عنوان: **ASP.NET MVC #21**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۲/۰۵ ۱۷:۴۸:۰۰
آدرس: www.dotnettips.info
گروه‌ها: MVC

آشنایی با تکنیک‌های Ajax در ASP.NET MVC

اهمیت آشنایی با Ajax، ارائه تجربه کاربری بهتری از برنامه‌های وب، به مصرف کنندگان نهایی آن می‌باشد. به این ترتیب می‌توان درخواست‌های غیرهمزمانی (asynchronous) را با فرمت XML یا Json به سرور ارسال کرد و سپس نتیجه نهایی را که حجم آن نسبت به یک صفحه کامل بسیار کمتر است، به کاربر ارائه داد. غیرهمزمان بودن درخواست‌ها سبب می‌شود تا ترد اصلی رابط کاربری برنامه قفل نشده و کاربر در این بین می‌تواند به سایر امور خود بپردازد. به این ترتیب می‌توان برنامه‌های وبی را که شبیه به برنامه‌های دسکتاپ هستند تولید نمود؛ کل صفحه مرتبا به سرور ارسال نمی‌شود، flickering و چشمک زدن صفحه کاهش خواهد یافت (چون نیازی به ترسیم مجدد کل صفحه نخواهد بود و عموما قسمتی جزئی از یک صفحه به روز می‌شود) یا بدون نیاز به ارسال کل صفحه به سرور، به کاربری خواهیم گفت که آیا اطلاعاتی که وارد کرده است معتبر می‌باشد یا نه (نمونه‌ای از آن را در قسمت Remote validation اعتبار سنجی اطلاعات ملاحظه نمودید).

مروری بر محتویات پوشه Scripts یک پروژه جدید ASP.NET MVC در ویژوال استودیو

با ایجاد هر پروژه ASP.NET MVC جدیدی در ویژوال استودیو، یک سری اسکریپت هم به صورت خودکار در پوشه Scripts آن اضافه می‌شوند. تعدادی از این فایل‌ها توسط مایکروسافت پیاده سازی شده‌اند. برای مثال:

MicrosoftAjax.debug.js
MicrosoftAjax.js
MicrosoftMvcAjax.debug.js
MicrosoftMvcAjax.js
MicrosoftMvcValidation.debug.js
MicrosoftMvcValidation.js

این فایل‌ها از ASP.NET MVC 3 به بعد، صرفا جهت سازگاری با نگارش‌های قبلی قرار دارند و استفاده از آن‌ها اختیاری است. بنابراین با خیال راحت آن‌ها را delete کنید! روش توصیه شده جهت پیاده سازی ویژگی‌های Ajax ایی، استفاده از کتابخانه‌های مرتبط با jQuery می‌باشد؛ از این جهت که 100ها افزونه برای کار با آن توسط گروه وسیعی از برنامه نویس‌ها در سراسر دنیا تاکنون تهیه شده است. به علاوه فریم ورک jQuery تنها منحصر به اعمال Ajax ایی نیست و از آن جهت دستکاری DOM (document object model) و CSS صفحه نیز می‌توان استفاده کرد. همچنین حجم کمی نیز داشته، با انواع و اقسام مرورگرها سازگار است و مرتبا هم به روز می‌شود.

در این پوشه سه فایل دیگر پایه کتابخانه jQuery نیز قرار دارند:

jquery-xyz-vsdoc.js
jquery-xyz.js
jquery-xyz.min.js

فایل vsdoc برای ارائه نهایی برنامه طراحی نشده است. هدف از آن ارائه Intellisense بهتری از jQuery در VS.NET می‌باشد. فایلی که باید به کلاینت ارائه شود، فایل min یا فشرده شده آن است. اگر به آن نگاهی بیندازیم به نظر obfuscated مشاهده می‌شود. علت آن هم حذف فواصل، توضیحات و همچنین کاهش طول متغیرها است تا اندازه فایل نهایی به حداقل خود کاهش پیدا کند. البته این فایل از دیدگاه مفسر جاوا اسکریپت یک مرورگر، فایل بی‌نقصی است!
اگر علاقمند هستید که سورس اصلی jQuery را مطالعه کنید، به فایل jquery-xyz.js مراجعه نمائید.

محل الحاق اسکریپت‌های عمومی مورد نیاز برنامه نیز بهتر است در فایل master page یا layout برنامه باشد که به صورت پیش فرض اینکار انجام شده است.
سایر فایل‌های اسکریپتی که در این پوشه مشاهده می‌شوند، یک سری افزونه عمومی یا نوشته شده توسط تیم ASP.NET MVC برفراز jQuery هستند.

به چهار نکته نیز حین استفاده از اسکریپت‌های موجود باید دقت داشت:
الف) همیشه از متد Url.Content همانند تعاریفی که در فایل Views\Shared_Layout.cshtml مشاهده می‌کنید، برای مشخص سازی مسیر ریشه سایت، استفاده نمائید. به این ترتیب صرفنظر از آدرس جاری صفحه، همواره آدرس صحیح قرارگیری پوشه اسکریپت‌ها در صفحه ذکر خواهد شد.
ب) ترتیب فایل‌های js مهم هستند. ابتدا باید کتابخانه اصلی jQuery ذکر شود و سپس افزونه‌های آن‌ها.
ج) اگر اسکریپت‌های jQuery در فایل layout سایت تعریف شده‌اند؛ نیازی به تعریف مجدد آن‌ها در Viewهای سایت نیست.
د) اگر View ایی به اسکریپت ویژه‌ای جهت اجرا نیاز دارد، بهتر است آن‌را به شکل یک section داخل view تعریف کرد و سپس به کمک متد RenderSection این قسمت را در layout سایت مقدار دهی نمود. مثالی از آن‌را در قسمت 20 این سری مشاهده نمودید (افزودن نمایش جمع هر ستون گزارش).

یک نکته
اگر آخرین به روز رسانی‌های ASP.NET MVC را نیز نصب کرده باشید، فایلی به نام packages.config به صورت پیش فرض به هر پروژه جدید ASP.NET MVC اضافه می‌شود. به این ترتیب VS.NET به کمک NuGet این امکان را خواهد یافت تا شما را از آخرین به روز رسانی‌های این کتابخانه‌ها مطلع کند.

