عنوان: **نحوه ایجاد یک اسلایدشو به صورت داینامیک**
نویسنده: پویا امینی
تاریخ: ۱۳۹۱/۰۴/۱۵ ۱۷:۲۹
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, jQuery, DynamicSlideShow

در این آموزش قصد دارم نحوه ایجاد یک Slideshow به صورت داینامیک را با ASP.NET طراحی کنم(منظور از ایجاد Slideshow به صورت داینامیک این است که عکس‌ها را به صورت داینامیک از DB بخواند).

اولین گام این است که Plugin مورد نظر را دریافت کنید که من از پلاگین Orbit استفاده کرده ام

ابتدا یک DataBase با نام DynamicSlideShow ایجاد و یک جدول با ساختار زیر با نام Picturesدرون آن ایجاد می‌کنیم

Id	int	☐
PicturePath	nvarchar(50)	☐
PictureText	nvarchar(200)	☑
		☐

گام بعدی ایجاد یک پروژه Asp.Net با زبان C# و اضافه کردن فایلهای پلاگین به پروژه و ایجاد یک Handler برای بازیابی داده‌ها از DB است. ساختار Solutionما باید به صورت زیر باشد

برای اینکه بتوانیم با DB ارتباط برقرار کنیم از EF استفاده می‌کنیم به همین منظور ابتدا یک Model به نام DynamicSlideShowModel ایجاد می‌کنیم

در گام بعد بر روی GenerateFromDatabase انتخاب کرده و بر روی دکمه Next کلیک می‌کنیم و در مراحل بعد ابتدا DB مربوط به مثال (DynamicSlideShow) و جدول آن را انتخاب می‌کنیم . حال یک Model به درون پوشه App_Code اضافه شده است

در ادامه برای واکشی رکوردهای موجود در جدول Pictures کدهای زیر را درون Handler می‌نویسیم

```
var ctx = new DynamicSlideShowEntities();
  var list = ctx.Pictures.ToList();
  string str = JsonConvert.SerializeObject(list);
  context.Response.Write(str);
```

در این کدها تنها نکته‌ای که احتیاج به توضیح دارد این است که ابتدا یک لیست از رکوردها را از جدول Pictures واکشی می‌کنیم و برای پاس دادن به کلاینت ما آن‌ها را به فرمت Json تبدیل کرده ایم که برای تبدیل از کتابخانه آماده Newtonsoft.Json.dll استفاده کرده ایم

حال باید با استفاده از jQuery کدهای درون Handler را اجرا کنیم؛ برای همین منظور یک صفحه با نام default.aspx ایجاد کرده و کدهای زیر را درون آن می‌نویسیم

```
<head runat="server">
    <title>Dynamic SlideShow</title>
    <link href="CSS/orbit-1.2.3.css" rel="stylesheet" type="text/css" />
    <script src="Script/jquery-1.7.1.min.js" type="text/javascript"></script>
    <script src="Script/jquery.orbit-1.2.3.min.js" type="text/javascript"></script>
    <script type="text/javascript">
        $(function () {
            $.ajax({
                url: "Handler.ashx",
                contentType: "application/json; charset=utf-8",
                success: function (data) {
                    $.each(data, function (i, b) {
                        var str = '<img src="' + b.PicturePath + '" alt="' + b.PictureText + '"/>';
                        $("#featured").append(str);
                    });
                    $('#featured').orbit();
                },
                dataType: "json"

            });
            });
    </script>
</head>
<body>
    <form id="form1" runat="server">
    <div id="featured">
    </div>
    </form>
</body>
```

در اینجا ابتدا با استفاده از متد ajax کتابخانه jQuery کدهای درون Handler را اجرا کرده و به ازای هر المان موجود در جدول یک تگ img به صفحه اضافه می‌کنیم.

نظرات خوانندگان

نویسنده: مسعود زیانی
تاریخ: ۱۳۹۱/۰۴/۱۷ ۱۰:۴۶

خیلی عالی بود ممنون

نویسنده: میثم
تاریخ: ۱۳۹۱/۰۴/۱۹ ۱۳:۳۱

مرسی خیلی عالی بود ، این عکس‌ها یکی یکی از سمت سرور اضافه می‌شوند یا همه با هم ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۹ ۱۳:۴۶

کاری که در اینجا انجام شده ساخت div زیر به صورت پویا است (بر اساس لیست تصاویر دریافت شده از سرور):

```
<div id="featured">
     <img src="img1.jpg" alt="img1" />
     <img src="img2.jpg" alt="img2" />
     <img src="img3.jpg" alt="img3" />
</div>
```

و سپس فراخوانی متد زیر روی آن:

```
$('#featured').orbit();
```

نویسنده: صادق
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۳:۱

با سلام

من کدهای بالا رو همه رو درست انجام دادم ولی در فایل handler متاسفانه Tolist(); نمیشه. به فایل ضمیمه دقت کنید. با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۳:۶

ذکر using System.Linq را فراموش کردید.

نویسنده: صادق
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۳:۲۷

ممنونم از شما استاد عزیز

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۱۴

یه سوال واسه من پیش اومد مهندس
ممکنه ما 100 تصویر داشته باشیم که بالطبع لود همه آنها یکباره درست نیست
آیا میشه کاری کرد که در هر درخواست که اسلایدشو تعویض میشه تصویر دینامیک لود کرد یا مثلا هر دفعه سه تصویر لود کرد و به پلاگین اضافه کرد تا نمایش بده؟
چون اینجوری همه تصاویر به یکباره لود شده و سرعت خیلی پایین میاد

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۲:۲۱

می‌تونید از ایده‌های افزونه‌هایی تحت عنوان « jQuery loading images on demand » استفاده کنید. مثلا: (^)

نویسنده: حمید خرازان
تاریخ: ۱۳۹۲/۰۱/۰۳ ۲۲:۵۷

سلام،مهندس جان من وقتی از این کد استفاده میکنم فقط رکورد اول رو واسم میاره و بقیه رو undefined میده، میدونید دلیلش چیه؟

نویسنده: مهدی آزادی
تاریخ: ۱۳۹۲/۰۱/۱۸ ۱۰:۳

عالی بود.

پاینده باشین.

نویسنده: پویا امینی
تاریخ: ۱۳۹۲/۰۱/۱۸ ۱۶:۳۴

دوست عزیز باید کدتون رو ببینم و نظر بدم.

نویسنده: Abolfazl
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۲:۲۸

با سلام ..
من همه چیز را همان طوری که شما گفتید انجام دادم ولی نمی‌دونم چرا اجرا نی کنه ..بعد از لود صفحه یک لحظه نشون میده بعد فقط فلش که برای حرکت تصویر است نشون میده و اون هم در گوشه بالای صفحه :
این هم کد :::

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;


public class GallerySite
{
    int iD;

    public int ID
```

```
    {
        get { return iD; }
        set { iD = value; }
    }
    string imagePath;

    public string ImagePath
    {
        get { return imagePath; }
        set { imagePath = value; }
    }
    string imageText;

    public string ImageText
    {
        get { return imageText; }
        set { imageText = value; }
    }
    public List<GallerySite> GetImage()
    {
        List<GallerySite> _Gallery = new List<GallerySite>() { };
        SqlConnection cnn = new SqlConnection("Data Source=.;Initial
Catalog=RoshanZamirDataBase;Integrated Security=True");
        cnn.Open();
        SqlCommand cmd = new SqlCommand("Select * From Gallery", cnn);
        SqlDataReader datareadfewr = cmd.ExecuteReader();
        if (datareadfewr.HasRows)
        {
            while (datareadfewr.Read())
            {
                _Gallery.Add(new GallerySite()
                {
                    ID = Convert.ToInt32(datareadfewr["ID"]),
                    ImagePath = (string)datareadfewr["ImagePath"],
                    ImageText=(string)datareadfewr["ImageText"]
                });
            }
        }
        return _Gallery;
    }
}

.................................................................................................
.........................



    <link href="orbit-1.2.3.css" rel="stylesheet" />
    <script src="jquery-1.8.3.min.js"></script>
    <script src="jquery.orbit-1.2.3.min.js"></script>
    <script type="text/javascript">
        $(function () {
            $.ajax({
                url: "Handler.ashx",
                contentType: "application/json; charset=utf-8",
                success: function (data) {
                    $.each(data, function (i, b) {
                        var str = '<img src="' + b.ImagePath + '" alt="' + b.ImageText + '"/>';
                        $("#featured").append(str);
                    });
                    $('#featured').orbit();
                },
                dataType: "json"
            });
        });
    </script>



.................................................................................................
...................

