عنوان: **چندین Submit در یک Html Form و انتساب Action های مجزا به هر یک از Submit ها در MVC**
نویسنده: سید مهران موسوی
تاریخ: ۱۳۹۱/۰۹/۲۲ ۱:۰
آدرس: www.dotnettips.info
برچسب‌ها: C#, ASP.Net MVC

تا به حال به این نکته برخورد کردید که برای یک فرم Html نیاز به چندین Submit داشته باشید که هر کدوم یک Action مجزا داشته باشن و یک کار متفاوت انجام بدن ؟
برای مثال فرمی داریم که داده‌های وارد شده در ان باید به دو صورت برای یک کاربر ارسال بشن یا از طریق پیامک یا از طریق ایمیل (این فقط یک مثال پیش فرض هست) و .... در حالت عادی ما در یک فرم نمیتونیم دو عدد Submit داشته باشیم که هر کدوم به یک Action جدا بسط داده بشه خب راه حل چیه ؟ شاید با خودتون بگید خب دو input از نوع radio قرار میدیم و در یک اکشن کنترل میکنیم که کدوم یکی انتخاب شده و عملیات رو با اون معیار انجام میدیم ... به نظرتون زیباتر نیست برای هر عملیات که ممکن باشه هر کدوم کاملا روال کاری متفاوتی داشته باشه یک Action وجود داشته باشه ؟ در این صورت خوانایی کد خیلی بالاتر میره و Unit Test هر Action کاملا مشخص هست که قراره چه فرایندی رو مورد تست قرار بده و مجبور نیستیم چندین حالت رو با عبارات شرطی از هم جدا کنیم و همه چی قاطی بشه با هم ... من در کل با امکاناتی که C# و MVC در اختیارم قرار میده حاظر نیستم تن به کد نویسی به صورت کلاسیک و قاطی پاتی بدم سعی میکنم با مطالعه‌ی سورس MVC بهترین حالت رو انتخاب کنم شما چطور ؟ معلومه که همه همینو میخوان پس بریم سر اصل مطلب .
قطعه کد Html و Razor ساده‌ی زیر رو در نظر بگیرید برای View :

```
@model Models.MyModel
@{
    Layout = null;
}
<!DOCTYPE html>
<html>
<head>
    <title>ViewPage1</title>
</head>
<body>
    <div>
        @using (Html.BeginForm("SendMessage", "Home", FormMethod.Post))
        {

            @Html.LabelFor(x => x.Name);
            @Html.TextBoxFor(x => x.Name);

            <input type="submit" value="ارسال توسط پیامک" name="Send_sms" />
            <input type="submit" value="ارسال توسط ایمیل" name="Send_email" />
        }
    </div>
</body>
</html>
```

خب ما دو تا Submit داریم . یکم اگه شیطنت کنید و مقادیر ارسال شده بعد از submit این فرم رو توسط ابزارهای مانیتورینگ بررسی کنید میبینید که روی هر کدوم از Submitها که کلیک میشه داده ای با نام اون که در خاصیت name اون و مقدار موجود در value اون همراه اون فرم به سرور ارسال میشه و اون یکی Submit از این اتفاق بی نصیب میمونه ... خب ما هم استفاده‌ی لازم رو از این موضوع شیرین میبریم و با یک تکنیک تهاجمی از این موضوع برای رسیدن به هدفمون استفاده میکنیم .

این هم کلاس Model ماست :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel;

namespace Models
{
    public class MyModel
    {
        [DisplayName("نام خود را وارد کنید :")]
```

```
        public string Name { get; set; }
    }
}
```

و اما یک نکته‌ی دیگه . توجه داشته باشید که ما در قسمت View نام Action رو در فرم, SendMessage مشخص کردیم . ولی ... اصلا در واقع همچین اکشنی وجود نداره ! پس چرا ما همچین نامی رو واسه اکشن فرم گذاشتیم !؟
دلیل اینه که ما قصد داریم با یک ActionNameSelectorAttribute درخواست کاربر رو شکار کنیم و اون رو به اکشن دلخواه ارجاع بدیم ... جالبه نه ؟ ولی چه جوری ... کلاس زیر رو بهش دقت مضاعف کنید و در پروژتون ایجادش کنید :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Reflection;

namespace ActionHandlers
{
    public class SendMessageHandlerAttribute : ActionNameSelectorAttribute
    {
        public override bool IsValidName(ControllerContext controllerContext, string actionName,
MethodInfo methodInfo)
        {
            if (actionName.Equals(methodInfo.Name, StringComparison.InvariantCultureIgnoreCase))
                return true;

            if (!actionName.Equals("SendMessage", StringComparison.InvariantCultureIgnoreCase))
                return false;

            var request = controllerContext.RequestContext.HttpContext.Request;
            return request[methodInfo.Name] != null;
        }
    }
}
```