آشنایی با Ajax Helpers توکار ASP.NET MVC

اگر به تعاریف خواص و متدهای کلاس WebViewPage دقت کنیم:

```
using System;

namespace System.Web.Mvc
{
    public abstract class WebViewPage<TModel> : WebViewPage
    {
        protected WebViewPage();
        public AjaxHelper<TModel> Ajax { get; set; }
        public HtmlHelper<TModel> Html { get; set; }
        public TModel Model { get; }
        public ViewDataDictionary<TModel> ViewData { get; set; }
        public override void InitHelpers();
        protected override void SetViewData(ViewDataDictionary viewData);
    }
}
```

علاوه بر خاصیت Html که وهله‌ای از آن امکان دسترسی به Html helpers توکار ASP.NET MVC را در یک View فراهم می‌کند، خاصیتی به نام Ajax نیز وجود دارد که توسط آن می‌توان به تعدادی متد AjaxHelper توکار دسترسی داشت. برای مثال توسط متد Ajax.ActionLink می‌توان قسمتی از صفحه را به کمک ویژگی‌های Ajax، به روز رسانی کرد.

مثالی در مورد به روز رسانی قسمتی از صفحه به کمک متد Ajax.ActionLink

ابتدا نیاز است فایل Views\Shared_Layout.cshtml را اندکی ویرایش کرد. برای این منظور سطر الحاق jquery.unobtrusive-ajax.min.js را به فایل layout برنامه اضافه نمائید (اگر این سطر اضافه نشود، متد Ajax.ActionLink همانند یک لینک معمولی رفتار خواهد کرد):

```
<head>
    <title>@ViewBag.Title</title>
    <link href="@Url.Content("~/Content/Site.css")" rel="stylesheet" type="text/css" />
    <script src="@Url.Content("~/Scripts/jquery-1.5.1.min.js")" type="text/javascript"></script>
    <script src="@Url.Content("~/Scripts/jquery.unobtrusive-ajax.min.js")"
type="text/javascript"></script>
</head>
```

سپس مدل ساده و منبع داده زیر را نیز به پروژه اضافه کنید:

```
namespace MvcApplication18.Models
{
    public class Employee
    {
        public int Id { set; get; }
        public string Name { set; get; }
    }
}
```

```
using System.Collections.Generic;

namespace MvcApplication18.Models
{
    public static class EmployeeDataSource
    {
        public static IList<Employee> CreateEmployees()
        {
            var list = new List<Employee>();
            for (int i = 0; i < 1000; i++)
            {
                list.Add(new Employee { Id = i + 1, Name = "name " + i });
            }
            return list;
        }
    }
}
```

در ادامه کنترلر جدیدی را به برنامه با محتوای زیر اضافه کنید:

```
using System.Linq;
using System.Web.Mvc;
using MvcApplication18.Models;

namespace MvcApplication18.Controllers
{
    public class HomeController : Controller
    {
        [HttpGet]
        public ActionResult Index()
        {
            return View();
        }

        [HttpPost] //for IE-8
        public ActionResult EmployeeInfo(int? id)
        {
            if (!Request.IsAjaxRequest())
                return View("Error");

            if (!id.HasValue)
                return View("Error");

            var list = EmployeeDataSource.CreateEmployees();
            var data = list.Where(x => x.Id == id.Value).FirstOrDefault();
            if (data == null)
```

```
                return View("Error");
            return PartialView(viewName: "_EmployeeInfo", model: data);
        }
    }
}
```

بر روی متد Index کلیک راست کرده و گزینه Add view را انتخاب کنید. یک View خالی را به آن اضافه نمائید. همچنین بر روی متد EmployeeInfo کلیک راست کرده و با انتخاب گزینه Add view در صفحه ظاهر شده یک partial view را اضافه نمائید. جهت تمایز بین partial view و view هم بهتر است نام partial view با یک underline شروع شود.
کدهای partial view مورد نظر را به نحو زیر تغییر دهید:

```
@model MvcApplication18.Models.Employee
<strong>Name:</strong> @Model.Name
```

سپس کدهای View متناظر با متد Index را نیز به صورت زیر اعمال کنید:

```
@{
    ViewBag.Title = "Index";
}
<h2>
    Index</h2>

<div id="EmployeeInfo">
    @Ajax.ActionLink(
     linkText: "Get Employee-1 info",
     actionName: "EmployeeInfo",
     controllerName: "Home",
     routeValues: new { id = 1 },
     ajaxOptions: new AjaxOptions
                  {
                      HttpMethod = "POST",
                      InsertionMode = InsertionMode.Replace,
                      UpdateTargetId = "EmployeeInfo",
                      LoadingElementId = "Progress"
                  })
</div>

<div id="Progress" style="display: none">
    <img src="@Url.Content("~/Content/images/loading.gif")" alt="loading..."  />
</div>
```

توضیحات جزئیات کدهای فوق

متد Ajax.ActionLink لینکی را تولید می‌کند که با کلیک کاربر بر روی آن، اطلاعات اکشن متد واقع در کنترلری مشخص، به کمک ویژگی‌های jQuery Ajax دریافت شده و سپس در مقصدی که توسط UpdateTargetId مشخص می‌گردد، بر اساس مقدار InsertionMode، درج خواهد شد (می‌تواند قبل از آن درج شود یا پس از آن و یا اینکه کل محتوای مقصد را بازنویسی کند).
HttpMethod آن هم به POST تنظیم شده تا با IE مشکلی نباشد. از این جهت که IE پیغام‌های GET را کش می‌کند و مساله ساز خواهد شد. توسط پارامتر routeValues، آرگومان مورد نظر به متد EmployeeInfo ارسال خواهد شد.
به علاوه یکی دیگر از خواص کلاس AjaxOptions، برای معرفی حالت بروز خطایی در سمت سرور به نام OnFailure در نظر گرفته شده است. در اینجا می‌توان نام یک متد JavaScript ایی را مشخص کرده و پیغام خطای عمومی را در صورت فراخوانی آن به کاربر نمایش داد. یا توسط خاصیت Confirm آن می‌توان یک پیغام را پیش از ارسال اطلاعات به سرور به کاربر نمایش داد.
به این ترتیب در مثال فوق، id=1 به متد EmployeeInfo به صورت غیرهمزمان ارسال می‌گردد. سپس کارمندی بر این اساس