Handler :


public class Handler : IHttpHandler {

    public void ProcessRequest (HttpContext context) {
```

```
        var _Gallery = new GallerySite();
        var List = _Gallery.GetImage();
        string str = JsonConvert.SerializeObject(List);
        context.Response.Write(str);
    }
    public bool IsReusable {
        get {
            return false;
        }
    }
}
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۳:۱۴

با نحوه دیباگ اسکریپت‌های jQuery آشنا هستید؟ افزونه فایرباگ رو نصب کنید. بعد مراجعه کنید به قسمت مثلا network آن. بعد مشاهده کنید، آیا چیزی رد و بدل شده؟ آیا در کنسول اسکریپت‌های مرورگر خطایی لاگ شده. آیا مثلا CSSهای سایت روی این افزونه تاثیر داشته. مرورگر کروم هم در قسمت developer tools آن شبیه به فایرباگ رو داره. دکمه F12 آن.

نویسنده: Abolfazl
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۴:۵۳

من نتونستم درستش کنم ..یه راه دیگه وجود نداره :؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۷:۳۷

کد رو دیباگ کن:

```
var str = '<img src="' + b.ImagePath + '" alt="' + b.ImageText + '"/>';
```

ببین اینجا نهایتا چه رشته‌ای تشکیل میشه.
آیا src درستی به مرورگر ارائه شده؟ مثلا آیا c:\path\data\image\01.jpg است؟ یا اینکه به نحو صحیحی به صورت یک مسیر نسبی از ریشه سایت مانند siteimage/01.jpg مقدار دهی شده؟

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۸:۳۹

سلام؛ من از روش بالا در mvc استفاده کردم می‌خواستم ببینم آیا کار صحیحی انجام دادم یا خیر ؟
در مورد Http Hndler و Http Module هم اطلاعات کمی دارم و آیا استفاده از این‌ها در mvc کار صحیحی است یا خیر ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۹:۲

نیازی نیست. مطلب فوق در مورد وب فرم‌ها نوشته شده. برای انتقال آن به MVC بجای فایل ashx یک اکشن متد با خروجی Json خواهید داشت.

عنوان: jQuery Tips #1
نویسنده: پویا امینی
تاریخ: ۱۳۹۱/۰۴/۱۷ ۲۰:۵۴
آدرس: www.dotnettips.info
گروه‌ها: jQuery, jQuery-Tips

چگونگی تغییر سایز فونت صفحه با استفاده از jQuery

کد زیر را در نظر بگیرید

```
$(function () {
            // اندازه واقعی فونت
            var originalFontSize = $('#test').css('font-size');
            $(".resetFont").click(function () {
                $('#test').css('font-size', originalFontSize);
            });
            // افزایش اندازه فونت
            $(".increaseFont").click(function () {
                var currentFontSize = $('#test').css('font-size');
                var currentFontSizeNum = parseFloat(currentFontSize);
                var newFontSize = currentFontSizeNum + 5;
                $('#test').css('font-size', newFontSize);
                return false;
            });
            // کاهش اندازه فونت
            $(".decreaseFont").click(function () {
                var currentFontSize = $('#test').css('font-size');
                var currentFontSizeNum = parseFloat(currentFontSize);
                var newFontSize = currentFontSizeNum - 5;
                $('#test').css('font-size', newFontSize);
                return false;
            });
        });
```

و کد HTML زیر را

```
<div id="test">
        jQuery Tips By Mohsen Bahrzadeh
    </div>
    <a href="#">decreaseFont</a>
    <a href="#">Increase</a>
    <a href="#">resetFont</a>
```

در این کد ابتدا اندازه فونت را درون متغیر originalFontSize ذخیره و سپس 3 متد تعریف کرده‌ایم، که اولین متد اندازه فونت را به اندازه فونت اولیه بر می‌گرداند و در متد دوم می‌خواهیم اندازه فونت تگی با ID=test را افزایش دهیم.
برای این کار ابتدا اندازه جاری تگ را گرفته و درون متغیر currentFontSize قرار داده سپس مقدار متغیر currentFontSize را به Float تبدیل کرده و 5 واحد به آن اضافه کرده ایم و سپس عدد بدست آمده را به عنوان اندازه فونت جدید در نظر گرفته ایم.
برای متد سوم دقیقاً همین اتفاق می‌افتد فقط به جای + از - استفاده شده است

نظرات خوانندگان

نویسنده: ashi mashi
تاریخ: ۱۳۹۱/۰۴/۱۸ ۱۰:۵

با سلام..
کد html مربوطه یه مشکلی داره.اونم اینه که اسم کلاس هایی که برای تابع کلیک با jquery براشون نوشتید در تگ‌های A ست نشده است.
مثال :

```
<a class="decreaseFont" href="#">decreaseFont</a>
```

و برای لینک‌های دیگر نیز بایستی کلاس‌های مربوطه ست شود.

عنوان: jQuery Tips #2
نویسنده: پویا امینی
تاریخ: ۱۳۹۱/۰۴/۱۸ ۲۱:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, jQuery-Tips

چگونگی استفاده از Cookie در jQuery

در این پست قصد دارم نحوه‌ی کاربا Cookie را با استفاده از jQuery برسی کنم و در پست بعدی یک مثال عملی را برسی می‌کنیم.

همانطور که می‌دانید کوکی یکی از اشیاء بسیار مهم برای نگه داری داده‌ها در بحث وب می‌باشد که یک فایل متنی است که سمت Client ذخیره می‌شود. و ما زمانی که از کتابخانه jQuery استفاده می‌کنیم خیلی مهم است که بدانیم چگونه باید با Cookieها کار کرد.

برای کار با کوکی‌ها در jQuery باید از Plugin های موجود استفاده کرد . برای ایجاد یک Cookie ابتدا فایل jQuery و سپس این کتابخانه را به صفحه مورد نظر اضافه نموده و کد زیر را برای ایجاد یک کوکی می‌نویسیم

```
<script src="jquery-1.7.1.min.js" type="text/javascript"></script>
    <script src="jquery.cookie.js" type="text/javascript"></script>
    <script type="text/javascript">
        $(function () {
            $.cookie("TestCookie", "Test Cookie By Mohsen Bahrzadeh ");
        });
    </script>
```

در اینجا یک کوکی با نام TestCookie با مقدار Test Cookie By Mohsen Bahrzadeh ایجاد می‌کنیم
حال پروژه را اجرا می‌کنیم. و در تصویر زیر مشاهده می‌کنید که کوکی ما ایجاد شده است

یکی از آیتم‌های بسیار مهم در کوکی‌ها تعریف زمان انقضاء کوکی است برای ست کردن تاریخ از کد زیر استفاده می‌کنیم

```
$(function () {
            $.cookie("TestCookie", "Test Cookie By Mohsen Bahrzadeh ", { expires: 7 });
        });
```

و برای خواندن مقدار کوکی از کد زیر استفاده می‌کنیم

```
$(function () {

            alert($.cookie("TestCookie"));
        });
```

و برای حذف کوکی از کد زیر استفاده می‌کنیم

```
$(function () {

            $.cookie("TestCookie", null);
        });
```

Options
General
Tabs
Content
Applications
Privacy
Security
Sync
Advanced
Tracking
Tell websites I do not want to be
History
Firefox will:
Use custom settings fo
Always use private browsing
Remember my browsing
Remember download hi
Remember search and f
Accept cookies from sit
Accept third-party c
Keep until:
they expir
Clear history when Firef
Location Bar
When using the location bar, sugges
OK
Cancel
Help
Cookies
Search:
TestCookie
The following cookies match your search:
Site
Cookie Name
Name: <no cookie selected>
Content: <no cookie selected>
Domain: <no cookie selected>
Path: <no cookie selected>
Send For: <no cookie selected>
Expires: <no cookie selected>
Remove Cookies
Remove All Cookies
Close

عنوان: **فعال سازی قسمت ارسال فایل و تصویر ویرایشگر آنلاین RedActor در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۳ ۲۳:۲۰
آدرس: www.dotnettips.info
گروه‌ها: MVC, ASP.Net MVC, jQuery