خب حالا بخش Controller رو بهش دقت کنید که ما در اون دو اکشن رو با نام هایی که برای هر Submit مشخص کردیم مینویسیم و ActionNameSelectorAttribute نوشته شده رو به اونها بسط میدیم.

```
[SendMessageHandler ]
        [HttpPost]
        public ContentResult Send_sms(MyModel mdl)
        {
            /// Do something ...
           return string.Empty ;
        }

        [SendMessageHandler ]
        [HttpPost]
        public ContentResult Send_email(MyModel mdl)
        {
           /// Do something ...
          return  string.Empty;
        }
```

خب حالا بعد از کلیک بر روی هر Submit اکشن متناظر با اون اجرا میشه . بعد از ارسال درخواست به سرور MVC در بین اکشن‌های موجود در Controller مشخص شده به دنبال اکشن معین شده میگرده و وقتی به اکشن‌های ما میرسه میبینه عجب ! اون دوتا ActionNameSelectorAttribute سفارشی دارن پس میره ببینه چه خبره اونجا که ما با یک حرکت تهاجمی بررسی میکنیم که اگه نام اکشن مشخص شده در فرم با نام اکشن در حال بررسی مساوی بود که همینو اجرا کن (یعنی ما میتونی اکشنی با نام SendMessage هم داشته باشیم) . اگه نام اکشن مشخص شده در فرم اون نامی نبود که ما میخوایم که کلا بیخیال هندل کردن اکشن میشیم میزاریم خود MVC تصمیم بگیره . و در اخر بررسی میکنیم که ایا در درخواست جاری مقداری با نام اکشن در حال بررسی وجود داره !؟ اگه داشت یعنی همون Submit که ما میخوایم وصل بشه به این اکشن کلیک شده پس اکشن در حال بررسی رو بسط میدیم به درخواست ارسال شده ... به همین سادگی ...

پیروز و موفق باشید .

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۹/۲۲ ۱۷:۳۱

راه جالبی است. نمی‌دونستم که ActionNameSelectorAttribute وجود خارجی دارد!
چند راه حل دیگر:
الف) استفاده از قابلیت‌های binding . مثلا اگر نام پارامترها را به نام همان دکمه‌های موجود تنظیم کنیم، این نام موقع دریافت از دکمه کلیک شده، نال نیست:

```
[HttpPost]
public ActionResult MyAction([other params here], string buttonName1, string buttonName2, etc)
{
     if(!string.IsNullOrEmtpy(buttonName1)) { //button1 was clicked}
}
```

ب) میشه از ویژگی‌های جدید HTML5 مثل data-form-action استفاده کرد:

```
<input type="submit" value="Standard action">
<input data-form-action="@Url.Action("mysecondaction")" type="submit" value="Second action">
```

بعد موقع ارسال کمی از jQuery استفاده کرد

```
$(document).on('click', '[type="submit"][data-form-action]', function(event) {
    var $this = $(this),
        formAction = $this.attr('data-form-action'),
        $form = $($this.closest('form'));
        $form.attr('action', formAction);
});
```

نویسنده: سید مهران موسوی
تاریخ: ۱۳۹۱/۰۹/۲۲ ۱۸:۱

ممنون دوست عزیز که توجه میکنی به مطالب بنده . به جز روشی که در مقاله بالا ذکر شد استفاده از data-form-action موجود در نگارش 5 از HTML اصولیترین روش یا بهتره بگیم تمیزترین روش در بین مواردی هست که ذکر کردید ولی خب برنامه نویسای وب همیشه با مرورگرهای مختلف در جنگ هستن و فعلا زیاد استفاده از این روش جایز نیست چون بعضی از مرورگرهای کمی قدیمیتر پشتیبانی نمیکنند پس بهتره از تکنیکهای مورد اطمینانتر استفاده کنیم که نمونش در مقاله‌ی بالا ذکر شد ...

این نمونه روش‌های خلاقانه و جالب از جمله ActionNameSelectorAttribute به وفور در نسخه‌های جدید mvc وجود داره خوشبختانه . در مقالات بعدیم سعی میکنم مطالب تخصصی و تکنیکهای جالب دیگرو هم با دوستان به اشتراک بزارم . انصافا mvc یکی از شیرین‌ترین تکنولوژی هایی هست که هر سری سورسش رو مطالعه میکنم راه کارهای جدیدی برای پیاده سازی نرم افزارها توسط اون تکنیک‌ها دستگیرم میشه ...
پاینده باشید و موفق