یافت شده و در ادامه partial view مورد نظر بر اساس اطلاعات کاربر مذکور، رندر خواهد شد. نتیجه کار، در یک div با id مساوی EmployeeInfo درج می‌گردد (InsertionMode.Replace). متد Ajax.ActionLink از این جهت داخل div تعریف شده‌است که پس از کلیک کاربر و جایگزینی محتوا، محو شود. اگر نیازی به محو آن نبود، آن‌را خارج از div تعریف کنید.
عملیات دریافت اطلاعات از سرور ممکن است مدتی طول بکشد (برای مثال دریافت اطلاعات از بانک اطلاعاتی). به همین جهت بهتر است در این بین از تصاویری که نمایش دهنده انجام عملیات است، استفاده شود. برای این منظور یک div با id مساوی Progress تعریف شده و id آن به LoadingElementId انتساب داده شده است. این div با توجه به display: none آن، در ابتدای امر به کاربر نمایش داده نخواهد شد؛ در آغاز کار دریافت اطلاعات از سرور توسط متد Ajax.ActionLink نمایان شده و پس از خاتمه کار مجددا مخفی خواهد شد.
به علاوه اگر به کدهای فوق دقت کرده باشید، از متد Request.IsAjaxRequest نیز استفاده شده است. به این ترتیب می‌توان تشخیص داد که آیا درخواست رسیده از طرف jQuery Ajax صادر شده است یا خیر. البته آنچنان روش قابل ملاحظه‌ای نیست؛ چون امکان دستکاری Http Headers همیشه وجود دارد؛ اما بررسی آن ضرری ندارد. البته این نوع بررسی‌ها را در ASP.NET MVC بهتر است تبدیل به یک فیلتر سفارشی نمود؛ به این ترتیب حجم if و else نویسی در متدهای کنترلرها به حداقل خواهد رسید. برای مثال:

```
[AttributeUsage(AttributeTargets.Class|AttributeTargets.Method)]
public class AjaxOnlyAttribute : ActionFilterAttribute
{
  public override void OnActionExecuting(ActionExecutingContext filterContext)
  {
    if (filterContext.HttpContext.Request.IsAjaxRequest())
    {
      base.OnActionExecuting(filterContext);
    }
    else
    {
      throw new InvalidOperationException("This operation can only be accessed via Ajax requests");
    }
  }
}
```

و برای استفاده از آن خواهیم داشت:

```
[AjaxOnly]
public ActionResult SomeAjaxAction()
{
    return Content("Hello!");
}
```

در مورد کلمه unobtrusive در قسمت بررسی نحوه اعتبار سنجی اطلاعات، توضیحاتی را ملاحظه نموده‌اید. در اینجا نیز از ویژگی‌های data-* برای معرفی پارامترهای مورد نیاز حین ارسال اطلاعات به سرور، استفاده می‌گردد. برای مثال خروجی متد Ajax.ActionLink به شکل زیر است. به این ترتیب امکان حذف کدهای جاوا اسکریپت از صفحه فراهم می‌شود و توسط یک فایل jquery.unobtrusive-ajax.min.js که توسط تیم ASP.NET MVC تهیه شده، اطلاعات مورد نیاز به سرور ارسال خواهد گردید:

```
<a data-ajax="true" data-ajax-loading="#Progress" data-ajax-method="POST"
    data-ajax-mode="replace" data-ajax-update="#EmployeeInfo"
    href="/Home/EmployeeInfo/1">Get Employee-1 info</a>
```

در کل این روش قابلیت نگهداری بهتری نسبت به روش اسکریپت نویسی مستقیم داخل صفحات را به همراه دارد. به علاوه جدا سازی افزونه اسکریپت وفق دهنده این اطلاعات با متد jQuery.Ajax از صفحه جاری، که امکان کش شدن آن‌را به سادگی میسر

می‌سازد.

به روز رسانی اطلاعات قسمتی از صفحه بدون استفاده از متد Ajax.ActionLink

الزامی به استفاده از متد Ajax.ActionLink و فایل jquery.unobtrusive-ajax.min.js وجود ندارد. اینکار را مستقیما به کمک jQuery نیز می‌توان به نحو زیر انجام داد:

```
<a href="#" onclick="LoadEmployeeInfo()">Get Employee-1 info</a>
@section javascript
{
    <script type="text/javascript">
        function LoadEmployeeInfo() {
            showProgress();
            $.ajax({
                type: "POST",
                url: "/Home/EmployeeInfo",
                data: JSON.stringify({ id: 1 }),
                contentType: "application/json; charset=utf-8",
                dataType: "json",
                // controller is returning a simple text, not json
                complete: function (xhr, status) {
                    var data = xhr.responseText;
                    if (status === 'error' || !data) {
                        //handleError
                    }
                    else {
                        $('#EmployeeInfo').html(data);
                    }
                    hideProgress();
                }
            });
        }
        function showProgress() {
            $('#Progress').css("display", "block");
        }
        function hideProgress() {
            $('#Progress').css("display", "none");
        }
    </script>
}
```

توضیحات:

توسط متد jQuery.Ajax نیز می‌توان درخواست‌های Ajax ایی خود را به سرور ارسال کرد. در اینجا type نوع http verb مورد نظر را مشخص می‌کند که به POST تنظیم شده است. Url آدرس کنترلر را دریافت می‌کند. البته حین استفاده از متد توکار Ajax.ActionLink، این لینک به صورت خودکار بر اساس تعاریف مسیریابی برنامه تنظیم می‌شود. اما در صورت استفاده مستقیم از jQuery.Ajax باید دقت داشت که با تغییر تعاریف مسیریابی برنامه نیاز است تا این Url نیز به روز شود.

