

اگر مطالب سایت جاری را مطالعه و دنبال کرده باشید، تاکنون به صورت پراکنده نکات زیادی را در مورد استفاده از jQuery Ajax تهیه و ارائه کرده‌ایم. در این مطلب قصد داریم تا این نکات را نظم بخشیده و جهت استفاده مجدد، به صورت یک افزونه کپسوله سازی کنیم.

در کدها و افزونه‌ای که در ادامه ارائه خواهند شد، این مسایل در نظر گرفته شده است:

- چگونه اعتبار سنجی سمت کاربر را در حین استفاده از Ajax فعال کنیم.
- چگونه از چندبار کلیک کاربر در حین ارسال فرم به سرور جلوگیری نمائیم.
- چگونه Complex Types قابل تعریف در EF Code first را نیز در اینجا مدیریت کنیم.
- نحوه تعریف صحیح آدرس‌های کنترلرها چگونه باید باشد.
- نحوه اعلام وضعیت لاگین شخص به او، در صورت بروز مشکل.
- ارسال صحیح anti forgery token در حین اعمال Ajax ایی.
- بررسی Ajax بودن درخواست رسیده و تهیه یک فیلتر سفارشی مخصوص آن.
- از کش شدن اطلاعات Ajax ایی جلوگیری شود.

ابتدا معرفی مدل برنامه

```
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;

namespace jQueryMvcSample01.Models
{
    public class User
    {
        [Required(ErrorMessage = "(*)"), DisplayName("نام")]
        public string Name { set; get; }

        public PhoneInfo PhoneInfo { set; get; }
    }

    public class PhoneInfo
    {
        [Required(ErrorMessage = "(*)"), DisplayName("تلفن")]
        public string Phone { get; set; }

        [Required(ErrorMessage = "(*)"), DisplayName("پیش شماره")]
        public string Ext { get; set; }
    }
}
```

همانطور که ملاحظه می‌کنید، خاصیت PhoneInfo، تو در تو یا به نوعی Complex است. اگر از ابزارهای Scaffolding توکار VS.NET برای تولید View متناظر استفاده کنیم، فیلد تو در تو PhoneInfo را لحاظ نخواهد کرد، اما ... مهم نیست. تعریف دستی آن هم کار می‌کند.

کدهای کنترلر برنامه

```
using System.Web.Mvc;
using jQueryMvcSample01.Models;
using jQueryMvcSample01.Security;

namespace jQueryMvcSample01.Controllers
{
    public class HomeController : Controller
```

```

{
    [HttpGet]
    public ActionResult Index()
    {
        return View(); //نمایش فرم
    }

    [HttpPost]
    [AjaxOnly] // فقط در حالت ای‌جکس قابل دسترسی باشد
    [ValidateAntiForgeryToken]
    public ActionResult Index(User user)
    {
        if (this.ModelState.IsValid)
        {
            // ذخیره سازی در بانک اطلاعاتی ...
            System.Threading.Thread.Sleep(3000);

            return Content("ok");//کار بودن کار
        }

        return Content(null);//ارسال خطا
    }
}
}

```

در اینجا در متد Index، اطلاعات شیء User به صورت Ajaxی دریافت شده و پس از آن برای مثال قابلیت ذخیره سازی را خواهد داشت.

چند نکته در اینجا حائز اهمیت هستند:

الف) استفاده از ویژگی AjaxOnly (که کدهای آن را در پروژه پیوست می‌توانید مشاهده نمائید)، جهت صرفا پردازش درخواست‌های Ajaxی.

ب) استفاده از ویژگی ValidateAntiForgeryToken در حین اعمال اجکسی. اگر سایت‌های مختلف را در اینباره جستجو کنید، عموماً برای پردازش آن در حین استفاده از jQuery Ajax بسیار مشکل دارند.

ج) استفاده از return Content برای اعلام نتیجه کار. اگر اطلاعات ثبت شد، یک ok یا هر عبارت دیگری که علاقمند بودید ارسال گردیده و در غیراینصورت null بازگشت داده می‌شود.

