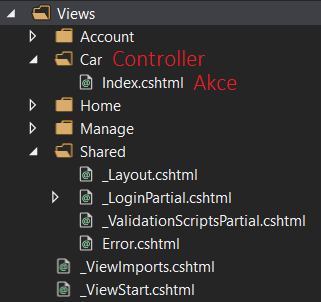
# Razor View Engine

Razor View Engine je renderovací engine vytvořený společností Microsoft. Jedná se o ASP .NET komunitou nejvíce používaný renderovací engine. Umožnuje vytváření dynamických stránek pomocí kombinace HTML značkovacího jazyka a Razor Syntaxe, která nám umožňuje vkládat C# kód do HTML. Soubor s Razor šablonou má koncovku *„.cshtml“*.

## Využití Razor View Engine v MVC projektu

### Kde se šablony nacházejí?



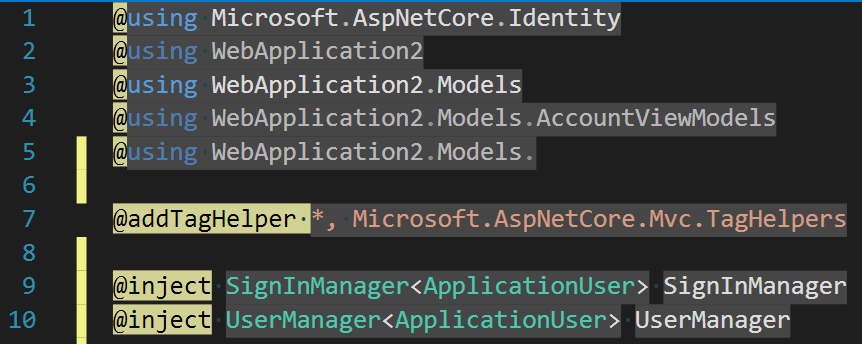
Jednotlivé šablony se nacházejí ve složce „Views“. Struktura této složky musí následovat konvenci určenou MVC návrhem. Každá podsložka nese stejný název jako jeden z kontrolerů a šablony v ní jsou určeny pro jednotlivé akce tohoto kontroleru. Výjimkou je pouze složka Shared, která obsahuje sdílené šablony.

### Složka Shared

Tato složka obsahuje sdílené šablony. Můžeme v ní najít např. *\_Layout.cshtml* nebo částečná zobrazení, tzv. *„partial views*“. Typicky jsou v této složce převážně šablony, které se využívají větším množstvím jiných šablon. Kód šablony se tedy objeví pouze na jednou místě v projektu a programátor se tak může jednoduše vyhnout redundantnímu kódu.

### Soubor \_ViewImports.cshtml

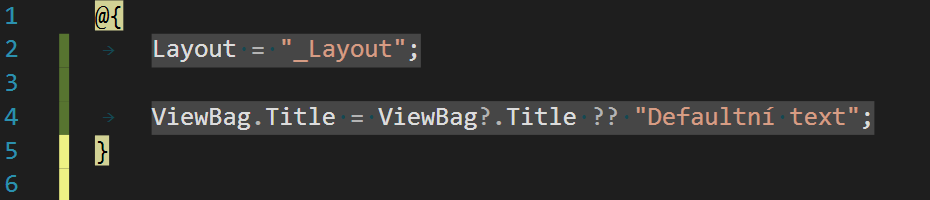
V tomto souboru se importují knihovny, vkládají závislosti a definují *„helper tagy“*, které jsou využívány ve větším množství ostatních šablon. Tento soubor nám tedy umožňuje vyhnout se redundantnímu kódu a mít vše přehledně na jednom místě.



### Soubor ViewStart.cshtml

Kód obsažený v tomto souboru se spustí před tím, než započne renderování šablony. Tento soubor se nachází přímo v kořenové složce Views obsahující všechny šablony. Umístění souboru právě v této složce má svůj důvod. Razor View Engine nejdříve hledá \_ViewStart.cshtml soubor v podsložce související s daným řadičem a pokud tam tento soubor nenajde, začne hledat v kořenové složce.

Typicky se v tomto souboru nachází specifikování Layoutu. Můžeme ho však využít i k jiným účelům. Na následujícím obrázku je demonstrován jeden z nich.



V momentě, kdy je ViewBag.Title specifikován, nenastane žádná změna. Pokud však specifikován není, nastaví se mu výchozí hodnota.

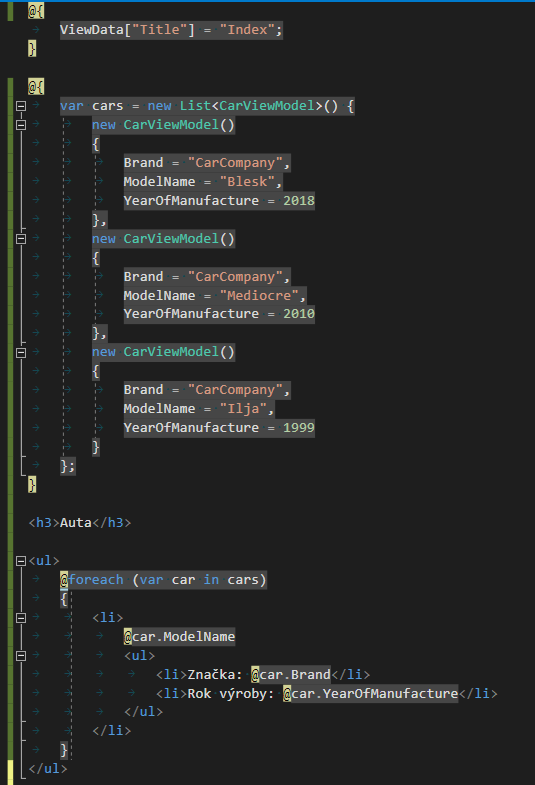
## Razor syntaxe

Hlavní účel Razor šablon je dynamické vytváření obsahu. Toho můžeme docílit použitím Razor syntaxe, která nám umožňuje v šabloně spustit jakýkoliv C# kód. Razor syntaxe se vyznačuje znakem „@“.

### Dělení

* implicitní
  + výraz je pouze dosazen do šablony
  + př. @DateTime.Now
    - dosadí do šablony aktuální čas
* explicitní
  + výraz se vypočítá a až poté se dosadí do šablony
  + př. @(DateTime.Now – TimeSpan.FromHours(1))
    - odečte od aktuálního času 1 hodinu a výsledek dosadí do šablony
* blok kódu
  + slouží k vložení bloku C# kódu
  + vypočítané hodnoty můžeme uložit do proměnných, které poté lze nadále v dané šabloně využívat
  + př. @{ var cislo = 5; } @cislo
    - uloží hodnotu 5 do proměnné číslo a poté dosadí tuto hodnotu do šablony

### Ukázka Razor Syntaxe v praxi





V šabloně je definována kolekce aut, která se renderuje jakožto neorganizovaný list. Přes kolekci aut iterujeme pomocí foreach cyklu a každé auto pak renderujeme jako „list item“.

Správně bychom však položky určené k zobrazení neměli vkládat přímo do šablony, ale raději do databáze. Z těchto záznamů by se pak tvořili modely, které by byly následně šabloně předány.