سه سطر بعدی نوع اطلاعاتی را که باید به سرور POST شوند مشخص می‌کند. نوع json است و همچنین contentType آن برای ارسال اطلاعات یونیکد ضروری است. از متد JSON.stringify برای تبدیل اشیاء به رشته کمک گرفته‌ایم. این متد در تمام مرورگرهای امروزی به صورت توکار پشتیبانی می‌شود و استفاده از آن سبب خواهد شد تا اطلاعات به نحو صحیحی encode شده و به سرور ارسال شوند. بنابراین این رشته ارسالی اطلاعات را به صورت دستی تهیه نکنید؛ چون کاراکترهای زیادی هستند که ممکن است مشکل ساز شده و باید پیش از ارسال به سرور اصطلاحا escape یا encode شوند.

متداول است از پارامتر success برای دریافت نتیجه عملیات متد jQuery.Ajax استفاده شود. اما در اینجا از پارامتر complete آن استفاده شده است. علت هم اینجا است که return PartialView یک رشته را بر می‌گرداند. پارامتر success انتظار دریافت خروجی از نوع json را دارد. به همین جهت در این مثال خاص باید از پارامتر complete استفاده کرد تا بتوان به رشته بدون فرمت خروجی بدون مشکل دسترسی پیدا کرد.

به علاوه چون از یک section برای تعریف اسکریپت‌های مورد نیاز استفاده کرده‌ایم، برای درج خودکار آن در هدر صفحه باید قسمت هدر فایل layout برنامه را به صورت زیر مقدار دهی کرد:

```
@RenderSection("javascript", required: false)
```

دسترسی به اطلاعات یک مدل در View، به کمک jQuery Ajax

اگر جزئی از صفحه که قرار است به روز شود، پیچیده است، روش استفاده از partial viewها توصیه می‌شود؛ برای مثال می‌توان اطلاعات یک مدل را به همراه یک گرید کامل از اطلاعات، رندر کرد و سپس در صفحه درج نمود. اما اگر تنها به اطلاعات چند خاصیت از مدلی نیاز داشتیم، می‌توان از روش‌هایی با سربار کمتر نیز استفاده کرد. برای مثال متد جدید زیر را به کنترلر Home اضافه کنید:

```
[HttpPost] //for IE-8
public ActionResult EmployeeInfoData(int? id)
{
            if (!Request.IsAjaxRequest())
                return Json(false);

            if (!id.HasValue)
                return Json(false);

            var list = EmployeeDataSource.CreateEmployees();
            var data = list.Where(x => x.Id == id.Value).FirstOrDefault();
            if (data == null)
                return Json(false);

            return Json(data);
}
```

سپس View برنامه را نیز به نحو زیر تغییر دهید:

```
<a href="#" onclick="LoadEmployeeInfoData()">Get Employee-2 info</a>
@section javascript
{
    <script type="text/javascript">
        function LoadEmployeeInfoData() {
            showProgress();
            $.ajax({
                type: "POST",
                url: "/Home/EmployeeInfoData",
                data: JSON.stringify({ id: 1 }),
                contentType: "application/json; charset=utf-8",
                dataType: "json",
                // controller is returning the json data
                success: function (result) {
                    if (result) {
                        alert(result.Id + ' - ' + result.Name);
                    }
                    hideProgress();
                },
                error: function (result) {
                    alert(result.status + ' ' + result.statusText);
                    hideProgress();
                }
            });
        }

        function showProgress() {
            $('#Progress').css("display", "block");
        }
        function hideProgress() {
            $('#Progress').css("display", "none");
        }
    </script>
}
```

در این مثال، کنترلر برنامه، اطلاعات مدل را تبدیل به Json کرده و بازگشت خواهد داد. سپس می‌توان به اطلاعات این مدل و خواص آن در View برنامه، در پارامتر success متد jQuery.Ajax، مطابق کدهای فوق دسترسی یافت. اینبار چون خروجی کنترلر تعریف شده از نوع Json است، امکان استفاده از پارامتر success فراهم شده است. همه چیز هم در اینجا خودکار است؛ تبدیل یک شیء به Json و برعکس.
یک نکته: اگر نوع متد کنترلر، HttpGet باشد، نیاز خواهد بود تا پارامتر دوم متد بازگشت Json، مساوی JsonRequestBehavior.AllowGet قرار داده شود.

ارسال اطلاعات فرم‌ها به سرور، به کمک ویژگی‌های Ajax

متد کمکی توکار دیگری به نام Ajax.BeginForm در ASP.NET MVC وجود دارد که کار ارسال غیرهمزمان اطلاعات یک فرم را به سرور انجام داده و سپس اطلاعاتی را از سرور دریافت و قسمتی از صفحه را به روز خواهد کرد. مکانیزم کاری کلی آن بسیار شبیه به متد Ajax.ActionLink می‌باشد. در ادامه با تکمیل مثال قسمت جاری، به بررسی این ویژگی خواهیم پرداخت.
ابتدا متد جستجوی زیر را به کنترلر برنامه اضافه کنید:

```
[HttpPost] //for IE-8
public ActionResult SearchEmployeeInfo(string data)
{
            if (!Request.IsAjaxRequest())
                return Content(string.Empty);

            if (string.IsNullOrWhiteSpace(data))
                return Content(string.Empty);

            var employeesList = EmployeeDataSource.CreateEmployees();
            var list = employeesList.Where(x => x.Name.Contains(data)).ToList();
            if (list == null || !list.Any())
                return Content(string.Empty);

            return PartialView(viewName: "_SearchEmployeeInfo", model: list);
}
```

سپس بر روی نام متد کلیک راست کرده و گزینه add view را انتخاب کنید. در صفحه باز شده، گزینه create a stronlgly typed view را انتخاب کرده و قالب scaffolding را هم بر روی list قرار دهید. سپس گزینه ایجاد partial view را نیز انتخاب کنید. نام آن‌را هم _SearchEmployeeInfo وارد نمائید. برای نمونه خروجی حاصل به نحو زیر خواهد بود:

```
@model IEnumerable<MvcApplication18.Models.Employee>

<table>
    <tr>
        <th>
            Name
        </th>
    </tr>

@foreach (var item in Model) {
    <tr>
        <td>
            @Html.DisplayFor(modelItem => item.Name)
        </td>
    </tr>
}

</table>
```