کدهای افزونه PostMvcFormAjax

```

// 
(function ($) {
    $.fn.PostMvcFormAjax = function (options) {
        var defaults = {
            postUrl: '/',
            loginUrl: '/login',
            beforePostHandler: null,
            completeHandler: null,
            errorHandler: null
        };
        var options = $.extend(defaults, options);

        var validateForm = function (form) {
            // فعال سازی دستی اعتبار سنجی جی‌کوئری
            var val = form.validate();
            val.form();
            return val.valid();
        };

        return this.each(function () {
            var form = $(this);
            // اگر فرم اعتبار سنجی نشده، اطلاعات آن ارسال نشود
            if (!validateForm(form)) return;

            // در اینجا می‌توان مثلاً دکمه‌ای را غیرفعال کرد
            if (options.beforePostHandler)
                options.beforePostHandler(this);

            // اطلاعات نباید کش شوند
            $.ajaxSetup({ cache: false });

            $.ajax({
</pre>
</div>
<div data-bbox="495 957 529 971" data-label="Page-Footer">۲/۱۰</div>
```

```

        type: "POST",
        url: options.postUrl,
        data: form.serialize(), //می‌کند آنرا ارسال می‌کند
        complete: function (xhr, status) {
            var data = xhr.responseText;
            if (xhr.status == 403) {
                window.location = options.loginUrl; //اجرا می‌شود
            }
            else if (status === 'error' || !data) {
                if (options.errorHandler)
                    options.errorHandler(this);
            }
            else {
                if (options.completeHandler)
                    options.completeHandler(this);
            }
        }
    });
})(jQuery);
// ]]>

```

چند نکته مهم در تهیه این افزونه رعایت شده:

الف) فعال سازی دستی اعتبار سنجی جی کوئری، از این جهت که این نوع اعتبار سنجی به صورت پیش فرض تنها در حالت postback و ارسال کامل صفحه به سرور فعال می‌شود.

ب) استفاده از متد serialize جهت پردازش یکباره کل اطلاعات و فیلدهای یک فرم.

نکته مهم این متد ارسال فیلد مخفی anti forgery token نیز می‌باشد. فقط باید دقت داشت که این فیلد در حالتی که dataType به json تنظیم شود و همچنین از متد serialize استفاده گردد، در ASP.NET MVC پردازش نمی‌گردد (خیلی مهم!). به همین جهت در اینجا dataType تنظیمات jQuery Ajax حذف شده است.

ج) تنظیم cache به false در تنظیمات ابتدایی jQuery Ajax تا اطلاعات ارسالی و دریافتی کش نشوند و مشکل ساز نگردند.

د) بررسی xhr.status == 403 که توسط SiteAuthorizeAttribute (جایگزین بهتر فیلتر Authorize توکار ASP.NET MVC که کدهای آن در پروژه پیوست قابل دریافت است) و هدایت کاربر به صفحه لاگین

تعریف View ایی که از اشیاء تو در تو استفاده می‌کند و همچنین از افزونه فوق برای ارسال اطلاعات بهره خواهد برد:

```

@model jQueryMvcSample01.Models.User
@{
    ViewBag.Title = "تعریف کاربر";
    var postUrl = Url.Action(actionName: "Index", controllerName: "Home");
}
@using (Html.BeginForm(actionName: "Index", controllerName: "Home",
    method: FormMethod.Post,
    htmlAttributes: new { id = "UserForm" })))
{
    @Html.ValidationSummary(true)
    @Html.AntiForgeryToken()

    <fieldset>
        <legend>تعریف کاربر</legend>
        <div class="editor-label">
            @Html.LabelFor(model => model.Name)
        </div>
        <div class="editor-field">
            @Html.EditorFor(model => model.Name)
            @Html.ValidationMessageFor(model => model.Name)
        </div>
        <div class="editor-label">
            @Html.LabelFor(model => model.PhoneInfo.Ext)
        </div>
        <div class="editor-field">
            @Html.EditorFor(model => model.PhoneInfo.Ext)
            @Html.ValidationMessageFor(model => model.PhoneInfo.Ext)
        </div>
        <div class="editor-label">
            @Html.LabelFor(model => model.PhoneInfo.Phone)
        </div>
    </fieldset>
}