در سایت جاری از ویرایشگر آنلاین RedActor استفاده شده و کار کردن با آن هم بسیار ساده است:
یک TextArea ساده را به صفحه اضافه کرده و این افزونه جی‌کوئری را بر روی آن اجرا می‌کنید. به این ترتیب TextArea به صورت خودکار تبدیل به یک ویرایشگر مطلوب خواهد شد. برای مثال:

```
@Html.TextAreaFor(model => model.ArticleBody, htmlAttributes: new { style = "width:98%; height:500px"
})

<script type="text/javascript">
$('#ArticleBody').redactor({
                autoformat: false,
                convertDivs: false
            });
</script>
```

اما فعال سازی قسمت ارسال فایل و تصویر همراه با آن چطور؟
یک سری مثال نوشته شده با PHP به همراه این ویرایشگر آنلاین هستند که برای ایده گرفتن بد نیستند (البته به این معنا نیست که این ویرایشگر نیازی به PHP دارد. تنها قسمت سمت سرور مثال‌های آن با PHP است). برای مثال اگر در PHP از دستور echo برای ارائه یک نتیجه نهایی به ویرایشگر RedActor استفاده شده، معادل آن در ASP.NET MVC مساوی return Content است.

```
<script type="text/javascript">
$('#ArticleBody').redactor({
                imageUpload: "@Url.Action(result: MVC.RedactorUpload.ImageUpload())",
                fileUpload: "@Url.Action(result: MVC.RedactorUpload.FileUpload())",
                linkFileUpload: "@Url.Action(result: MVC.RedactorUpload.FileLinkUpload())"
                , autoformat: false
                , convertDivs: false
            });
</script>
```

برای فعال سازی قسمت آپلود این ادیتور نیاز است پارامترهای fileUpload، imageUpload و linkFileUpload مقدار دهی شوند.
همانطور که ملاحظه می‌کنید از T4MVC برای م شخص سازی مسیرها استفاده شده. برای مثال
MVC.RedactorUpload.ImageUpload به این معنا است که در کنترلری به نام RedactorUpload، اکشن متدی به نام ImageUpload پذیرای ارسال فایل ادیتور خواهد بود و به همین ترتیب در مورد سایر پارامترها.
RedactorUploadController هم ساختار بسیار ساده‌ای دارد. برای مثال هر کدام از متدهای آپلود یاد شده یک چنین امضایی دارند:

```
[HttpPost]
public virtual ActionResult ImageUpload(HttpPostedFileBase file)
{
}
```

البته در مورد مسایل امنیتی آپلود هم پیشتر در سایت بحث شده است . برای مثال در اینجا استفاده از فیلتر زیر را فراموش نکنید:

```
[AllowUploadSpecialFilesOnly(".jpg,.gif,.png")]
```

در هر کدام از متدهای آپلود (به سه متد برای سه پارامتر یاد شده نیاز است)، ابتدا HttpPostedFileBase را در پوشه‌ایی که مدنظر دارید ذخیره کنید. سپس باید محتوایی را به RedActor بازگشت دهید و اصل کار یکپارچگی با ASP.NET MVC نیز در همینجا است:
در حالت imageUpload، محتوایی به شکل زیر باید بازگشت داده شود:

```
return Content("<img src='" + path + "' />");
```

در حالت fileUpload، پس از ذخیره سازی فایل در سرور، مسیر آن باید به نحو زیر بازگشت داده شود:

```
return Content("<a href=" + path + ">" + someName + "</a>");
```

و در حالت linkFileUpload فقط باید مسیر نهایی فایل ذخیره شده بر روی سرور را بازگشت دهید:

```
return Content(path);
```

همچنین باید دقت داشت که کار ارسال فایل به سرور توسط خود افزونه RedActor انجام می‌شود و نیازی به کدنویسی ندارد. فقط باید سمت سرور آن‌را به نحوی که عنوان شد مدیریت کنید. ابتدا فایل را در سرور ذخیره کنید. سپس باید یک محتوای رشته‌ای را به نحو یاد شده، ساخت و توسط return Content بازگشت داد.

پ.ن.
قسمتی از مطالب متن فوق در نگارش جدید این ویرایشگر به نحو زیر تغییر کرده است.

نظرات خوانندگان

نویسنده: امیرحسین
تاریخ: ۱۳۹۱/۰۵/۰۴ ۳:۹

آقای نصیری واقعا شرمنده کردی استاد!

واقعا انتظار نداشتم که به این سرعت پاسخ بدید.
یک دنیا ممنونم

نویسنده: امیرحسین
تاریخ: ۱۳۹۱/۰۵/۰۴ ۳:۳۹

توی این لینک و این هم در مورد تابع‌های سمت سرورش هست که با جمع شدن با این مطلب نیاز شما برطرف می‌شه.
البته این موضوع برای سی کی ادیتور نوشته شده که تفاوتی با این ادیتور نمی‌کنه.

نویسنده: Zahra
تاریخ: ۱۳۹۱/۰۵/۰۴ ۹:۵

سلام

آقای نصیری میشه طریقه استفاده از این ادیتور رو برای asp.net webform هم شرح بدهید؟

باتشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۰:۲۱

در وب فرم‌ها:
- یک TextBox را به صفحه اضافه کنید. TextMode آن باید MultiLine باشد تا تبدیل به TextArea شود. همچنین ClientID آن‌را هم مقدار دهی کنید تا بشود در jQuery ازش استفاده کرد.
- تمام توضیحات یکی است با این تفاوت که:
الف) return Content در اینجا می‌شود Response.Write
ب) بجای کنترلر شما از یک http handler می‌تونید استفاده کنید (فایل‌های ashx معروف)

```
public class Upload : IHttpHandler {
    public void ProcessRequest (HttpContext context) {
        HttpPostedFile uploads = context.Request.Files["upload"];
        //... save the file
        // return context.Response.Write(...)
        // and then  context.Response.End();
```

در اینجا context.Request.Files امکان دسترسی به فایل‌های آپلود شده را می‌دهد.
آن‌ها را ذخیره کنید. در آخر کار هم با context.Response.Write مواردی را که در مقاله فوق توضیح داده شد، بازگشت دهید.

نویسنده: ali ghomi
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۹:۱۳

سلام من در Action مقدار زیرو برمی‌گردونم ولی در RedActor نشون نمیده

ولی آپلود درست انجام میشه.

```
string fileName = Path.GetFileName(file.FileName);
string filePath = "~/App_Data/Uploads/Pic/" + file.FileName;
if (file.ContentLength > 0 && fileName != null)
{
string path = Path.Combine(Server.MapPath("~/App_Data/Uploads/Pic"), fileName);
file.SaveAs(path);
}

return Content("<img src='" + filePath + "' />");
```

```
return Content("<img src='" + path + "' />");
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۹:۵۵

درسته. علت این است که پوشه App_Data محافظت شده است و از طریق فراخوانی لینک آن به صورت مستقیم قابل دسترسی نیست.
اگر فایل‌ها رو در این پوشه ارسال می‌کنید نیاز هست مثلا یک کنترلر دیگر به نام file درست کنید و کار این کنترلر فایل، دریافت نام تصویر و ارسال آن به صورت return File به مرورگر کاربر باشد.
یا اینکه کلا از یک پوشه معمولی استفاده کنید.
+
ضمن اینکه کاراکتر ~ برای تگ معمولی img مفهومی ندارد و نباید از آن در پایان کار استفاده شود.

نویسنده: محمد آزاد
تاریخ: ۱۳۹۱/۰۵/۱۱ ۲:۳۷

جناب نصیری من بعد از اجرای خط

```
return Content("<img src='" + filename + "' />");

//filename =C:\Users\mta\Documents\Visual Studio 2010\Projects\IAUG\GIAU\Images\Actors.png
```

یک ارور جاوا اسکریپتی دارم از این خط:

```
e.match(/\{.*\}/)[0]
```

که مقدارش نال هست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۱ ۸:۱۷

filename یک مسیر نسبی است (نسبت به ریشه سایت) مثلا:
GIAU/Images/Actors.png
یا مثلا (یک مسیر مطلق):
http://mysite.com/GIAU/Images/Actors.png

نویسنده: محمد آزاد
تاریخ: ۱۳۹۱/۰۵/۱۱ ۲۳:۲۹

جناب نصیری من کدم رو اصلاح کردم .البته بازم همین مشکل رو دارم

```
virtualpath = "Images/Main.jpg";
  return Content("<img src='" + virtualpath + "' />");
```

ولی بازهم تو کتابخونه جاوا اسکریپت RedActor از همون خطی که گفتم ارور میگیره

```
var c=e.match(/\{.*\}/)[0];

//e=<IMG src="Images/Main.jpg">

//e.match=null !
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۱ ۲۳:۵۰

من این سطر **e.match** رو در سورس redactor (فایل **js**. redactor) پیدا نمی‌کنم. مطمئن هستید تداخلی در برنامه ندارید؟
ضمنا این ویرایشگر با نگارش‌های نهایی مرورگرها سازگاری خوبی دارد.