تا اینجا یک متد جستجو را ایجاد کرده‌ایم که می‌تواند لیستی از رکوردهای کارمندان را بر اساس قسمتی از نام آن‌ها که توسط کاربری جستجو شده است، بازگشت دهد. سپس این اطلاعات را به partial view مورد نظر ارسال کرده و یک جدول را بر اساس آن تولید خواهیم نمود.
اکنون به فایل Index.cshtml مراجعه کرده و فرم Ajax ایی زیر را اضافه نمائید:

```
@using (Ajax.BeginForm(actionName: "SearchEmployeeInfo",
                       controllerName: "Home",
                       ajaxOptions: new AjaxOptions
                                    {
                                         HttpMethod = "POST",
                                         InsertionMode = InsertionMode.Replace,
                                         UpdateTargetId = "EmployeeInfo",
                                         LoadingElementId = "Progress"
                                    }))
    {
      @Html.TextBox("data")
      <input type="submit" value="Search" />
    }
```

اینبار بجای استفاده از متد Html.BeginForm از متد Ajax.BeginForm استفاده شده است. به کمک آن اطلاعات Html.TextBox تعریف شده، به کنترلر Home و متد SearchEmployeeInfo آن، بر اساس HttpMethod تعریف شده، ارسال گردیده و نتیجه آن در یک div با id مساوی EmployeeInfo درج می‌گردد. همچنین اگر اطلاعاتی یافت نشد، به کمک متد return Content یک رشته خالی بازگشت داده می‌شود.
متد Ajax.BeginForm نیز از ویژگی‌های data-* برای تعریف اطلاعات مورد نیاز ارسالی به سرور استفاده می‌کند. بنابراین نیاز به سطر الحاق jquery.unobtrusive-ajax.min.js در فایل layout برنامه جهت وفق دادن این اطلاعات unobtrusive به اطلاعات مورد نیاز متد jQuery.Ajax وجود دارد.

```
<form action="/Home/SearchEmployeeInfo" data-ajax="true"
          data-ajax-loading="#Progress" data-ajax-method="POST"
          data-ajax-mode="replace" data-ajax-update="#EmployeeInfo"
          id="form0" method="post">
       <input id="data" name="data" type="text" value="" />
       <input type="submit" value="Search" />
</form>
```

کتابخانه کمکی «ASP.net MVC Awesome - jQuery Ajax Helpers»

علاوه بر متدهای توکار Ajax همراه با ASP.NET MVC، سایر علاقمندان نیز یک سری Ajax helper را بر اساس افزونه‌های jQuery تدارک دیده‌اند که از آدرس زیر قابل دریافت هستند:

http://awesome.codeplex.com/

افزودن فرم‌ها به کمک jQuery.Ajax و فعال سازی اعتبار سنجی سمت کلاینت

در ASP.NET MVC چون ViewState حذف شده است، امکان تزریق فرم‌های جدید به صفحه یا به روز رسانی قسمتی از صفحه توسط jQuery Ajax به سهولت و بدون دریافت پیغام «viewstate is corrupted» در حین ارسال اطلاعات به سرور، میسر است.
در این حالت باید به یک نکته مهم نیز دقت داشت: «اعتبار سنجی سمت کلاینت دیگر کار نمی‌کند». علت اینجا است که در حین بارگذاری متداول یک صفحه، متد زیر به صورت خودکار فراخوانی می‌گردد:

```
$.validator.unobtrusive.parse("#{form-id}");
```

اما با به روز رسانی قسمتی از صفحه، دیگر اینچنین نخواهد بود و نیاز است این فراخوانی را دستی انجام دهیم. برای مثال:

```
$.ajax
({
    url: "/{controller}/{action}/{id}",
    type: "get",
    success: function(data)
    {
        $.validator.unobtrusive.parse("#{form-id}");
    }
});

//or
$.get('/{controller}/{action}/{id}', function (data) { $.validator.unobtrusive.parse("#{form-id}"); });
```

شبیه به همین مساله را حین استفاده از Ajax.BeginForm نیز باید مد نظر داشت:

```
@using (Ajax.BeginForm(
    "Action1",
    "Controller",
    null,
    new AjaxOptions {
        OnSuccess = "onSuccess",
        UpdateTargetId = "result"
    },
    null)
)
{
    <input type="submit" value="Save" />
}

var onSuccess = function(result) {
    // enable unobtrusive validation for the contents
    // that was injected into the <div id="result"></div> node
    $.validator.unobtrusive.parse("#result");
};
```

در این مثال در پارامتر UpdateTargetId، مجددا یک فرم رندر می‌شود. بنابراین اعتبار سنجی سمت کلاینت آن دیگر کار نخواهد کرد مگر اینکه با مقدار دهی خاصیت OnSuccess، مجددا متد unobtrusive.parse را فراخوانی کنیم.

نظرات خوانندگان

نویسنده: ilia
تاریخ: ۱۳۹۱/۰۲/۰۵ ۱۹:۰۹:۳۶

با سلام. مقالات شما را با جدیت دنبال می کنم . خیلی برام کاربرد دارند. متشکرم آقای نصیری .

نویسنده: amir hosein jelodari
تاریخ: ۱۳۹۱/۰۲/۰۶ ۱۹:۴۴:۰۸

سلام ... خسته نباشید ... این رو هم امتحان کنید :

/http://jsaction.codeplex.com

نویسنده: Naser Tahery
تاریخ: ۱۳۹۱/۰۲/۰۶ ۲۲:۰۴:۰۴

ممنون.
آیا استفاده از این افزونه /http://awesome.codeplex.com رایگان است؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۲/۰۶ ۲۲:۲۶:۵۰

مجوز آن «Microsoft Public License» است. به این معنا که مجاز هستید از آن در شکل «باینری» در هر نوع پروژه‌ای استفاده کنید.

نویسنده: Naser Tahery
تاریخ: ۱۳۹۱/۰۲/۰۶ ۲۳:۰۷:۵۹

عذرخواه. ولی متوجه نمیشم در شکل باینری یعنی چی؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۲/۰۶ ۲۳:۱۷:۲۴

یک سری از مجوزهای سورس باز به این شکل هستند. نمونه دیگر آن LGPL است. مثلا NHibernate مجوز LGPL دارد. به این معنا که مجاز هستید از آن به شکل باینری (یعنی فایل‌های dll کامپایل شده آن) در هر نوع پروژه تجاری، غیرتجاری، باز، بسته ... بدون محدودیت استفاده کنید. اما اگر سورس آن‌ها را مستقیما به پروژه خود اضافه و کامپایل کنید، نیاز است تا سورس کارتان را هم ارائه دهید.