```

```

<div class="editor-field">
    @Html.EditorFor(model => model.PhoneInfo.Phone)
    @Html.ValidationMessageFor(model => model.PhoneInfo.Phone)
</div>
<p>
    <input type="submit" id="btnSave" value="ارسال" />
</p>
</fieldset>
}
@section JavaScript
{
    <script type="text/javascript">
        $(document).ready(function () {
            $("#btnSave").click(function (event) {
                // جلوگیری از پست یک به سرور
                event.preventDefault();

                var button = $(this);

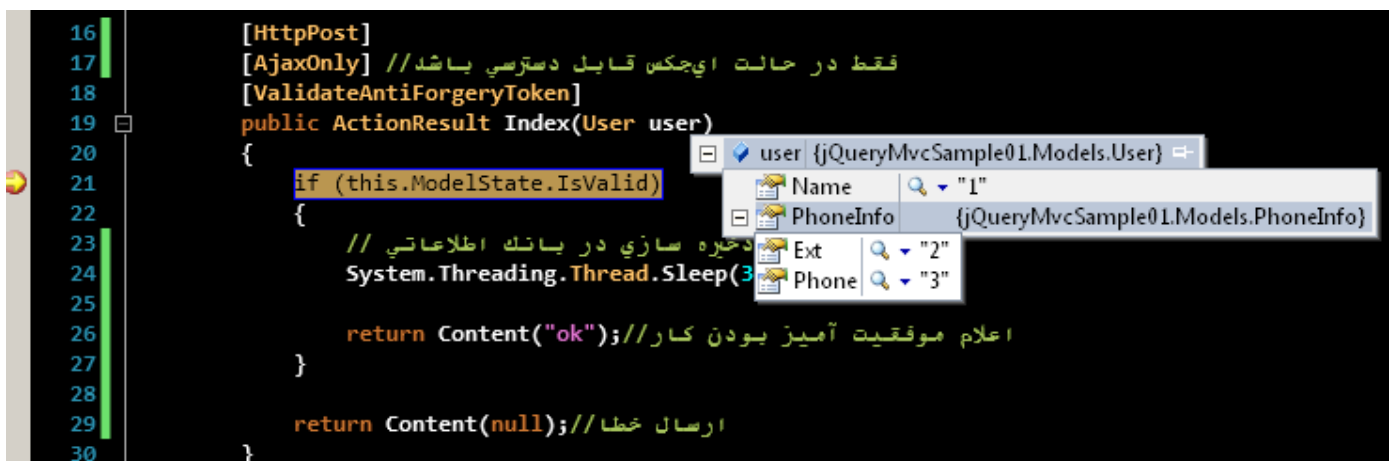
                $("#UserForm").PostMvcFormAjax({
                    postUrl: '@postUrl',
                    loginUrl: '/login',
                    beforePostHandler: function () {
                        // غیرفعال سازی دکمه ارسال
                        button.attr('disabled', 'disabled');
                        button.val("...");
                    },
                    completeHandler: function () {
                        // فعال سازی مجدد دکمه ارسال
                        alert('انجام شد');
                        button.removeAttr('disabled');
                        button.val("ارسال");
                    },
                    errorHandler: function () {
                        alert('خطایی رخ داده است');
                    }
                });
            });
        });
    </script>
}

```

همانطور که عنوان شد، مهم نیست که اشیاء تو در تو توسط ابزار Scaffolding پشتیبانی نمی‌شود. این نوع خواص را به همان نحو متداول ذکر زنجیره وار خواص می‌توان معرفی و استفاده کرد:

```
@Html.EditorFor(model => model.PhoneInfo.Phone)
```

