

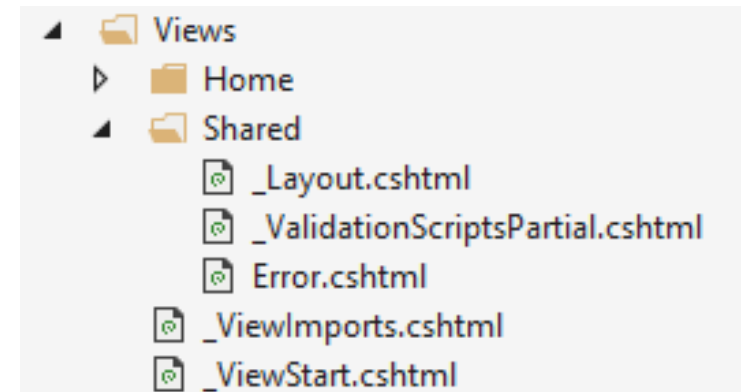
ASP.NET Core MVC Layout

Ingo Köster

Diplom Informatiker (FH)

Layouts in MVC Projekten

- › Für ein einheitliches Aussehen aller Seiten einer ASP.NET MVC Anwendung können Layoutseiten verwendet werden
- › In dem Ordner Views eines MVC-Projektes liegt der Unterordner Shared, welcher die Datei `_Layout.cshtml` enthält



Layoutseite

- › Eine Layoutseite ist ein vollständiges HTML-Dokument, welches Razor-Anweisungen enthalten kann