نویسنده: محمد آزاد
تاریخ: ۱۳۹۱/۰۵/۱۲ ۰:۱۵

بله درست گفتید اسم فایلش اینه

```
eval code[dynamic]
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۲ ۰:۱۸

من از نسخه معمولی redactor استفاده می‌کنم.
اگر نیازی به فشرده سازی آن بود از این روش استفاده کنید: (^)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۲ ۹:۴۲

یک نکته تکمیلی رو اینجا اضافه کنم.
نحوه عملکرد RedActor در نگارش جدید آن با نگارش قبلی که من استفاده کردم، کاملا تفاوت کرده و قسمت‌هایی از مطلب فوق دیگر معتبر نیست.
RedActor اینبار به خروجی json نیاز دارد. برای مثال به فایل مثال image_upload.php آن دقت کنید. در نگارش جدید به این نوع خروجی نیاز دارد:

```
// displaying file
$array = array(
'filelink' => '/images/'.$filename
);

echo stripslashes(json_encode($array));
```

معادل آن در ASP.NET MVC به نحو زیر است:

```
var array = new { filelink = "Images/k1.jpg" };
return Json(array, System.Net.Mime.MediaTypeNames.Text.Plain, JsonRequestBehavior.AllowGet);
```

یک anonymously typed object دلخواه را مطابق array مثال PHP آن درست کنید و به Json result به نحو فوق ارسال کنید.
برای مابقی هم به همین ترتیب است. مثلا file_upload.php مثال آن اینبار به شکل زیر تعریف شده:

```
$array = array(
'filelink' => '/files/'.$_FILES['file']['name'],
'filename' => $_FILES['file']['name']
);

echo stripslashes(json_encode($array));
```

معادل ASP.NET MVC آن به نحو زیر است:

```
var array = new { filelink = "files/myfile.zip", filename="myfile.zip" };
return Json(array, System.Net.Mime.MediaTypeNames.Text.Plain, JsonRequestBehavior.AllowGet);
```

نویسنده: امیرحسین
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۱:۵۹

سلام
لطفا این اصلاحات رو توی اصل متن اصلاح کنید
چون همه کامنت‌ها را تا انتها نمی‌خونند! و حدود دو روز مثل من علاف بشوند!

نویسنده: donya
تاریخ: ۱۳۹۱/۰۶/۱۵ ۱۵:۳۳

سلام
آقای نصیری من از ادیتور redactor در استفاده کردم، محتوای ادیتور رو در دیتابیس ذخیره می‌کنم اما برای نمایش دادنش مشکل دارم کل تگهای Html رو تو خروجی نشون میده.
در asp.net از literal برای نمایش محتویات ادیتور استفاده می‌کردیم ولی در MVC لیترال وجود نداره و نمیدونم چطوری محتوای ادیتور رو نمایش بدم ممنون میشم اگه جواب بدین.
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۵ ۱۵:۵۳

نکته‌اش استفاده از Html.Raw است که در قسمت پنجم سری MVC توضیح داده شده.

نویسنده: roya
تاریخ: ۱۳۹۱/۰۶/۱۶ ۱۵:۱

آقای نصیری من برای ذخیره یه متن با redactor به مشکل خوردم. Error ای به صورت زیر میدهد:

```
System.Web.HttpRequestValidationException: A potentially dangerous Request.Form value was detected from
the client (Body="<p>aaaaaaa...").
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۶ ۱۷:۱۴

مراجعه کنید به قسمت 17 سری MVC سایت؛ قسمت «بررسی اعتبار درخواست (Request Validation) در ASP.NET MVC».

نویسنده: احمد احمدی

تاریخ: ۱۳۹۱/۰۶/۲۱ ۱۵:۳۸

در صورت امکان در مورد وضعیت لایسنس این ادیتور کمی توضیح بدید .
بنده تصمیم دارم به دو صورت از این ادیتور استفاده کنم ، یکی در یک پروژه‌ی شخصی ، بار دیگر هم در یک پروژه‌ی رایگان و متن باز .
وضعیت لایسنس در هریک از شرایط بالا به چه صورت است ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۱ ۱۶:۱۰

مطابق توضیحات این صفحه ، اگر نیاز به پشتیبانی خاصی از تیم مربوطه داشته باشید، نیاز است هزینه کرده و مثلا در ایمیل خودتون Paypal transaction number را نیز ارسال کنید تا به شما پاسخ دهند.

نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۶/۲۷ ۱۳:۵۸

لطفا در مورد دکمه‌ی ارسال کد کمی توضیح بدید .
آیا از پلاگین خاصی باید استفاده کنیم یا باید بصورت دستی این امکان رو اضافه کنیم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۷ ۱۴:۶

نیاز هست افزونه نویسی کنید. استاندارد نیست.

نویسنده: کیارش سلیمان زاده
تاریخ: ۱۳۹۱/۱۰/۲۰ ۱۲:۴۷

به نظر من دیگه نسخه رایگان این ادیتور وجود نداره! اگه امکانش هست میشه نسخه ای که شما دارید و رایگانه برای دانلود بذارید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۲۰ ۱۳:۱۱

آخرین نسخه‌ای که من دارم: redactor803.zip

نویسنده: کیارش سلیمان زاده
تاریخ: ۱۳۹۱/۱۰/۲۰ ۱۳:۱۵

ممنون وحید جان،
اینم هستش.

نویسنده: Mohsen
تاریخ: ۱۳۹۱/۱۱/۱۱ ۲۳:۲۹

با سلام
زمانی که از این ادیتور استفاده می‌کنیم آیا کاربر می‌تواند فونت را تغییر دهد؟ ممنونم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۱۱ ۲۳:۴۲

خیر. احتمالا به این علت که آنچنان مرسوم نیست از چندین قلم در وب استفاده شود، منهای CSS اصلی سایت و تعریف قلم‌های اصلی برای هدرها و امثال آن. به علاوه قلم متن نمایش داده شده در صفحات باید تابع CSS سایت باشد نه ادیتور آن.

ولی در کل می‌تونید براش افزونه بنویسید تا اینکار را انجام دهد.

نویسنده: Mohsen
تاریخ: ۱۳۹۱/۱۱/۱۱ ۲۳:۵۲

من هر کاری کردم که فونت پیشفرض اون رو تغییر دهم نتونستم این کار رو انجام بدم حتی تمام propertyهای مربوط به فونت رو از فایل Css حذف کردم ولی باز فونت خودش رو می‌گیره فکر کردم که شاید این کار امکان پذیر نباشد ولی شما این کار رو کردید . امکانش هست بفرمایید چگونه فونت پیشفرض این ادیتور رو تغییر دادید. ممنونم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۱۲ ۰:۶

من چیزی رو از فایل css آن حذف نکردم. فقط یک tahoma به اول تعاریف فونت‌های آن اضافه کردم.
به علاوه باید دقت داشت که تمام این ادیتورها در یک iframe بارگذاری می‌شوند. یعنی در حین نمایش، body آن‌ها تابع css خودشون است و نه css سایت. به همین جهت نیاز به کمی تغییر دارند:

```
body{
font-family:Tahoma,sans-serif;
font-size:9pt;
margin:0;padding:0;
overflow-x:hidden;
background:#fff;
direction:rtl
}
```

نویسنده: ناصر
تاریخ: ۱۳۹۱/۱۲/۲۰ ۱۹:۰

ببخشید آقای نصیری این نسخه ای که شما گذاشتید (این فایل zip) آیا خروجی json میخاد یا خروجی باید مثل مثال بالا باشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۰ ۱۹:۹

روش تشخیص رو اینجا توضیح دادم. نیاز به json داره.