هم اعتبار سنجی سمت کلاینت آن کار می‌کند و هم اطلاعات آن به اشیاء و خواص متناظر به خوبی نگاشت خواهد شد:



```

16 [HttpPost]
17 [AjaxOnly] // فقط در حالت ای‌جکس قابل دسترسی باشد
18 [ValidateAntiForgeryToken]
19 public ActionResult Index(User user)
20 {
21     if (this.ModelState.IsValid)
22     {
23         // ذخیره سازی در بانک اطلاعاتی
24         System.Threading.Thread.Sleep(3000);
25
26         return Content("ok"); // اعلام موفقیت آمیز بودن کار
27     }
28
29     return Content(null); // ارسال خطا
30 }

```

در ادامه نحوه استفاده از افزونه PostMvcFormAjax را مشاهده می‌کنید. چند نکته نیز در اینجا حائز اهمیت هستند:

الف) توسط `htmlAttributes` یک `id` برای فرم تعریف کرده‌ایم تا در افزونه `PostMvcFormAjax` مورد استفاده قرار گیرد.

ب) `loginUrl` و `postUrl` را همانند متغیر تعریف شده در ابتدای `View` توسط `Url.Action` باید تعریف کرد تا در صورتیکه سایت ما در ریشه اصلی قرار نداشت، باز هم به صورت خودکار مسیر صحیحی محاسبه و ارائه گردد.

ج) نحوه غیرفعال سازی و فعال سازی دکمه `submit` را در روال‌های `beforePostHandler` و `completeHandler` ملاحظه می‌کنید.

این مساله برای جلوگیری از کلیک‌های مجدد یک کاربر ناشکیبا و جلوگیری از ثبت اطلاعات تکراری بسیار مهم است.

د) کل این اطلاعات، در یک `section` به نام `JavaScript` ثبت شده است. این `section` در فایل `layout` برنامه به صورت زیر مورد استفاده قرار خواهد گرفت و به این ترتیب مقدار دهی خواهد شد:

```
<head>
  <title>@ViewBag.Title</title>
  <link href="@Url.Content("Content/Site.css")" rel="stylesheet" type="text/css" />
  <script src="@Url.Content("~/Scripts/jquery-1.9.1.min.js")" type="text/javascript"></script>
  <script src="@Url.Content("~/Scripts/jquery.validate.min.js")" type="text/javascript"></script>
  <script src="@Url.Content("~/Scripts/jquery.unobtrusive-ajax.min.js")"
type="text/javascript"></script>
  <script src="@Url.Content("~/Scripts/jquery.validate.unobtrusive.min.js")"
type="text/javascript"></script>
  <script src="@Url.Content("~/Scripts/jquery.PostMvcFormAjax.js")" type="text/javascript"></script>
  @RenderSection("JavaScript", required: false)
</head>
```

دریافت کدهای کامل این قسمت

[jQueryMvcSample01.zip](#)

نظرات خوانندگان

نویسنده: molana11

تاریخ: ۱۶:۵۵ ۱۳۹۲/۰۱/۰۴

با سلام.

دستور val.form چه کاری انجام میدهد؟

با تشکر.

نویسنده: وحید نصیری

تاریخ: ۱۷:۰۶ ۱۳۹۲/۰۱/۰۴

[مطابق مستندات](#) افزونه jQuery Validator که در ASP.NET MVC استفاده می‌شود، متد Validate فقط یک سری روال رویدادگردان را تنظیم می‌کند تا در زمان postback کامل به سرور فعال شوند. برای اینکه در اینجا postback کامل به سرور نداریم (عملیات Ajax ایی است)، نیاز است عملیات اعتبارسنجی را دستی فراخوانی کنیم و اینکار توسط [متد form آن انجام می‌شود](#).

نویسنده: molana11

تاریخ: ۱۵:۲۶ ۱۳۹۲/۰۱/۱۴

با سلام.

