|  |
| --- |
| ­­­Instruktion |

Räkna med ASP.NET MVC 3

Introduktionsuppgift

|  |
| --- |
| *Författare:* Mats Loock  *Kurs:* ASP.NET MVC  *Kurskod:*1DV409 |

# Innehåll

Du ska följa ”steg-för-steg”-instruktionen i denna introduktionsuppgift och skapa en ASP-NET MVC-applikation som ska addera två heltal en användare matar in i två textfält. Du kommer därefter att modifiera applikationen så att användaren kan välja vilket räknesätt som ska användas.

|  |  |
| --- | --- |
|  | Starta Visual Studio 2010.    Figur . |
|  | Du ska skapa ett nytt projekt för en webbapplikation, välj därför File ► New ► Project....    Figur . |
|  | Dialogrutan New Project visas.    Figur .   1. Under Installed Templates expandera Visual C# och markera Web. 2. Kontrollera så att .NET Framework 4 visas i den nedrullningsbara listrutan. 3. Markera ASP.NET MVC 3 Web Application. 4. Vid Name skriver du in projektets namn (MyValuableCalculator). 5. Ange vid Location en lämplig katalog där projektet ska sparas (C:\Projects). |
|  | Dialogrutan New ASP.NET MVC 3 Project visas.    Figur .   1. Under **Select a Templates** markera **Empty**. 2. Kontrollera så att View engine är satt till Razor. 3. Välj Use HTML5 semantic markup. |
|  | Projektet Visual Studio skapar innehåller filer och kataloger.    /Content  Här placerar du CSS-filer, bilder och övrigt icke dynamiskt innehåll; Javascript placerar i en egen katalog.  /Controllers  Här placerar du Controller-klasser som hanterar URL-begäran som skickas till webbapplikationen.  /Models  Här placerar du klasser som representerar och manipulerar data och affärsobjekt.  /Scripts  Här placerar du JavaScript-bibliotek och -skript (.js).  /Views  Här placerar du vy-filer som ansvar för rendering av t.ex. HTML. |
|  | Högerklicka på katalogen Models och välj Add ► Class….    Figur . |
|  | Dialogrutan Add New Item visas.    Figur .   1. Ange namnet på klassen (ElementaryMath) vid Name. 2. Klicka på Add. |
|  | Klassen ElementaryMath,i namnområdet MyValuableCalculator.Models, ansvarar för att två heltal ska adderas. De två autoimplementerade egenskaperna Op1 och Op2 kommer att innehålla de värden användaren matar in i formulärets textfält. Metoden Compute utför additionen av Op1 och Op2 och summan lagras i ”read-only”-egenskapen Result.    Figur . |
|  | För att Visual Studio ska bli fullt medveten om att en ny klass lagts till projektet bygger du projektet genom att trycka på Skift+F6. Du kan också välja Build ► Build MyValuableCalculator.  C:\Users\mats\AppData\Local\Temp\SNAGHTML2b9bb82c.PNG  Figur . |
|  | Högerklicka på katalogen Controller och välj Add ► Controller….    Figur . |
|  | Dialogrutan Add Controller visas.    Figur .   1. Vid Controller Name skriver du in namnet, HomeController, på kontrollern. ***OBS!*** *Det är mycket viktigt att namnet på kontrollern avslutas med Controller*. 2. Klicka på Add. |
|  | Kontrollern HomeController skapas.    Figur . |
|  | Komplettera metoden Index för att skapa en modell, en instan  s av klassen ElementaryMath, som skickas till vyn.    Figur . |
|  | Högerklicka någonstans i metoden Index och välj Add View….    Figur . |
|  | Dialogrutan Add View visas.    Figur .   1. Det är viktigt att vyn har samma namn som metoden i kontrollern, d.v.s. Index. 2. Kontrollera så att Razor är valt under View engine:. 3. Markera Create a strongly-typed view. 4. Vid Model class: välj klassen ElementaryMath (MyValuableCalculator.Models). 5. Under Scaffold template ska Empty vara valt. 6. Reference script libraries ska vara markerad. 7. Klicka på Add. |
|  | Filen Index.cshtml skapas i katalogen \Views\Home.  Vyer som tillhör kontrollern HomeController måste placeras i katalogen \Views\Home. Det är vanligt att vyernas namn är detsamma som kontrollmetoden som använder dem, men det behöver inte vara så.    Figur .  ***OBS!*** *Vy-filen har filändelsen .cshtml och saknar en ”code behind”-fil; den behövs inte.* |
|  | Med Html-hjälpmetoder skapas ett formulär. Den första raden talar om att vyn är starkt typad och dess modell är av typen ElementaryMath.    Figur . På raderna 9, 11 och 13 refererar m till objektet som utgör modellen. |
|  | Webbapplikationen är nu färdig för att provköras för första gången. Enklast provkör du den genom att trycka på Ctrl+F5.    Figur .  Den renderade HTML-koden innehåller inget du inte skulle skrivit för hand. Textfälten namnges automatiskt med namnet egenskaperna, Op1 respektive Op2, har i klassen ElementaryMath. Någon HTML för egenskapen Result renderas inte eftersom den har värdet null.    Figur . |