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    ...
```

```
  </head>
```

```
  <body>
```

```
    @RenderBody()
```

```
  </body>
```

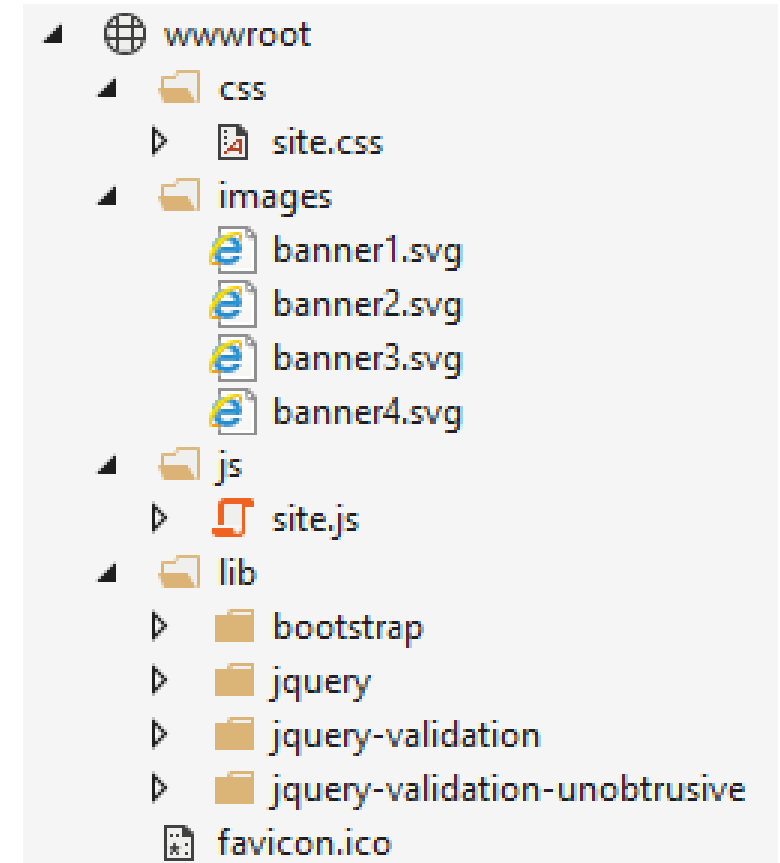
```
</html>
```

Einbinden von Ressourcen

- › Eine Layoutseite erlaubt zudem das Einbinden von Ressourcen wie CSS, JavaScript, jQuery und anderen Frameworks
- › Beispiel CSS
 - › `<link rel="stylesheet" href="~/css/site.css" />`
- › Beispiel jQuery
 - › `<script src="~/lib/jquery/dist/jquery.js"></script>`

Einbinden von Ressourcen

- › Statische Inhalte wie CSS-Dateien, Bilder, etc. müssen im Ordner `~/wwwroot` abgelegt werden
- › In dem Ordner `wwwroot` existieren weitere Unterordner für
 - › Bilder
 - › Eigene Skripte
 - › Frameworks
 - › Z.B. jQuery, etc.



Inhalte der Layoutseite

- › Die Layoutseite
 - › Verwendet Bootstrap
 - › Erstellt ein Layout für Navigation, Inhalt und Footer
 - › Enthält ein Navigationsmenü (mittels Tag Helper)
 - › Enthält einen Platzhalter, an welchem die Inhalte von den einzelnen Views eingebunden werden

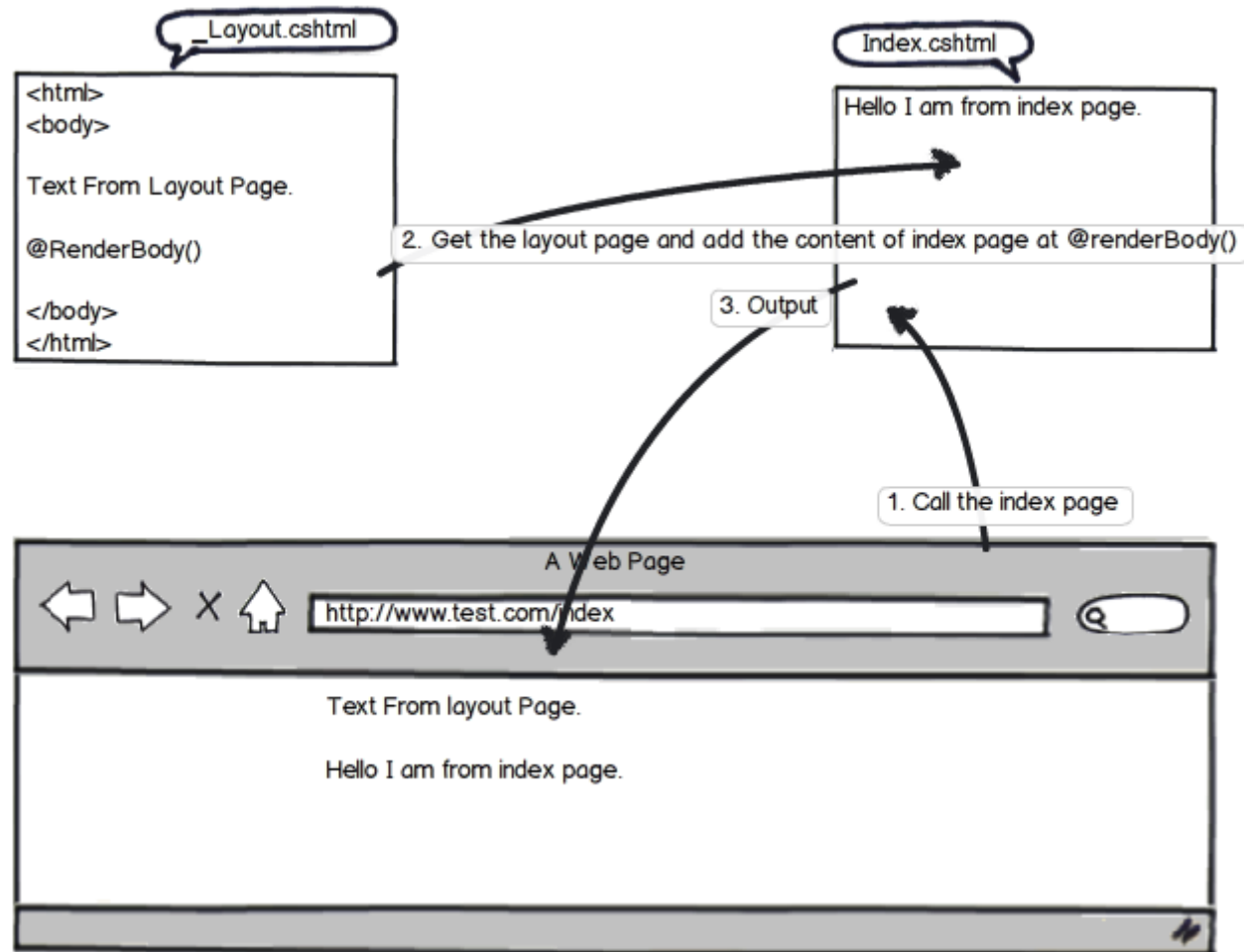
RenderBody

- › Um festzulegen, wo der Inhaltsbereich einer View in der Layoutseite angezeigt werden soll, wird `@RenderBody()` verwendet