نویسنده: ناصر
تاریخ: ۱۳۹۱/۱۲/۲۰ ۱۹:۵۳

با عرض پوزش من در یک برنامه asp.net web form دارم از این ادیتور استفاده میکنم و نتونستم معادل دستورات زیر در asp.net رو بفهمم .

```
var array = new { filelink = "files/myfile.zip", filename="myfile.zip" };
return Json(array, System.Net.Mime.MediaTypeNames.Text.Plain, JsonRequestBehavior.AllowGet);
```

چیزی که خودم حدس زدم یک خروجی شبیه زیر بود :) اما مثل اینکه درست نبود

```
context.Response.Write("{'filelink'='" + filePath + "'}");
```

لطفا برای هر دو مورد فایل و تصویر راهنمایی بفرمایید چه خروجی json مدنظر هست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۰ ۲۱:۴۶

مراجعه کنید به مطلب آشنایی با کلاس JavaScriptSerializer

نویسنده: ناصر
تاریخ: ۱۳۹۱/۱۲/۲۱ ۲۲:۴۶

با تشکر فراوان از لطفتون. همچنین یک راهنمای جالب برای این سایت اینجاست. دوستانی که میخوان کمی تغییرات در این ادیتور بدن مثلا زبان فارسی اضافه کنن یا یک سری امکانات این چنینی حتما آدرس زیر رو ببین. چیزهای جالبی داخلش هست.
/http://imperavi.com/redactor/docs

همچنین داخل این راهنما خروجی هم دقیقا آئرده شده که من بعد از سوال پیدا کردم. خروجی برای image باید به شکل زیر باشه

```
{ "filelink": "/images/img.jpg" }
```

و برای فایل هم به شکل زیر باشه

```
{ "filelink": "/files/file.doc", "filename": "Filename" }
```

چیزی که خیلی وقت من رو گرفت اون اسلش قبل نام فایل بود که نمیزاشتم و خروجی مد نظر رو بم نمیداد. دوستان این قضیه رو حتما مد نظر داشته باشن تا مثل من وقتشون زیاد گرفته نشه :)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۱ ۲۲:۵۹

یک نکته: سعی نکنید این فرمت‌ها رو دستی درست کنید. علت این است که کلاس JavaScriptSerializer ذکر شده، زمانیکه var array ساخته شده رو دریافت می‌کنه، یک سری از کاراکترها رو هم escape می‌کنه .

نویسنده: بهروز
تاریخ: ۱۳۹۱/۱۲/۲۲ ۱۳:۶

آقای نصیری ظاهرا آخرین ورژن (redactor803) که شما گذاشتید با آخرین ورژن کنونی جیکوری (jquery-1.9.1.min.js) سازگاری ندارد و با زدن روی دکمه image پنجره ای برای ارسال باز نمی‌شود ولی با jquery-1.8.1.min.js تست کردم مشکلی نبود
آیا راهی برای حل این مشکل وجود دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۲ ۱۳:۱۳

- من دقیقا از آخرین نسخه رایگان این ادیتور استفاده می‌کنم. نیازی هم به نگارش‌های بعدی آن ندارم، چون نگارش مورد استفاده بدون مشکل کار می‌کند.
- همچنین از jquery-1.8.3.min.js برای کار با این نسخه استفاده می‌کنم.
- اگر نیاز به نگارش دیگری دارید بهتر است در انجمن تهیه کنندگان آن این مسایل رو مطرح کنید. البته ابتدا باید هزینه لایسنس نگارش‌های جدید آن‌را پرداخت کنید.
- بسیاری از افزونه‌های jQuery، با نگارش‌های جدید بعد از 1.9 آن سازگار نیستند و فقط این یک مورد نیست. بهتر است عجله نکنید و حداقل 6 ماهی برای ارتقاء صبر کنید.
- پروژه‌ای وجود دارد به نام jQuery Migrate برای پوشش مواردی که از جی‌کوئری 1.9 به بعد حذف شدن. این مورد رو باید به پروژه اضافه کنید تا با افزونه‌های قدیمی بتونید کار کنید.

نویسنده: میهمان
تاریخ: ۱۳۹۱/۱۲/۲۲ ۱۳:۳۳

سلام جناب آقای مهندس
در لینکی که در اول پست گذاشته اید ، وقتی دکمه Get Redactor رو میزنیم ، هیچ نسخه free وجود ندارد!
3 تا گزینه buy a licence وجود دارد و یک گزینه trial version
ممنون میشم اگر لینک مستقیم نسخه رایگان رو بگذارید
با سپاس

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۲ ۱۳:۳۹

کمی بالاتر در بین نظرات ارسال شده.

نویسنده: مولانا
تاریخ: ۱۳۹۲/۰۲/۱۷ ۲۱:۵۴

با سلام.
من خاصیتی از ویومدل را بصورت زیر تعریف کردم:

```
[Required(ErrorMessageResourceType = typeof(ValidationErrorsResource),
            ErrorMessageResourceName = ResourceKeys.ValidationErrorsResource.RequiredField)]
        [MaxLength]
        [DisplayName("متن")]
        [AllowHtml]
        public string FullDescription { get; set; }
```

در ویو هم بصورت زیر:

```
<div>متن: </div>
                <div>@Html.TextAreaFor(p => p.FullDescription)</div>
                <div>@Html.ValidationMessageFor(p => p.FullDescription)</div>

$('#FullDescription').redactor({
            autoformat: false,
            convertDivs: false
        });
```

ولی وقتی کاربر حتی چیزی وارد نمیکند در دیتابیس مقدار "
 " ذخیره می‌شود و همین باعث میشود Required کار نکند و پیغامی به کاربر نمایش نمی‌دهد.
ایراد کارم از کجاست؟ باتشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۷ ۲۲:۴۴

اینطور نیست. مقداری که عنوان کردید به معنای new line وارد شده است که در مرورگرهای مختلف مقدار بومی آن متفاوت است.

نویسنده: مولانا
تاریخ: ۱۳۹۲/۰۲/۱۸ ۶:۳

با سلام
من پوشه ای بنام upload درست کردم و کد مربوط به کنترل من هم بصورت زیر است:

```
[HttpPost]
        [AllowUploadImagesOnly(".jpg,.gif,.png")]
```

```
        public virtual ActionResult ImageUpload(HttpPostedFileBase file)
        {
             var newFileName = Server.MapPath(Path.Combine(Links.SiteContents.Upload.Url(),
file.FileName));
            file.SaveAs(newFileName);
            var array = new { filelink = newFileName };
            return Json(array, MediaTypeNames.Text.Plain, JsonRequestBehavior.AllowGet);
        }
```

فایل به درستی درون پوشه آپلود قرار میگیرد ولی درون ادیتور نمایش داده نمی‌شود. علت چیست؟
خطایی هم دریافت نمیکنم.
باتشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۸ ۸:۵۴

علت این است که نباید مستقیما newFileName را بازگشت دهید. newFileName در اینجا چیزی است مثل c:\path\folder\site\images\img01.png. یعنی مسیر مطلق است که باید تبدیل به مسیر نسبی images\img01.png شود برای استفاده در src تگ img.

نویسنده: افشار محبی
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۲:۱۰

اگر نخواهیم از نسخه‌های قدیمی‌تر (که رایگان بودند) استفاده کنیم و همیشه به روز باشیم باید دست به دامان نسخه‌های کرک یا نسخه‌هایی شویم که از این طرف و اون طرف به صورت غیر قانونی برای دانلود گذاشته می‌شوند.

اگر این موضوع درست باشد پس نمی‌توان از RedActor به صورت به روز و بی‌دردسر استفاده کرد. در نتیجه شاید بهتر باشد از رقبای رایگان مثل CKEditor استفاده شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۲:۲۲

سایت جاری از این ادیتور (نگارش قدیمی آن) استفاده می‌کند و به شخصه مشکلی با کیفیت آن ندارم و پاسخگوی نیاز کاری ما است.

نویسنده: افشار محبی
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۲:۳۵

آن طور که گفته می‌شود RedActor خیلی بهتر از CKEditor است. در همین سایت هم بدون مشکل در حال کار است. اما امکان به روز رسانی همیشگی باعث می‌شود خیال آدم از آینده راحت‌تر باشد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۲:۳۸

- هر طور صلاح می‌دونید.
- هدف از این بحث صرفا به اشتراک گذاشتن یک سری نکته بود که شاید هرجایی به این سادگی یافت نشود.

