عنوان: **توسعه کنترلر و مدل در F# MVC4**
نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۲/۱۲/۱۷ ۲۳:۵
آدرس: www.dotnettips.info
گروه‌ها: #MVC, F

در پست قبلی با F# MVC4 Template آشنا شدید. در این پست به توسعه کنترلر و مدل در قالب مثال خواهم پرداخت. برای شروع ابتدا یک پروژه همانند مثال ذکر شده در پست قبلی ایجاد کنید. در پروژه C# ساخته شده که صرفا برای مدیریت Viewها است یک View جدید به صورت زیر ایجاد نمایید:

```
@model IEnumerable<FsWeb.Models.Book>
<!DOCTYPE html>
<html>
<head>
    <title>@ViewBag.Title</title>
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet"
        href="http://code.jquery.com/mobile/1.0.1/jquery.mobile-1.0.1.min.css" />
</head>

<body>
    <div data-role="page" data-theme="a" id="booksPage">
        <div data-role="header">
            <h1>Guitars</h1>
        </div>
        <div data-role="content">
        <ul data-role="listview" data-filter="true" data-inset="true">
            @foreach(var x in Model) {
                <li><a href="#">@x.Name</a></li>
            }
        </ul>
        </div>
    </div>
    <script src="http://code.jquery.com/jquery-1.6.4.min.js">
    </script>
    <script src="http://code.jquery.com/mobile/1.0.1/jquery.mobile-1.0.1.min.js">
    </script>

    <script>
        $(document).delegate("#bookPage", 'pageshow', function (event) {
            $("div:jqmData(role='content') > ul").listview('refresh');
        });
    </script>
</body>
</html>
```

از آنجا که هدف از این پست آشنایی با بخش F# پروژه‌های وب است در نتیجه نیازی به توضیح کدهای بالا دیده نمی‌شود.
برای ساخت کنترلر جدید، در پروژه F# ساخته شده یک Source File ایجاد نمایید و کدهای زیر را در آن کپی نمایید:

```
namespace FsWeb.Controllers

open System.Web.Mvc
open FsWeb.Models

[<HandleError>]
type BooksController() =
    inherit Controller()
    member this.Index () =
        seq { yield  Book(Name = "My F# Book")
              yield Book(Name = "My C# Book") }
        |> this.View
```

در کدهای بالا ابتدا یک کنترلر به نام BookController ایجاد کردیم که از کلاس Controller ارث برده است(با استفاده از inherit). سپس یک تابع به نام Index داریم(به عنوان Action مورد نظر در کنترلر) که آرایه ای از کتاب‌ها به عنوان پارامتر به تابع View می‌فرستد.(توسط اپراتور Pipe - |>). در نهایت دستور this.View معادل فراخوانی اکشن ()View در پروژه‌های C# است که View متناظر با اکشن را فراخوانی می‌کند. همان طور که ملاحظه می‌نمایید بسیار شبیه به پیاده سازی C# است.
اما نکته ای که در مثال بالا وجود دارد این است که دو نمونه از نوع Book را برای ساخت seq وهله سازی می‌کند. در نتیجه باید

Book Type را به عنوان مدل تعریف کنیم. به صورت زیر:

```
namespace FsWeb.Models
 type Book = { Id : Guid; Name : string }
```

البته در F# 3.0 امکانی فراهم شده است به نام Auto-Properties که شبیه تعریف خواص در C# است. در نتیجه می‌توان تعریف بالا را به صورت زیر نیز بازنویسی کرد:

```
namespace FsWeb.Models
type Book() = member val Name = "" with get, set
```

Attributeها مدل

اگر همچون پروژه‌های C# قصد دارید با استفاده از Attributeها مدل خود را اعتبارسنجی نمایید می‌توانید به صورت زیر اقدام نمایید:

```
open System.ComponentModel.DataAnnotations
 type Book() = [<Required>] member val Name = "" with get, set
```

هم چنین می‌توان Attributeهای مورد نظر برای مدل EntityFramework را نیز اعمال نمود(نظیر Key):

```
namespace FsWeb.Models
open System
open System.ComponentModel.DataAnnotations
type Book() = [<Key>] member val Id = Guid.NewGuid() with get, set
[<Required>] member val Name = "" with get, set
```

نکته: دستور open معادل با using در C# است. در پست بعدی برای تکمیل مثال جاری، روش طراحی Repository با استفاده از EntityFramework بررسی خواهد شد.

نظرات خوانندگان

نویسنده: vahid
تاریخ: ۱۳۹۲/۱۲/۱۹ ۱۲:۴۳

دلیل استفاده F# در MVC چیست ؟ چه مزایایی برای ما دارد ؟ ممنون

نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۲/۱۲/۱۹ ۱۴:۴۶

در این پست درباره مزایای F# بحث شد. البته در اینجا نیز مزایای F# برای پیاده سازی پروژه‌های وب از زبان طراحان آن بیان شده است.