نویسنده: علیرضا
تاریخ: ۱۳۹۱/۰۴/۲۳ ۱۶:۷

با سلام و درود و تشکر فراوان از شما
استاد من وقتی در partial View این @Molde.Name رو ست میکنم
در مرورگر Name. MvcApplication7.Models.Employee **Name:** این را مشاهده میکنم
علت چیست؟
سپاس از شما

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۳ ۱۶:۴۳

Molde نیست. Model است همچنین M آن هم باید بزرگ باشد تا با @model اشتباه نشود.
در کل امکان دیباگ پروژه‌های شخصی از راه دور میسر نیست. باید پروژه شما باشه تا بشود آن‌را بررسی کرد.

نویسنده: سعید یزدانی
تاریخ: ۱۳۹۱/۱۱/۲۳ ۱۷:۱۱

سلام خسته نباشید
در روش استفاده مستقیم از ajax در jquery , خاصیت content type با DataType چه تفاوتی دارد .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۳ ۱۷:۳۷

contentType معادل است با mime type در کارهای وب و بیانگر نوع اطلاعات ارسالی است به سرور. مثلا یک فایل pdf است یا یک فایل تصویر png و امثال آن.
ذکر dataType در jQuery نوع اطلاعات برگشتی از سرور رو مشخص می‌کنه. برای مثال اگر xml ذکر شود، اطلاعات دریافتی از سرور را به فرمت xml در اختیار شما قرار می‌دهد.

نویسنده: مولانا
تاریخ: ۱۳۹۲/۰۱/۲۷ ۱۹:۴

با سلام.
علت اینکه پارامتر ids مربوط به اکشن delete همواره null میگیرد چیست؟

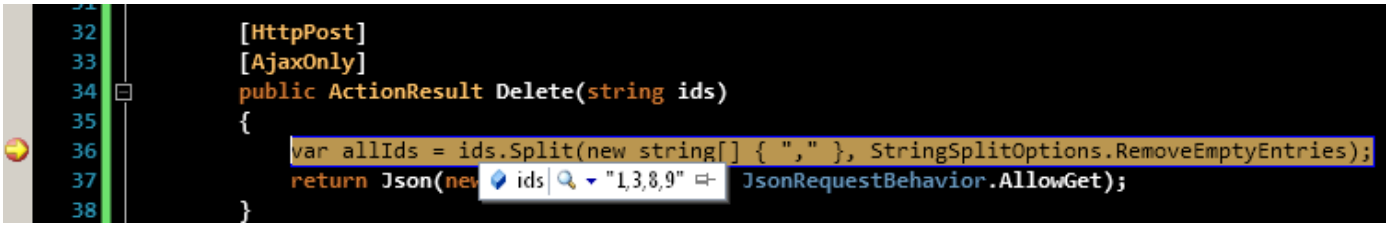
```
@{
    var postUrl = Url.Action(actionName: "Delete", controllerName: "Student");
}
<div class="deleteDialog">
    <div>
        آیتم‌های انتخاب شده حذف خواهند شد. آیا تأیید می‌کنید؟
    </div>
    <p>
        <input type="submit" id="btn_SubmitDelete" value="حذف" />
        <input type="submit" id="btn_CancelDelete" value="انصراف" />
    </p>
</div>
<script type="text/javascript">
    $(function () {
        $("#btn_SubmitDelete").click(function (e) {
            var button = $(this);
            e.preventDefault();
            var data = "1,3,8,9";
            $.ajax({
                type: "POST",
                url: "@postUrl",
                data:  JSON.stringify({ ids: data}),
                contentType: "application/json; charset=utf-8",
                dataType: 'json',
                cache: false,
                beforeSend: function () { },
                success: function (html) {
                    alert(html);
                    $(".deleteDialog").parent("div").css("display", "none");
                },
                complete: function () {
                    button.removeAttr('disabled');
                    button.val("حذف");
                }
            });
        });
        $("#btn_CancelDelete").click(function (e) {
            e.preventDefault();
            var button = $(this);
            $(".deleteDialog").parent("div").css("display", "none");
        });
```

```
    });
</script>
```

```
[HttpGet]
 public ActionResult Delete()
 {
       return PartialView("Pv_Delete");
 }
 [HttpPost]
 [AjaxOnly]
  public ActionResult Delete(string ids)
  {
            var allIds = ids.Split(new string[] { "," },
StringSplitOptions.RemoveEmptyEntries).ToList();
            Thread.Sleep(2000);
           if (true)
           {
                return Json(new { result = "ok" }, JsonRequestBehavior.AllowGet);
           }
            return Json(new { result = "error" });
 }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۲۷ ۲۰:۴۱

اینجا کار می‌کنه: (با استفاده از فایرباگ بررسی کنید آیا خطایی در ارسال هست یا خیر. برگه network آن؛ همچنین JSON.stringify از IE8 به بعد به صورت توکار وجود دارد و از فایرفاکس 3.1 به بعد)



نویسنده: میثم خوشقدم
تاریخ: ۱۳۹۲/۰۴/۲۰ ۱۱:۲۵

سلام

خسته نباشید

در مقالتون فرمودین بجای Ajax.BeginForm می‌توان از $.ajax استفاده نمود . اما در ستفاده از Ajax داده به صورت دستی در پارامتر Data پاس داده شده . آیا امکان این هست که Ajax را همانند Ajax.Beginform دات نت پیاده سازی کرد تا با Post شدن فرم اطلاعات اتوماتیک و بدون مقدار دهی پارامتر مذکور ارسال شود.

چرا که من در فرم، فایل آپلود می‌کنم ولی کتابخانه unobtrusive با HttpPostFileBase مشکل داره و می‌خوام از Helper دت نت استفاده نکنم و مستقیما از Ajax و بدون کتابخانه ذکر شده استفاده کنم.

در این خصوص نظر شما چیه ؟ آیا کاری که من به سبب مشکلی که در آپلود فایل در Helperهای Ajax دارم صحیح است؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۲۰ ۱۱:۵۵

- مراجعه کنید به دوره « استفاده از افزونه‌ها و امکانات jQuery در ASP.NET MVC » برای تکمیل بحث. برای نمونه مطلب « افزونه‌ای برای کپسوله سازی نکات ارسال یک فرم ASP.NET MVC به سرور توسط jQuery Ajax » اطلاعات یک فرم رو به صورت خودکار توسط متد form.serialize به سرور ارسال می‌کند.
- نمی‌تونید از Ajax معمولی (یا به عبارتی XMLHttpRequest) برای ارسال فایل استفاده کنید. یا باید از سیلورلایت یا فلش استفاده کنید، یا از مرورگرهایی که XMLHttpRequest *Level 2* را پشتیبانی کنند (از IE 10 به بعد مثلا)، امکان Ajax upload توکار به همراه گزارش درصد آپلود را بدون نیاز به فلش یا سیلورلایت، دارند. یک نمونه پیاده سازی آن

نویسنده: سهیل
تاریخ: ۱۳۹۲/۰۴/۲۶ ۱۹:۱۹

با سلام
کدی که در آخر مطلب جهت فعال کردن دوباره اعتبار سنجی نوشتید چگونه باید استفاده کرد!؟
منظورم:

