigenschaften

- `Html.EditorFor`
  - Eingabefelder für Model-Eigenschaften
  - Erzeugt
    - Checkboxen für boolesche Eigenschaften
    - Eingabefelder in allen anderen Fällen

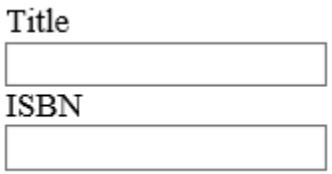
Zusätze

- Kann um weitere Attribute erweitert werden
  - `@Html.EditorFor(m => m.KontoNr, new { htmlAttributes = new { @placeholder = "Kontonummer" } })`

- Neben EditorFor gibt es noch weitere Hilfsmethoden
  - `TextBoxFor`
  - `TextAreaFor`
  - `CheckBoxFor`
  - `RadioButtonFor`
  - `DropDownListFor`
  - `ListBoxFor`

Eingabemaske für eine Model-Klasse

- Mittels des Helpers `EditorForModel` wird für jedes Property der Model-Klasse ein entsprechendes Eingabefeld inklusive Label erzeugt
- Beispiel
  - `@Html.EditorForModel()`
- Erzeugt



Validierung

Html.ValidationSummary()

- Aus
  - @Html.ValidationSummary()
- Wird

```
<ul><li>Please enter your last name</li>
    <li>Please enter a valid email address</li> </ul>
```

Html.ValidationMessageFor()

- Aus

```
@Html.ValidationMessageFor(model => model.Email)
```

- Wird
  - Please enter a valid email address

Validierung

- `Html.ValidationSummary`
  - Zeigt Fehlermeldungen an einer zentralen Stelle an
  - Z.B. für fehlerhafte oder unausgefüllte Formularfelder

- `Html.ValidationMessageFor`
  - Gibt eine Validierungsfehlermeldung für ein einzelnes Feld an

- Werden in Kombination mit den Validierungsfunktionen von MVC verwendet

Formulare

› Html.BeginForm()

› Aus

```
@using (Html.BeginForm("Create", "Photo",
        FormMethod.Post,
        new { enctype = "multipart/form-data" }))
{
    @* Hier werden alle Formular-Elemente eingefügt *@
}
```

› Wird

```
<form action="/Photo/Create" method="post"
    enctype="multipart/form-data">
</form>
```

Versteckte Formularfeld

› `Html.HiddenFor`
  › Erzeugt ein verstecktes Input Feld in einem Formular

› Die Anweisung…

```
@Html.HiddenFor(model => model.Vorname)
```

› …erzeugt das versteckte Input Feld mit Informationen aus dem Model

```
<input id="Vorname" name="Vorname" type="hidden" value="Chong" />
```

Beispiel – ID soll nicht angezeigt werden

› Razor

```
@using (Html.BeginForm()) {
  @Html.HiddenFor(model => model.PersonId)
  @Html.EditorFor(model => model.Vorname)
}
```

› Html

```
<form action="/Home/Edit" method="post">
 <input id="PersonId" name="PersonId" type="hidden" value="0" />
 <input id="Vorname" name="Vorname" type="text" value="Peter" />
</form>
```