|  | För att kunna behandla formulärdatat behöver klassen HomeController kompletteras.    Figur .  Metoden med attributet HttpPost tar hand om URL-begäran av typen POST. Formulärdatat används för att initiera ElementaryMath-objektet parametern elementaryMath refererar till. Egenskaperna Op1 och Op2 kommer att innehålla de värden användaren matat in i respektive textfält i formuläret. |
|  | Provkör webbapplikationen med Ctrl+F5. Skriv in värden 25 och 17. Klicka på kommandoknappen och konstatera att den beräkna summan helt korrekt blir 42.    Figur .  Radera innehållet i textfälten och klicka på kommandoknappen. Konstatera att textfälten nu formateras annorlunda och att summan nu blir 0. Varför? Någon validering är i egentlig mening inte implementerad! Någon beräkning som ger resultatet 0 ska överhuvudtaget inte göras.    Figur . |
|  | För att implementera validering måste klassen ElementaryMath kompletteras så att validering sker med hjälp av ”*data annotations*”.    Figur . |
|  | I kontrollermetoden måste modellens status undersökas. Om objektet uppfyller de krav valideringen ställer ska en beräkning ske och objektet, innehållande resultatet skickas vidare till vyn. Klarar inte objektet valideringen skickas objektet utan beräkning gjord men innehållande felmeddelanden tillbaka till vyn.    Figur . |
|  | Även vyn Home/Index.aspx måste modifieras så valideringsmeddelanden visas.    Figur . |
|  | Nu visas felmeddelande om användaren inte matar in något i textfälten och klickar på kommandoknappen.    Figur .  Felmeddelanden visas även då användaren matar in något som inte kan tolkas som ett heltal.    Figur . |
|  | Nu sker all validering på servern. För att valideringen även ska ske på klienten måste Shared/\_Layout.cshtml kompletteras.    Figur . |
|  | Nu sker valideringen omedelbart när användaren skriver i textfälten.    Figur .  Klickar användaren på kommandoknappen postas inte formuläret men felmeddelanden visas ändå.    Figur . |
|  | Som service till användaren av webbapplikationen skulle det vara trevligt om det första textfältet har fokus när sidan laddat klart. Det kan ordnas enkelt med jQuery. Vyn Home/Index.aspx behöver modifieras.    Figur .  (Fungerar skriptet verkligen? Det gör det eftersom en funktion som skickas som ett argument till jQuery-konstruktorn binds automatiskt till händelsen ready, varför $(document).ready() inte behöver anges explicit.) |
|  | Nu är webbapplikationen nästan klar. Punkterna som följer visar hur en nedrullningsbar listruta kan användas för att användaren ska kunna välja räknesätt. |
|  | Modellen, d.v.s. klassen ElementaryMath, måste modifieras så att den kan hantera de fyra räknesätten.    Figur . |
|  | Skapa en ny katalog, ViewModels, under projektroten.  I denna katalog placerar du filer med klasser, innehållande vyspecifikt data som inte har med modellen att göra, controllrar kan skapa instanser av för att skicka till vyer.    Figur . |
|  | Lägg till en ny klass med namnet HomeIndexViewModel i katalogen ViewModels.    Figur . |
|  | Klassen ElementaryMathViewModel innehåller två publika egenskaper. Egenskapen ElementaryMath refererar till objektet som utgör själva modellen. ”Read-only”-egenskapen ArithmeticOperations returnerar en lista med de fyra räknesätten som den nedrullningsbara listrutan i vyn ska visa.    Figur . |
|  | Vyn Home/Index.aspx måste modifieras så den använder vymodellklassen HomeIndexViewModel istället för modellklassen ElementaryMath. Dessutom ska +-tecknet ersättas med en nedrullningsbar listruta.  C:\Users\mats\AppData\Local\Temp\SNAGHTML3f6defd7.PNG  Figur .  Lägg märke till att Op1, Op2 och Result nu måste förgås av ElementaryMath, som ju egenskapen i vymodellklassen heter som refererar till modellobjektet av typen ElementaryMath. |
|  | Även metoderna i klassen HomeController måste modifieras att använda klassen HomeIndexViewModel.      Figur .  Det enda som behöver ändras är att objektet som skickas till vyn ska vara av typen HomeIndexViewModel. ASP.NET MVC kan fortfarande binda formulärdatat till objektet av typen ElementaryMath. Detta fungerar eftersom parametern (elementaryMath) har samma namn som egenskapen har i vymodellen (ElementaryMath). |
|  | Nu kan användaren välja om de två talen ska summeras, subtraheras, multipliceras eller divideras.    Figur .  Valideringen fungerar fortfarande trots förändringarna som gjorts. Det är bara att prova…  C:\Users\mats\AppData\Local\Temp\SNAGHTML3fd74816.PNG  Figur . |
|  | Nu är webbapplikationen klar. Förhoppningsvis har du nu fått en introduktion till ASP.NET MVC och förstår grundläggande koncept, även om det är mycket mer att lära. |