Razor-мен жұмыс

Жоба мысалын дайындау

Razor қалай жұмыс істейтінін көрсету үшін мен негізгі веб-қосымшаның жобасын жасадым ASP.NET (. NET Core) Razor деп аталады, алдыңғы тарауда сияқты бос үлгіні қолданады. Мен MVC инфрақұрылымын 5-1 листингінде көрсетілген өзгерістерді Startup класына енгіздім.

Listing 5-1. Razor қалтасындағы Startup.cs файлында MVC қосу

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.DependencyInjection;

namespace Razor
 {
        public class Startup
         {
                public void ConfigureServices(IServiceCollection services)
                {
                services.AddMvc();
                }
                public void Configure(IApplicationBuilder app, IHostingEnvironment env)
                {
                if (env.IsDevelopment())
                        {
                        app.UseDeveloperExceptionPage();
                }
                        //app.Run(async (context) =>
                        {
                        //    await context.Response.WriteAsync("Hello World!");
                        //
                        });
                app.UseMvcWithDefaultRoute();
                }
        }
 }
```

Модельді анықтау

Әрі қарай, мен Models қалтасын жасап, оған Product класының файлын қостым.мен 5-2 Листингте көрсетілген қарапайым модель класын анықтау үшін қолданған cs.

Listing 5-2. Models қалтасындағы Product.cs файлының мазмұны

```
namespace Razor.Models
{
        public class Product
    {
        public int ProductID{ get; set; }
        public string Name { get; set; }
        public string Description { get; set; }
        public decimal Price { get; set; }
        public string Category { set; get; }
    }
}
```

Контроллер құру

Мен Startup файлында орнатқан әдепкі Конфигурация.cs, әдепкі Home деп аталатын контроллерге сұраныстарды жіберу туралы MVC келісіміне сәйкес келеді. Мен Controllers қалтасын жасадым және оған HomeController.cs.деп аталатын сынып файлын қостым.мен 5-3 Листингте көрсетілген қарапайым контроллерді анықтау үшін қолданған

Listing 5-3. контроллер қалтасындағы HomeController.cs файлының мазмұны.

```
using Microsoft.AspNetCore.Mvc;
using Razor.Models;

namespace Razor.Controllers
{
    public class HomeController : Controller
    {
        public ViewResultIndex()
        {
        Product myProduct = new Product
                {
        ProductID = 1,
                    Name = "Kayak",
                    Description = "A boat for one person",
                    Category = "Watersports",
                    Price = 275M
                };
                return View(myProduct);
        }
    }
}
```

Контроллер Index деп аталатын әрекет әдісін анықтайды, онда мен өнім объектісінің қасиеттерін жасаймын және толтырамын. Мен өнімді көрініс әдісіне жеткіземін, сондықтан ол көріністі визуализациялау кезінде модель ретінде қолданылады. Мен View әдісін шақырған кезде көрініс файлының атын көрсетпеймін, сондықтан әрекет әдісі үшін әдепкі көрініс қолданылады.

Көрініс құру

Index әрекет әдісі үшін әдепкі көріністі жасау үшін мен Views / Home қалтасын жасап, оған index деп аталатын ViewMVC cshtml.файлын қостық.5-4 Листингте көрсетілген мазмұнды қосқан

Listing 5-4. The Contents of the Index.cshtml File in the Views/Home Folder

```
@model Razor.Models.Product
@{
   Layout = null;
}
<!DOCTYPE html>
<html>
        <head>
                <meta name="viewport" content="width=device-width" />
                <title>Index</title>
        </head>
        <body>
                Content will go here
        </body>
</html>
```

Келесі бөлімдерде мен Razor презентациясының әртүрлі бөліктерін қарап, онымен жасай алатын әртүрлі нәрселерді көрсетемін. Razor-ді үйрену кезінде қолданушыға модельдің бір немесе бірнеше бөлігін білдіретін көріністер бар екенін есте ұстаған жөн, бұл бір немесе бірнеше нысандардан алынған деректерді көрсететін HTML кодын құруды білдіреді. Егер сіз Мен әрқашан клиентке жіберілетін HTML парағын жасауға тырысқанымды есіңізде сақтасаңыз, онда Razor жасаған барлық нәрсе мағынаны ала бастайды. Егер сіз қосымшаны іске қоссаңыз, 5-1 суретте көрсетілген қарапайым нәтижені көресіз.

Model объектісімен жұмыс

Index ұсыну файлындағы бірінші жолдан бастайық.cshtml:
... @modelRazor.Models.Product ...
Razor өрнектері @символынан басталады. Бұл жағдайда @model өрнегі модель объектісінің түрін жариялайды, мен оны әрекет әдісінен көрініске беремін. Бұл маған 5-5 Листингте көрсетілгендей @Model арқылы көрініс моделі объектісінің әдістеріне, өрістеріне және қасиеттеріне сілтеме жасауға мүмкіндік береді, онда Index көрінісіне қарапайым қосымша көрсетіледі.
Listing 5-5. Index.cshtml файлындағы ViewModel объектісінің сипатына сілтеме Views / Home қалтасындағы cshtml

```
@model Razor.Models.Product
@{
   Layout = null;
 }
<!DOCTYPE html>
<html>
```

```
    <head>
        <meta name="viewport" content="width=device-width" />
        <title>Index</title>
    </head>
    <body>
            @Model.Name
    </body>
</html>
```

■ @model (m кіші әрпі) арқылы көрініс моделінің нысан түрін жариялайтынымды және @Model (m үлкен әрпі) арқылы Name қасиетіне жүгінетінімді ескеріңіз. Сіз Razor-мен жұмыс істей бастағанда бұл аздап шатастырады, бірақ ол тез арада екінші сипатқа ие болады.

Егер сіз қосымшаны іске қоссаңыз, 5-2 суретте көрсетілген нәтижені көресіз.
Түрін көрсету үшін @model өрнегін қолданатын көрініс қатаң терілген көрініс деп аталады. Visual Studio @Model өрнегін 5-3 суретте көрсетілгендей, @Model терген кезде мүшелер аттарын көрсету үшін қолдана алады.
Visual Studio-ның элемент атауларына арналған ұсыныстары Razor көріністерінде қателіктер жібермеуге көмектеседі. Егер сіз қаласаңыз, ұсыныстарды елемеуге болады және Visual Studio әдеттегі с #файлдарындағыдай түзетулер енгізу үшін мүшелер атауларына қатысты мәселелерді шешеді. Сіз проблемаларды бөлектеу мысалын 5-4 суретте көре аласыз, онда мен @ Model-ге сілтеме жасауға тырыстым.NotARealProperty. Visual Studio мен модель түрі үшін көрсеткен өнім класының мұндай қасиеті жоқ екенін түсінді және редактордағы қатені анықтады.