نویسنده: افشار محبی
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۲:۳۹

بله. مطمئناً بحث مفیدی بوده است. علاوه بر من، خیلی‌های دیگر استفاده کرده‌اند.

نویسنده: محمدرضا برنتی
تاریخ: ۱۳۹۲/۰۹/۱۴ ۱۲:۳۴

سلام آقای نصیری
امکان داره توضیح بدید چگونه دکمه Code و به این ادیتور اضافه کنیم ؟!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۱۴ ۱۳:۰

باید افزونه بنویسید. فایل paste_code.html آن در مسیر plugins:

```
<label>
    Code:</label>
<textarea id="redactor_insert_code_area" name="redactor_insert_code_area" style="height: 211px; width:
538px;" />
<label>
    Language:</label>
<select id="redactor_insert_code_lang">
    <option>CSharp</option>
    <option>VB</option>
    <option>JScript</option>
    <option>Sql</option>
    <option>XML</option>
    <option>CSS</option>
    <option>Java</option>
    <option>Delphi</option>
</select>
<br />
<input type="button" name="insert" id="redactor_insert_btn" value="%RLANG.insert%" />
```

و قسمت فعال سازی آن در فایل redactor.js ذیل setColorNone

```
showCodesPage: function () {
            this.modalInit('Insert Code', this.opts.path + '/plugins/paste_code.html', 600,
$.proxy(function () {
                var sel = this.getSelection();
                var currentCode = '';

                this.opts.codeElement = false;
                if ($.browser.msie) {
                    var parent = this.getParentNode();
                    if (parent.nodeName === 'PRE') {
                        this.opts.codeElement = parent;
                        currentCode = $(parent).text();
                    } else {
                        if (this.oldIE()) {
                            currentCode = sel.text;
                        } else {
                            currentCode = sel.toString();
                        }
                    }
                } else {
                    if (sel && sel.anchorNode && sel.anchorNode.parentNode.tagName === 'PRE') {
                        this.opts.codeElement = sel.anchorNode.parentNode;
                        currentCode = $(sel.anchorNode.parentNode).text();
                    } else {
                        currentCode = sel.toString();
                    }
                }

                if (this.opts.codeElement) {
                    $("#redactor_insert_btn").val("Update");
                }

                if (currentCode) $('#redactor_insert_code_area').val(currentCode);


                $('#redactor_insert_code_area').focus();
                $('#redactor_insert_btn').click($.proxy(this.insertCodesPage, this));

            }, this));
        },
        insertCodesPage: function () {
```

```
            var lang = $("#redactor_insert_code_lang").val();
            var code = $("#redactor_insert_code_area").val();
            code = code.replace(/\s+$/, ""); //rtrim;
            code = $('<span/>').text(code).html(); // encode

            this.$editor.focus();

            var preBlock;
            if (this.opts.codeElement) {
                preBlock = $(this.getParentNode());
            } else {
                preBlock = $("<pre/>");
            }
            preBlock.replaceWith("");

            var htmlCode = "<pre language='" + lang + "' name='code'>" + code + "</pre></div>";
            var codeBlock = "<div align='left' dir='ltr'>" + htmlCode + "</div><br/>";
            this.execCommand('inserthtml', codeBlock);

            this.modalClose();
        },
```

و بعد ثبت آن در فایل‌های public.js و default.js ذیل دکمه justify

```
code:
{
    title: 'Code',
    func: 'showCodesPage'
},
```

به css آن هم باید یک سطر ذیل را اضافه کنید:

```
body .redactor_toolbar li a.redactor_btn_code span          { background: url(../img/code_red.png) no-
repeat center; }
```

نویسنده: محمدرضا برنتی
تاریخ: ۱۳۹۲/۰۹/۱۴ ۱۹:۱۳

در این خط :

```
this.modalInit('Insert Code', this.opts.path + '/plugins/paste_code.html', 600, $.proxy(function () {
```

مقدار

```
this.opts.path
```

برابر undefined هست و در نتیجه بعد از کلیک روی دکمه code فقط تب خالی نمایش داده می‌شه !
و متاسفانه از این مرحله بیشتر نتونستم پیش برم ...

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۱۴ ۲۰:۰

ممکنه. چون نگارش‌های مختلفی ازش وجود داره و من از آخرین نگارش آن استفاده نمی‌کنم. برای مثال در نگارش‌های جدیدتر دنبال modal_table بگردید تا نحوه تعریف صفحه نمایش اطلاعات را به صورت توکار در همان فایل js نهایی، مشاهده کنید. کدهای ارسال شده با مقداری تغییر در نگارش‌های بعدی قابل استفاده هستند.

نویسنده: محمدرضا برنتی

تاریخ: ۱۳۹۲/۰۹/۱۵ ۱:۳۱

همین الان کارم تموم شد و تونستم دکمه code و با نگارش جدید این ادیتور هماهنگ کنم و در راه حل مشکلم تقریبا یاد گرفتم چجوری باید برای این ادیتور افزونه بنویسم ... و این بدون کمک شما امکان پذیر نبود ...
خیلی ممنونم ...

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۱۵ ۱۰:۲۳

اگر دوست داشتید فایل نهایی خودتون رو پیوست کنید. ممنون.

نویسنده: پرویز
تاریخ: ۱۳۹۲/۰۹/۲۷ ۲۲:۲۷

سلام؛ این آموزش را به صورت زیر انجام دادم. فایل آپلود میشه اما تو ادیتور نمایش داده نمیشه. اگه دوستان لطف کنند راهنمایی کنند ممنون میشم. در ضمن از نسخه ای که تو بخش نظرات ارسال شده، استفاده کردم. با تشکر.

کدها کنترلر

```
public ActionResult FileUpload(HttpPostedFileBase file)
            {

                if (file.ContentLength > 0)
                {
                  string   filePath = Path.Combine(HttpContext.Server.MapPath("../uploade/"),
Path.GetFileName(file.FileName));
                     file.SaveAs(filePath);
                }
              return Content("<img src='/uploade/" + file.FileName + "' />");
            }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۷ ۲۲:۳۲

- نسخه جدیدتر return Json دارد، بجای return Content. کمی بالاتر بحث شده.
+ مسیر نسبی هم باید برای تصاویر بازگشت داده شود.

نویسنده: فرزین بیاتی
تاریخ: ۱۳۹۲/۱۲/۰۱ ۱۶:۵۰

با سلام؛ فایل های public.js و default.js کجا قرار دارن؟ چون توی اون ورژنی که خودتون واسه دانلود گذاشتید و گفتید که از اون استفاده میکنید، من فقط فایل redactor.js رو پیدا کردم!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۱ ۱۶:۵۳

در نگارش جدیدتر آن این فایل‌ها با همان فایل اصلی ادغام و یکی شده‌اند.

نویسنده: فرزین بیاتی
تاریخ: ۱۳۹۲/۱۲/۰۱ ۱۷:۱۷

ممنون از پاسختون
من از ورژن v8.0.3 استفاده میکنم
میشه لطفاً بیشتر توضیح بدید که مثلاً چه جوری از فایل paste_code.html باید استفاده کرد یا اینکه code:

```
{
```

```
    title: 'Code',
    func: 'showCodesPage'
},
```

رو باید کجا اضافه کرد؟ تو سایت خودش واسه نوشتن پلاگین از یه روش دیگه استفاده کرده

نویسنده: فرزین بیاتی
تاریخ: ۱۳۹۲/۱۲/۰۴ ۱۲:۵۶

سلام میشه لطف کنید مراحل اضافه کردن دکمه‌ی کد رو بفرمایید. با اون روشی که آقای نصیری توضیح داده بودن من نتونستم انجامش بدم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۴ ۱۳:۵۹

یک مثال کامل برای ارسال کد؛ کار آقای احمدی هست: RedActor-InsertCode.rar
کدهای آن‌را با کدهایی که کمی بالاتر ارسال کردم، تطابق داده و تکمیل کنید.

نویسنده: فرزین بیاتی
تاریخ: ۱۳۹۲/۱۲/۰۶ ۱۴:۳۷

دوستان فرق linkFileUpload و fileUpload چیه و با کدوم دکمه فعال میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۶ ۱۷:۳

دکمه insert link چند برگه‌ی مختلف دارد که یکی از آن‌ها مربوط به linkFileUpload است. fileUpload مستقلا دکمه‌ی سنجاق مانندی در نوار ابزار دارد.

نویسنده: فرزین بیاتی
تاریخ: ۱۳۹۲/۱۲/۱۴ ۱۵:۳۳

چطوری میشه redactor رو readonly کرد؟ من readonly و disabled رو به TextArea اعمال کردم که جواب نداد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۴ ۱۶:۱۳

برای نمایش فقط خواندنی یک متن HTML از Html.Raw استفاده کنید؛ مانند نمایش متن یک مقاله در یک سایت.

نویسنده: جوادنبی
تاریخ: ۱۳۹۲/۱۲/۲۵ ۱۸:۵۳

این کد من که نوشتمش و جواب گرفتم.