```
<!DOCTYPE html>
<html>
<head>...</head>
<body>
  <nav class="navbar navbar-inverse navbar"
  <div class="container body-content">
    @RenderBody()
    <hr />
    <footer>
      <p>&copy; 2018</p>
    </footer>
  </div>
```

- › `RenderBody()` darf in einer Layoutseite nur einmal aufgerufen werden!

RenderBody



Sektionen

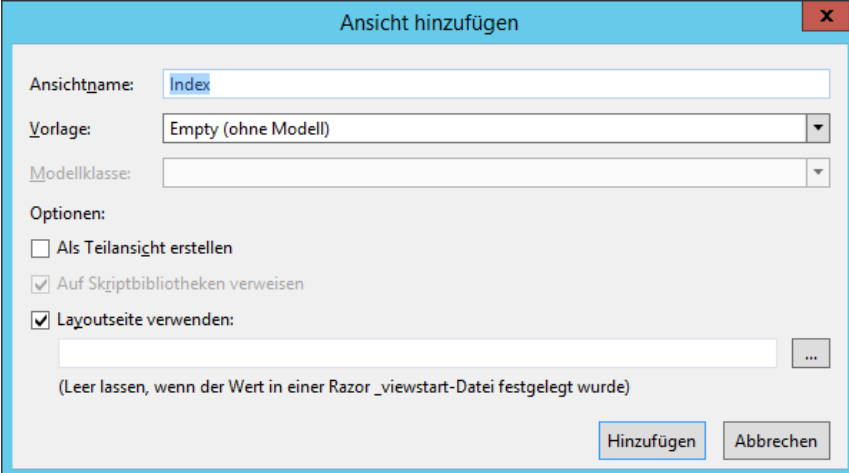
- › Anstelle ganzer Seiten können auch einzelne Bereiche in einer Layoutseite angezeigt werden
- › In der View wird eine Sektion angelegt

```
@section header {  
    <h1>Header Inhalt</h1>  
}
```
- › In der Layoutseite wird der Bereich wie folgt eingebunden