اطلاعات کنترلر من بصورت زیر است:

```
using MvcApplication3.Models;
...
namespace MvcApplication3.Controllers
{
    public class StudentController : Controller
    {
        public ActionResult Index()
        {
            //var data = new StudentsList();
            return View();
        }
        public ActionResult DataList()
        {
            var data = new StudentsList();
            return PartialView("Pv_DataList", data);
        }

        #region Edit
        [HttpGet]
        public ActionResult Edit(int? id)
        {
            var data = new StudentsList().FirstOrDefault(p => p.Id == id);
            return PartialView("Pv_Edit", data);
        }
        [HttpPost]
        [AjaxOnly]
        [ValidateAntiForgeryToken]
        public ActionResult Edit(StudentModel model)
        {
            Thread.Sleep(1000);
            if (this.ModelState.IsValid)
            {
                return Json("ok", JsonRequestBehavior.AllowGet);
            }
            return Json("error");
        }
        #endregion
    }
}
```

```

@using MvcApplication3.Models
@{
    ViewBag.Title = "Student Index";
}
<style>
    #div_StudentEditDialogContainer {
        padding: 15px;
        background-color: silver;
        border: 1px solid gray;
        -webkit-border-radius: 10px;
        -moz-border-radius: 10px;
        border-radius: 10px;
        display: none;
        position: absolute;
        -webkit-box-shadow: 0 1px 3px rgba(0,0,0,0.3);
        -moz-box-shadow: 0 1px 3px rgba(0,0,0,0.4);
        box-shadow: 0 1px 3px rgba(0,0,0,0.5);
    }
</style>
<h2>Student Index</h2>
<div id="div_StudentListViewContainer">
    @{ Html.RenderAction("DataList", "Student");}
</div>
<div id="div_StudentEditDialogContainer">
    @{ Html.RenderAction("Edit", "Student");}
</div>
<div id="div_StudentAddDialogContainer">
</div>
<div id="div_StudentRemoveDialogContainer">
</div>
<div id="div_StudentSearchDialogContainer">
</div>

@section JavaScript{
    <script type="text/javascript">
        $(document).ready(function () {
            $("#div_StudentListViewContainer table input[type='submit']").click(function (e) {
                //show edit dialog
                e.preventDefault();
                var id = $(this).parent().parent().attr('data-studentid');
                var url = '@Url.Action("Edit", "Student")';
                $.ajax({
                    type: "GET",
                    url: url,
                    data: { id: id },
                    beforeSend: function () {
                        //$(waitingPanel).css("display", "block");
                    },
                    success: function (html) {
                        if (html == "nodata") {
                            $("#div_StudentEditDialogContainer").html("دانشجویی با این مشخصات یافت نشد!");
                        } else {
                            $("#div_StudentEditDialogContainer").css("display", "block");
                            $("#div_StudentEditDialogContainer").html("").append(html);
                        }
                    },
                    complete: function () {
                        //$(waitingPanel).css("display", "none");
                    }
                });
            });
        });
    </script>
}

```

و یک دیالوگ برای ویرایش را بصورت داینامیک در صفحه ظاهر میکنم، اما اعتبارسنجی سمت کاربر برای آن کار نمیکند:

```

@using MvcApplication3.Models
@model StudentModel
@{
    var postUrl = Url.Action(actionName: "Edit", controllerName: "Student");
}