```
var onSuccess = function(result) {
// enable unobtrusive validation for the contents
// that was injected into the <div id="result"></div> node
$.validator.unobtrusive.parse("#result");
};
```

سوال دیگه اینکه چطور می‌تونم محل دقیق loading Image رو مشخص کرد، مثلا هنگام کلیک روی دکمه سرچ کنار آن نمایش داده شود و هنگام کلیک روی لینک کنار لینک نمایش داده شود.
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۲۶ ۱۹:۳۰

- دقیقا به همان نحوی که نوشته شده. onSuccess پس از پایان کار عملیات Ajax ایی فراخوانی می‌شود. در آنجا متد یاد شده را بر روی Id محتوای پویای بارگذاری شده فراخوانی کنید. یک نمونه مثال دیگر آن استفاده از این روش در مطلب « نمایش فرم‌های مودال Ajax ایی در ASP.NET MVC به کمک Twitter Bootstrap » است.
- محل قرارگیری تمام عناصر رو در صفحه با استفاده از jQuery می‌شود تغییر داد.
اگر با مفاهیمی مانند Id عناصر و نحوه استفاده از آن‌ها در jQuery آشنایی ندارید، یک دوره مقدماتی در اینباره در سایت موجود است .

نویسنده: امیر خلیلی
تاریخ: ۱۳۹۲/۰۸/۲۵ ۱۰:۴۴

با درود؛ من با استفاده از متد jQuery.Ajax و درخواست از یک کنترلر برای نمایش اطلاعات از دیتابیس به روش زیر عمل کردم

```
<script type="text/javascript">

        $(function () {
            getData();
        });

        function getData() {
        var $tbl = $('#tblEmployee');
            $.ajax({
            url: 'Home/EmployeeInfoData',
            type: 'Post',
                datatype: 'json',
                success: function (data) {
                    if (data.length > 0) {
                        $tbl.empty();
                        $tbl.append(' <tr><th>ID</th><th>Name</th><th>Family</th></tr>');
```

```
                        var rows = [];
                        for (var i = 0; i < data.length; i++) {
                            rows.push(' <tr><td>' + data[i].Id + '</td><td>' + data[i].Name +
'</td><td>' + data[i].Family + '</td></tr>');
                        }
                        $tbl.append(rows.join(''));
                    }
                }
            });
        }
    </script>
```

و کنترلر مربوط

```
[HttpPost]
        public ActionResult EmployeeInfoData()
        {

 InfoEmployee mp = new InfoEmployee();
             var names = mp.GetData();
         return Json(names);
        }
```

و سوال اینکه وقتی از Return View استفاده کردم هیچ رکوردی بازگردانده نشد و با یک صفحه سفید مواجه شدم و باید حتما از Return Json استفاده کنم تا اطلاعات درخواستی نمایش داده بشه؟ آیا حتما باید از Return Json استفاده کرد ؟ و یا در کد نویسی من جایی اشکال هست ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۵ ۱۱:۳۱

return View نهایتا یک رشته را باز می‌گرداند که حاوی اطلاعات رندر شده نهایی یک View است. این رشته دریافتی (که فرمت HTML را دارد) دیگر فرمت JSON استاندارد را نخواهد داشت و در قسمت **success** قابل پردازش نیست. البته اگر به مثال‌ها دقت کنید، می‌شود توسط jQuery Ajax یک محتوای HTMLایی از پیش پردازش شده را هم دریافت و سپس آن‌را به صفحه اضافه کرد (نمونه‌اش function LoadEmployeeInfo در مطلب فوق است). اینکار باید در قسمت **complete** انجام شود. ضمنا بهتر است از return PartialView برای این نوع کارهای خاص Ajaxایی که HTML بر می‌گردانند، استفاده شود تا ارجاعی به فایل layout در این partial view بازگشتی وجود نداشته باشد.

نویسنده: امیر خلیلی
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۱:۴۸

وقتی در مثال بالا در ابتدا یک کنترلر دیگه را اجرا کنیم مثلا در ابتدا کنترلر Login برنامه اجرا بشه و سپس کنترلر بالا (Home/EmployeeInfoData) را درخواست کنیم برنامه دچار مشکل میشه و با صفحه سفید مواجه هستیم! یعنی اگه در routes.MapRoute برنامه نام کنترلر دیگه ای باشه و سپس کنترلر بالا را فراخانی بشه هیچ مقداری نمایش داده نمیشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۲:۵۹

نیاز هست بتونید برنامه‌های ای‌جکسی را دیباگ کنید:
نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery

نویسنده: امیر خلیلی
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۳:۲۰

بسیار عالی
با دیباگ برنامه با افزونه فایرباگ خطای زیر مشاهده شد !