```
[HttpPost]
public ActionResult ImageUpload(HttpPostedFileBase file)
{
   string path = "";
   var FileName = "";
   if (file.ContentLength > 0)
   {
     FileName = Path.GetFileName(file.FileName);
     string Extension = file.ContentType;
     string[] ExtensionList = { "image/jpg", "image/jpeg", "image/gif", "image/pmg" };
     if (ExtensionList.Contains(Extension.ToLower()))
     {
```

```
            path = Path.Combine(Server.MapPath("~/Images/uploads"), FileName);
            file.SaveAs(path);
        }
    }

    var array = new { filelink = @"Images\uploads\" + FileName };
    return Json(array, System.Net.Mime.MediaTypeNames.Text.Plain, JsonRequestBehavior.AllowGet);
}
```

امید وارم به کارت بیاد.

نویسنده: پویا امینی
تاریخ: ۱۳۹۲/۱۲/۲۶ ۲۲:۲۲

با سلام ، من می‌خواستم قسمت ارسال فایل را همانند شما فعال کنم مطابق کدی که فرمودید تنظیمات رو انجام دادم ولی هنگام انتخاب عکس از قسمت زیر

```
if (this.uploadOptions.success)
{
if (typeof d !== 'undefined')
{
// Remove bizarre <pre> tag wrappers around our json data:
var rawString = d.body.innerHTML;
var jsonString = rawString.match(/\{.*\}/)[0];
this.uploadOptions.success(jsonString);
}
```

خط

```
var jsonString = rawString.match(/\{.*\}/)[0];
```

خطای زیر رو مگیره

```
0x800a138f - JavaScript runtime error: Unable to get property '0' of undefined or null reference
```

ممنون میشم راهنمایی کنید

نویسنده: جوادنبی
تاریخ: ۱۳۹۲/۱۲/۲۷ ۲۳:۱۰

با سلام؛ من یک ویو دارم که در آن از دوتا redactor استفاده کردم. در واقع سناریوی من به این شکل است. یکی از این redactor برای ورود خلاصه‌ی متن است و در زیر آن یک لینک ایجکسی وجود دارد که میره و یک partial را می‌یاره. توی partial اونredactor دومی قرار دارد و جالب وقتی که من یک فایلی را در red actor که در پارشال هست آپلود می‌کنم فایل آپلود می‌شود ولی هنگامی که در صفحه‌ی سایتم می‌خوام دانلود کنم Error زیر را می‌دهد.
.The resource you are looking for has been removed, had its name changed, or is temporarily unavailable
حالا باید چه کار کنم؟ ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۷ ۲۳:۳۶

لینک تولید شده نهایی در اکشن متد را با لینک واقعی فایل بر روی سرور تطابق دهید.

نویسنده: پرویز
تاریخ: ۱۳۹۳/۱۰/۱۶ ۱۲:۳۴

سلام
فرض کنید یه تعداد عکس را با این ویرایشگر آپلود کردیم ، بعد یه تعداد از عکس رو از درون ویرایشگر حذف کردیم، چون از درون ویرایشگر مسیر عکس‌ها حذف میشه خود عکس رو هاست باقی می‌مونه.
سوال: چطوری میشه با حذف عکس از درون ویرایشگر خود عکس هم از روی هاست پاک بشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۶ ۱۲:۳۸

چنین قابلیتی را ندارد. برای این مورد از Froala WYSIWYG Editor استفاده کنید (^).

عنوان: **بررسی سایز و پسوند فایل آپلود شده قبل از ارسال به سرور**
نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۱۲ ۱۳:۴۳
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, Validation

می توان قبل از اینکه کاربر فایلی را سمت سرور ارسال کند پسوند فایل را چک کرد و از یک رفت و برگشت بیهوده به سرور جلوگیری کرد .

یک پیاده سازی ساده به کمک jQuery :

```
var ext = $('#my_file_field').val().split('.').pop().toLowerCase();
if($.inArray(ext, ['gif','png','jpg','jpeg']) == -1) {
    alert('invalid extension!');
}
```

در کد بالا ابتدا پسوند فایل انتخاب شده توسط کاربر پیدا می‌شود ، سپس به کمک متد inArray با آرایه ای از پسوندهای دلخواه ما مقایسه می‌شود و در صورت معتبر نبودن پیغامی به کاربر نشان می‌دهد.

کد بالا را می‌توان در رویداد Change کنترل فایل یا در هنگام Post شدن Form اطلاعات قرار داد.
کاملا واضح است که این روش را می‌توان به راحتی دور زد و نباید به آن اکتفا کرد.
مثال این روش را اینجا بررسی کنید.

بررسی سایز فایل :

در اکثر برنامه‌های تحت وب کاربرها محدود به Upload فایل تا سایز خاصی هستند ، این مورد را هم می‌توان قبل از Upload کنترل و اعتبارسنجی کرد :

```
//binds to onchange event of your input field
$('#myFile').bind('change', function() {

  //this.files[0].size gets the size of your file.
  alert(this.files[0].size);

});
```

این روش تا زمانی که کاربر JavaScript را غیر فعال نکرده قابل اطمینان است.

نظرات خوانندگان

نویسنده: پژمان
تاریخ: ۱۳۹۱/۰۵/۱۲ ۱۸:۱۱

سلام؛ با تشکر.

اگر در همینجا بخواهیم مانع ارسال فایل بشویم چکار باید کرد؟ مثلا الان فقط یک پیغام نمایش می‌دهد، اما مانع ارسال نمی‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۲ ۲۳:۴۸

می‌تونید بجای alert قابلیت کلیک رو از یک دکمه (ارسال فایل به سرور) بگیرید:

```
$("#btn1").unbind("click");
```

نویسنده: شهروز
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۵:۵۸

سلام
ببخشید قسمت دوم که نمایش سایز فایل است را من هر کاری کردم نتونستم انجام بدم.
یه HTML ساده ساختم و کدهای شما رو روش قرار دادم ولی هیچ اتفاقی نیافتاد.
البته اینم بگم که رو سایت jsfiddle جواب میده و مشکلی نیست ، می‌خواهم بدونم که به جز jQuery Library آیا چیز دیگری باید اضافه کنم ؟
ممنون میشم اگه فایل تستی هم قرار بدید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۶:۲۰

این رو با فایرفاکس تست کردم جواب داد ولی با IE به نظر کار نمی‌کنه.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۶:۴۳

HTML 5 File API در IE 9 پشتیبانی نمیشه ، اما انگار تا حدودی در IE 10 پشتیبانی میشه .
فکر می‌کنم یک راه برای پشتیبانی در IE استفاده از ActiveXObject هست ، اما خب مشکلی که هست به صورت پیشفرض غیر فعال هست. این در صورت فعال بودن ActiveX باید کار کنه.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۶:۴۵

@شهروز سلام ،
اگر از IE استفاده می‌کنید رجوع کنید به پاسخی که زیر کامنت آقای نصیری نوشتم.
اگر نه به کمک FireBug در FireFox متن خطا را پیدا کنید.
آخر وقت امروز یک مثال در انتهای پست اضافه می‌کنم.

نویسنده: شهروز
تاریخ: ۱۳۹۱/۰۵/۱۶ ۹:۵۴

سلام
بله درسته من با IE چک کرده بودم ، بعدش با Chrome چک کردم درست بود پس فکر کنم IE هنوز ساپورتش نمی‌کنه
ممنون

عنوان: **Navigation در AJAX - لینک دائمی برای محتوای AJAX**
نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۱۲ ۱۵:۰
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, Ajax

استفاده از AJAX به ما امکان می‌دهد قسمتی از صفحه را بدون رفتن به صفحه‌ی جدید بروز کنیم.
فرض کنید لیستی از اسامی و قیمت کالاها را در اختیار داریم ، کاربر روی دکمه‌ی جزییات کالا کلیک می‌کند ، جزییات کالا را با یک درخواست AJAX بارگزاری می‌کنیم و در یک Dialog به کاربر نمایش می‌دهیم .
آیا لینک دائمی برای این محتوای لود شده وجود دارد ؟ منظور این است آیا کاربر می‌تواند بدون تکرار مراحل قبلی و با استفاده از یک لینک جزییات آن کالا را مشاهده کند ؟
برای فراهم ساختن یک لینک دائمی برای محتوای AJAX راه حل استفاده از windows.location.hash هست .
در این http://example.com/blah #456 مقدار HashTag ما برابر با 456 می‌باشد.

مقدار HashTag به سرور ارسال نمی‌شود و فقط سمت کلاینت قابل استفاده هستند.