```

```

@using (Html.BeginForm(actionName: "Edit", controllerName: "Student",
                      method: FormMethod.Post,
                      htmlAttributes: new { id = "frm_studentEdit" }))
{
    @Html.ValidationSummary(true)
    @Html.AntiForgeryToken()

    <div class="editor-label">
        @Html.LabelFor(model => model.Id)
    </div>
    <div class="editor-field">
        @Html.EditorFor(model => model.Id)
        @Html.ValidationMessageFor(model => model.Id)
    </div>

    <div class="editor-label">
        @Html.LabelFor(model => model.Code)
    </div>
    <div class="editor-field">
        @Html.EditorFor(model => model.Code)
        @Html.ValidationMessageFor(model => model.Code)
    </div>

    <div class="editor-label">
        @Html.LabelFor(model => model.FullName)
    </div>
    <div class="editor-field">
        @Html.EditorFor(model => model.FullName)
        @Html.ValidationMessageFor(model => model.FullName)
    </div>

    <div class="editor-label">
        @Html.LabelFor(model => model.BirthDate)
    </div>
    <div class="editor-field">
        @Html.EditorFor(model => model.BirthDate)
        @Html.ValidationMessageFor(model => model.BirthDate)
    </div>

    <div class="editor-label">
        @Html.LabelFor(model => model.IsMale)
    </div>
    <div class="editor-field">
        @Html.EditorFor(model => model.IsMale)
        @Html.ValidationMessageFor(model => model.IsMale)
    </div>

    <p>
        <input type="submit" id="btn_Save" value="ذخیره" />
        <input type="submit" id="btn_Cancel" value="انصراف" />
    </p>
}

<script type="text/javascript">
$(function () {
    $("#btn_Save").click(function (e) {
        e.preventDefault();
        var button = $(this);
        $("#frm_studentEdit").PostMvcFormAjax({
            postUrl: '@postUrl',
            loginUrl: '/login',
            beforePostHandler: function () {
                // غیرفعال سازی دکمه ارسال
                button.attr('disabled', 'disabled');
                button.val("...");
            },
            completeHandler: function (data) {
                // فعال سازی مجدد دکمه ارسال
                button.removeAttr('disabled');
                button.val("ذخیره");
            },
            errorHandler: function () {
                alert('خطایی رخ داده است');
            }
        });
    });
    $("#btn_Cancel").click(function (e) {
        e.preventDefault();
        var button = $(this);
        $(".editDialog").parent("div").css("display", "none");
    });
});

```



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

با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۵:۴۸ ۱۳۹۲/۰۱/۱۴

برای تکمیل بحث مراجعه کنید [به قسمت 21](#) سری MVC؛ بحث «افزودن فرم‌ها به کمک jQuery.Ajax و فعال سازی اعتبار سنجی سمت کلاینت» آن.

نویسنده: molanall
تاریخ: ۱۰:۱۷ ۱۳۹۲/۰۱/۱۵

با سلام.

اگر بخواهیم به جای return Content از return json به عنوان خروجی استفاده کنیم و مثلاً مقدار "ok" را برگردانیم، در تابع completeHandler چطور این مقدار را دریافت و بررسی کنیم؟
باتشکر.

نویسنده: وحید نصیری
تاریخ: ۱۱:۵۴ ۱۳۹۲/۰۱/۱۵

نیاز به کمی تغییر دارد. اگر خروجی اکشن متد شما به این نحو باشد:

```
return json(new { result = "ok" });
```

در سمت اسکریپت در ajax.\$:

```
//...  
success: function (data) {  
    if (data) {  
        alert("json data: " + data.result);  
    }  
}
```

نویسنده: محمد آزاد
تاریخ: ۱۷:۳۲ ۱۳۹۲/۰۱/۱۷

جناب نصیری با تشکر از مطلبتون. ی سوال داشتم اینکه استفاده از فرم ایجکس (Ajax.BeginForm) موجود در Razor بهتر و راحت‌تر نیست؟ یا اینکه نسبت به jquery محدودیت خاصی داره؟

نویسنده: وحید نصیری
تاریخ: ۱۸:۷ ۱۳۹۲/۰۱/۱۷

- Ajax.BeginForm هم [در پشت صحنه](#) از jQuery Ajax استفاده می‌کند. فقط پیاده سازی نهائیش از شما مخفی شده.
- در نمونه بحث جاری کنترل بیشتری بر روی رویدادها خواهید داشت. مثلاً قسمت 403 xhr.status == و هدایت کاربر به صفحه لاگین در صورت منقضی شدن اعتبارسنجی آن.
- Ajax.BeginForm برای کار کردن حتماً نیاز به submit button داره. در مطلب جاری از یک span هم می‌تونید استفاده کنید و مشکلی نداره.
- در Ajax.BeginForm آنچنان کنترلی بر روی پردازش نهایی خروجی اکشن متد ندارید. مثلاً در اینجا عنوان شد که اگر خروجی JSON بود و اگر دارای فیلد مشخصی با مقدار مشخصی بود نیاز است کار خاصی انجام شود. در حالت jQuery Ajax مستقیم،

پردازش JSON ساده‌تر است.