```
NetworkError: 404 Not Found - http://localhost:9303/Home/Home/EmployeeInfoData
```

اما چرا کنترلر Home دوبار در لینک آورده شده ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۳:۳۰

برای اینکه نکته « نحوه صحیح تولید Url در ASP.NET MVC » را رعایت نکردید.

نویسنده: امیر خلیلی
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۳:۴۱

بله دقیقا همین مشکل بود
مشکل اصلی امثال من اینه که همیشه مقدمه‌ها را ساده میگیریم
خیلی ممنوم ار شما

نویسنده: امیر خلیلی
تاریخ: ۱۳۹۲/۱۰/۲۲ ۱۱:۲۰

با درود؛ میشه لطفا بفرمایید در مرورگر IE از ورژن چند از Jquery Ajax پشتیبانی میشه ؟ برای بکارگیری فرمت Json

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۲۲ ۱۱:۵۲

- AJAX اختراع خود مایکروسافت هست و اولین بار در برنامه Microsoft Outlook Web Access از آن استفاده شد (بنابراین مرورگرهای آن اولین مرورگرهایی هستند که از AJAX پشتیبانی کرده‌اند).
- متد JSON.stringify از IE 8 به بعد به صورت توکار پشتیبانی می‌شود. برای مرورگرهای قدیمی‌تر از فایل json2.js استفاده کنید.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۸ ۱۴:۲۲

سلام ، باتشکر از آموزش خوبتون.
قسمت آخر آموزشتون (فعال سازی اعتبار سنجی سمت کلاینت) رو نتونستم متوجه بشم.
من این ویو رو توسط Ajax.ActionLink فراخونی کردم. برای اینکه اعتبار سنجی کلاینت اون کار کنه کدهای زیر کافیه؟

```
@model MvcApplication1.Models.Com
<script>
    var onSuccess = function (result) {
        $.validator.unobtrusive.parse("#result");
    };
</script>
<div id="result" >
@using (Ajax.BeginForm(
    "SendCommentAction",
    "Home",
    null,
    new AjaxOptions {
        OnSuccess = "onSuccess",
        UpdateTargetId = "result"
    },
    null)
)
{
    @Html.TextBoxFor(x=>x.Contect) <br/>
@Html.ValidationMessageFor(x=>x.Contect)
<br/>
@Html.ValidationSummary()
```

```
    <input type="submit" value="Save" />
}
</div>
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۸ ۱۴:۳۴

$.validator.unobtrusive.parse باید در جایی فراخوانی شود که کار load اولیه را انجام داده:

```
@Ajax.ActionLink("Test",
                 "action",
                 new AjaxOptions { HttpMethod = "POST",
                                   InsertionMode = InsertionMode.Replace,
                                   UpdateTargetId = "UserDiv",
                                   OnSuccess="$.validator.unobtrusive.parse('#my_form_id');"
                                 }
               )
```

اگر این کد را در partial view ایی که قرار است load شود قرار دادید، در آنجا فقط بنویسید:

```
<script type="text/javascript">
    $.validator.unobtrusive.parse("#my_frm_id");
</script>
```

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۹ ۲۰:۲۴

با تشکر از آموزش خوبتون،
امکان ویرایش Confirm در Ajax.BeginForm یا Ajax.ActionLink وجود داره؟
مثلا میخوایم از jQuery Impromptu استفاده کنیم .

```
@using(Ajax.BeginForm(MVC.Post.ActionNames.New ,
 MVC.Post.Name,
 new AjaxOptions(){
   HttpMethod = "Post" ,
   Confirm ="$.prompt('درخواست تایید - موافقید ؟',{prefix: 'dnt',buttons: { 'تایید': true, 'انصراف':
false }});" }))
```

ولی کار نمیکنه. یا تنها راهش استفاده از id دکمه‌ی Submit و Confirm دادن با کمک Jquery هستش؟
و یه موضوع دیگه . انجام اینکار درسته؟

```
@using(Ajax.BeginForm(MVC.Post.ActionNames.New ,
MVC.Post.Name,
new AjaxOptions(){
   HttpMethod = "Post" ,
   OnSuccess = "var noty = window.noty({ text: 'نظر شما ثبت شد بعد از تایید نمایش داده میشود', type:
'success', layout: 'center', timeout: 4000 });"
}))
```

منظورم نمایش پیغام به کاربر که عملیات موفق آمیز بوده ، یعنی وقتی متد OnSuccess اجرا میشود اطمینان داشته باشیم که Action مربوطه کاملا اجرا شده و اگه مشکلی در حین اجرای Action پیش بیاد این متد فراخونی نمیشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۹ ۲۱:۲۸

- فایل jquery.unobtrusive-ajax.js سورس باز هست و جزو پروژه. خاصیت confirm آن مستقیما از window.confirm استاندارد مرورگرها استفاده می‌کند و قابلیت سفارشی سازی ندارد. فقط یک متن را می‌شود اینجا تنظیم کرد.
- AjaxOptions خاصیت OnFailure هم دارد.

نویسنده: جوادنبی
تاریخ: ۱۳۹۳/۰۱/۰۳ ۱۶:۳۱

با سلام؛ در یک view من یک جدولی دارم که درون ستون آخر آن یک دکمه قرار دارد برای حذف آن رکورد که به صورت ایجکس می‌باشد.عمل حذف با موفقیت انجام می‌شود ولی چون صفحه رفرش نشده آن رکورد حذف شده هنوز نمایش داده می‌شود چگونه می‌توانم کاری کنم که بعد از حذف رکورد دیگر نمایش داده نشود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۰۳ ۱۶:۳۵

« حذف یک ردیف از اطلاعات به همراه پویانمایی محو شدن اطلاعات آن توسط jQuery در ASP.NET MVC »

نویسنده: ربال
تاریخ: ۱۳۹۳/۰۸/۲۲ ۱۶:۶

سلام. اگر قرار باشد در خاصیت OnSuccess مربوط به Ajax.BeginForm یک پارشال ویو هم در مدال رندر شود اون موقع چرا validator.unobtrusive.parse عمل نمیکند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۲ ۱۸:۱۹

این متد را باید در زمان نمایش نهایی فرم مودال، فراخوانی کنید. مطابق معمول روال jQuery، رخداد document ready فرم دوم (یا رخدادهایی مانند open، show یا complete و امثال آن مختص به فرم مودال) را باید مقدار دهی کنید؛ نه فراخوانی آن متد، در فرم جاری. یک مثال