```
@RenderSection("header")
```

Layout einbinden

- › Es gibt verschiedene Möglichkeiten ein Layout in einer Seite einzubinden:
 - › In jeder View
 - › Mittels der `_ViewStart.cshtml` Datei



The screenshot shows the 'Ansicht hinzufügen' (Add View) dialog box. It contains the following fields and options:

- Ansichtname:** A text box containing 'Index'.
- Vorlage:** A dropdown menu showing 'Empty (ohne Modell)'.
- Modellklasse:** An empty dropdown menu.
- Optionen:**
 - ☐ Als Teilansicht erstellen
 - ☒ Auf Skriptbibliotheken verweisen
 - ☒ Layoutseite verwenden:
- A text box for the layout page name, currently empty, with a small '...' button to its right.
- A note below the text box: '(Leer lassen, wenn der Wert in einer Razor _viewstart-Datei festgelegt wurde)'.
- At the bottom right, there are two buttons: 'Hinzufügen' (Add) and 'Abbrechen' (Cancel).

Layout in jeder View angeben

- › Im Kopf jeder View
- › Überschreibt ggf. andere Methoden um Layouts einzubinden
- › Die Layoutseite kann beim Erzeugen der View ausgewählt werden

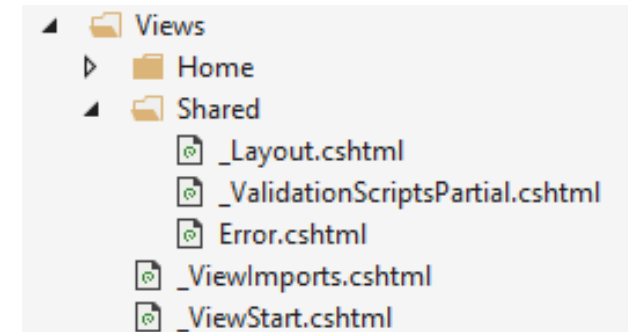
```
@{  
    Layout = "~/Views/Shared/_Layout.cshtml";  
}
```

Layout mittels `_ViewStart` Datei

- › Eine `_ViewStart.cshtml`-Datei in jedem View-Unterordner anlegen
- › Jede View in diesem Ordner verwendet anschließend diese Layoutseite
 - › Falls nicht anders angegeben
- › In der `_ViewStart.cshtml`-Datei kann auch die Route ausgewertet werden, um verschiedene Layout Seiten je nach aufzurufender Seite auszuwählen

Importieren gemeinsam verwendeter Anweisungen

- › Views können Razor-Anweisungen verwenden, um verschiedene Aktionen durchzuführen, u.a.
 - › Importieren von Namespaces
 - › Durchführen von Dependency Injection
- › Anweisungen, die von mehreren Ansichten gemeinsam verwendet werden, können in einer Datei angegeben werden
 - › `_ViewImports.cshtml`
- › Diese liegt im Ordner Views
 - › Enthält bereits das Einbinden des Model-Namespaces



Die _ViewImports-Datei unterstützt folgende Anweisungen (Auswahl)

Anweisung	Bedeutung
@model	Modelbindung
@using	Namensraum einbinden
@inject	Dependency Injection für Views
@inherits	Basisklasse von der die View erben soll
@addTagHelper	Tag-Helper in Views zur Verfügung stellen

Partielle Views

Partielle Views (Teilseiten)

- › Partielle Views werden verwendet wenn:
 - › Daten an verschiedenen Stellen in der Anwendung immer gleich erscheinen sollen
 - › Die Komplexität einer View in mehrere, klar strukturierte Teile aufgegliedert werden soll/muss
- › Namen von partiellen Views sollten immer mit einem Unterstrich beginnen ()
- › Partielle Views teilen sich das übergebene Model mit der Eltern-View
- › Partielle Views teilen sich ViewBag und ViewData mit der Eltern-View und dem Controller

Partielle Views

- › Beispiel (Dateiname: `_NameView.cshtml`)
 `@model string`
 `<h1>@Model</h1>`
- › Einbinden mittels der Helper-Methode `Html.RenderPartial`
 - › 1. Parameter: Name der partiellen View
 - › 2. Parameter: Ein Wert, welcher in der partiellen View als Modell dienen soll
 - › Ist optional, Partielle View und Eltern View teilen sich das Model
- › Beispiel:
 `@{ Html.RenderPartial("_NameView", model: "Tom"); }`
- › Alternativ den Tag Helper `<partial name="_NameView">` verwenden

Partielle View als Ergebnis einer Action Methode

- › Eine partielle View kann als Ergebnis einer ActionMethode zurückgeliefert werden
- › Die Methode PartialView erwartet den Namen der partiellen View sowie ggf. Model-Daten

- › Beispiel

```
public IActionResult Show() {  
    ...  
    return PartialView("_einePartielleView", dieDaten);  
}
```