نویسنده:

محمد آزاد

تاریخ:

۱۹:۱۹ ۱۳۹۲/۰۱/۱۷

- در نمونه بحث جاری کنترل بیشتری بر روی رویدادها خواهید داشت. مثلا قسمت `xhr.status == 403` و هدایت کاربر به صفحه لاگین در صورت منقضی شدن اعتبارسنجی آن.

با توجه به اینکه گفتید فرم ایجکس داره همین روال رو در پشت صحنه ایجاد می‌کنه برای هدایت کاربر به صفحه لاگین همیشه کد مربوطه رو تو `OnComplete` شی `AjaxOptions` نوشت؟

- `Ajax.BeginForm` برای کار کردن حتما نیاز به `submit button` داره. در مطلب جاری از یک `span` هم می‌تونید استفاده کنید و مشکلی نداره.

آیا همیشه این سبمیت رو با جاوااسکریپت پیاده سازی کرد؟ که بشه تو رویداد کلیک هر المنتی نوشت؟

- در `Ajax.BeginForm` آنچنان کنترلی بر روی پردازش نهایی خروجی اکشن متد ندارید. مثلا در اینجا عنوان شد که اگر خروجی JSON بود و اگر دارای فیلد مشخصی با مقدار مشخصی بود نیاز است کار خاصی انجام شود. در حالت `jQuery Ajax` مستقیم، پردازش JSON ساده‌تر است.

اینم همیشه نتیجه نهایی رو تو `OnComplete` شی `AjaxOptions` داشته باشیم؟

نویسنده:

وحید نصیری

تاریخ:

۲۱:۳۵ ۱۳۹۲/۰۱/۱۷

سورس فایل `jquery.unobtrusive-ajax.js` به همراه پروژه‌های MVC موجود است. می‌توانید موارد مدنظر رو در اون بررسی کنید، مثلا نحوه انتقال اطلاعات یا نحوه گوش فرا دادن به دکمه `submit` در آن. سورس باز هم هست. تغییرش بدید، بعد نتایج رو برای حذف کدهای تکراری کپسوله کنید (اصل بحث جاری).

نویسنده:

م کریمی

تاریخ:

۱۸:۴۰ ۱۳۹۲/۰۳/۲۱

با سلام خدمت آقای نصیری

با تشکر از مطلب کاربردی جاری، یک سوال داشتم چطور می‌شه با این روش `validation`‌های سمت سرور رو به کاربر نمایش بدهیم به طور مثال اگه در اینجا

```
[Required(ErrorMessage = "(*)"), DisplayName("نام")]
[MaxLength(3, ErrorMessage="لطفا بیشتر از 3 کاراکتر وارد ننمایید")]
public string Name { set; get; }
```

از `MaxLength` و یا یک سری `Att`‌های خاصی استفاده بشه چطور میشه پیغام "لطفا بیشتر از 3 کاراکتر وارد ننمایید" را به کاربر نشان داد، در حالت فعلی فقط پیغام خطایی رخ داده ظاهر می‌شه با تشکر

نویسنده:

وحید نصیری

تاریخ:

۲۰:۵۶ ۱۳۹۲/۰۳/۲۱

این افزونه خروجی ساده متنی داره. اگر نیاز به بازگرداندن اطلاعات بیشتر و ساختار یافته‌ای هست، باید خروجی JSON براس طراحی کنید و بعد در سمت `jQuery Ajax` این ساختار مدنظر رو پردازش کنید. مثلا ساختاری بر اساس خواصی مانند لیست خطاها، لیست پیام‌ها و وضعیت عملیات. بعد قسمت `complete` افزونه فوق باید کلا بازنویسی شود.