```
<span class='button goodDetail' id='123' >جزییات کالا</span>
```

به Span بالا یک Attribute سفارشی افزوده شده که حاوی کلید اصلی کالا می‌باشد. هنگامی که روی این دکمه کلیک می‌شود با یک درخواست AJAX اطلاعات جزییات این کالا واکشی می‌شود و قسمتی از صفحه بروزسانی می‌شود :

```
$('.goodDetail').click(function(){
//add to hash data that you need to make the AJAX request later
$(window).location.hash = $(this).attr('id');
$.ajax({
 type: "POST",
 url: 'some url',
 dataType: "html",
 data:'GoodId='+$(this).attr('id'),
 success: function(html) {
                //do something
 } })  })
```

در خط سوم کد بالا Location hash را به کلید اصلی کالایی که روی آن کلیک شده است مقدار داده ایم. بعد از کلیک روی آن دکمه URL ما اینگونه خواهد بود : www.mysite.com/goods.aspx#123
123 همان کلید اصلی کالا است که در Attribute سفارشی در دکمه‌ی جزییات کالا نگهداری می‌شده ، پس انتظار می‌رود اگر کاربر www.mysite.com/goods.aspx#123 را وارد کرد همان درخواست AJAX اجرا شود و جزییات کالای 123 به کاربر نشان داده شود.
در Document Ready صفحه‌ی جزییات کالا چک می‌کنیم آیا Hash tag وجود دارد یا خیر ، اگر وجود داشت مقدار آن را به دست می‌آوریم ، درخواست AJAX را صادر می‌کنیم و اطلاعات کالای 123 را به کاربر نشان می‌دهیم :

```
$(document).ready(function(){
if ($(window).location.hash.length))
{
//we will use $(window).location.hash.replace('#','') in the "data" section of the AJAX request
$.ajax({
 type: "POST",
 url: current_url,
 dataType: "html",
 data:'GoodId='+$(window).location.hash.replace('#',''),
 success: function(html) {
                //do something
 }  })
}
});
```

همانطور که مشاهده می‌کنید برای به دست آوردن مقدار Hashtag از کد پراپرتی hash شیء location استفاده شده ، این پراپرتی علامت # را هم بر می‌گرداند ، پس قبل یا بعد از ارسال مقدار این پراپرتی به سرور باید این علامت را حذف کرد. این مثال

Online را مشاهده کنید.

نظرات خوانندگان

نویسنده: ابراهیم
تاریخ: ۱۳۹۱/۰۵/۱۳ ۰:۲۱

ممنون از مطلب خوبتان. در حال حاضر راه حل دیگری هم برای این مشکل به وجود آمده که نیاز به # نیز ندارد.
https://developer.mozilla.org/en/DOM/Manipulating_the_browser_history#Adding_and_modifying_history_entries

اینجا هم مثالی از آن قرار دارد: http://html5.gingerhost.com/
موفق و پاینده باشید :)

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۱۳ ۰:۳۰

با PushState آشنایی نداشتم ، خیلی جالب هست ، ممنونم.

عنوان: **نمایش خروجی RSS سایت‌های دیگر به کمک jQuery**
نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۶/۰۴ ۲۱:۵۱
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Tips

شاید خیلی از دوستان (مثل گذشته نه چندان دور خودم) خیلی بیش از اندازه به برنامه نویسی‌های سمت سرور اهمیت می دهند که این کار باعث از دست دادن ، سرعت و سادگی برنامه نویسی سمت کلاینت می‌شود.

معمولا ما برای کار با خروجی‌های XML از کدهای سمت سرور استفاده می‌کنیم ، بدون اینکه از قدرت جی کوئری در این زمینه اطلاعی داشته باشیم. البته در این مقاله خیلی به پردازش XML توسط جی کوئری نمی‌پردازیم و کار اصلی را گوگل برای ما انجام می‌دهد.
برای انجام این کار ما ابتدا توسط یکی از کتابخانه‌های گوگل ، خروجی RSS یک وب سایت رامی خوانیم و بعد به کمک jQuery آن‌ها را نمایش می‌دهیم.

```
<script src="jquery-1.8.0.min.js" type="text/javascript"></script>
    <script type="text/javascript" src="https://www.google.com/jsapi"></script>
    <script type="text/javascript">

        google.load("feeds", "1");

        function initializeda() {
            var feed = new google.feeds.Feed("http://www.drupaleasy.ir/rss.xml");
            feed.setNumEntries(5);
            feed.setResultFormat(google.feeds.Feed.JSON_FORMAT);
            feed.load(function (result) {
                if (!result.error) {
                    for (var i = 0; i < result.feed.entries.length; i++) {
                        var entry = result.feed.entries[i];
                        $('#drupaleasy ul').append('<li><a href="' + entry.link + '">' + entry.title +
'</a></li>');
                    }
                }
            });
        }

google.setOnLoadCallback(initializeda);
    </script>
```

توضیحات کد :
ابتدا نیاز داریم کتابخانه گوگل را به صفحه اضافه کنیم. شما می‌توانید مستندات کامل این کتابخانه در این لینک بخوانید.

```
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
```

در ابتدا معرفی می‌کنیم که به کدام امکان این کتابخانه نیاز داریم ، در این جا ما از ورژن "1" تابع feed استفاده می‌کنیم.بد نیست برای آشنایی بیشتر سری هم به این لینک بزنید.

```
google.load("feeds", "1");
```

ابتدا لینک مورد نظر که باید خروجی از آن گرفته شود را معرفی می‌کنیم و آن را در متغییری قرار می‌دهیم.

```
var feed = new google.feeds.Feed("http://www.drupaleasy.ir/rss.xml");
```

تنظیمات دلخواه خودمان را نظیر تعداد پست واکشی شده و نوع خروجی مشخص می‌کنیم. تجربه (شخصی)نشان داده که بهترین نوع خروجی JSON است.

```
feed.setNumEntries(5);
feed.setResultFormat(google.feeds.Feed.JSON_FORMAT);
```

و بعد هم به کمک jQuery ساختار مورد نظر خودمان را تنظیم می‌کنیم. شما می‌توانید به کمک مستندات ، و سلیقه شخصی خودتان این کار را انجام بدهید ، و در آخر هم تنظیم می‌کنید ، اجرای تابع شما در چه زمانی اتفاق بیافتید.

```
feed.load(function (result) {
                if (!result.error) {
                    for (var i = 0; i < result.feed.entries.length; i++) {
                        var entry = result.feed.entries[i];
                        $('#drupaleasy ul').append('<li><a href="' + entry.link + '">' + entry.title +
'</a></li>');
                    }
                }
            });
```

در این جا من خروجی فید را به صورت یک لیست (ul) تنظیم کردم ، به این صورت که به ازای هر سطر یک li همراه با تگ a تولید کرده ام و به ul مورد نظر append کردم.

فایل HTML :

```
<div id="drupaleasy" class="feeds">
            <span>DrupalEasy.ir</span>
                <ul>
                </ul>
                <a href="http://drupaleasy.ir">more</a>
            </div>
```

همچنین خیلی راحت می‌توانید به کمک CSS استایل‌های دلخواه خودتون رو اعمال کنید.

```
.feeds
        {
            float: right;
            background-color: rgba(234, 242, 243, 0.73);
            margin: 5px;
            border-radius: 20px;
            padding: 8px;
            width: 293px;
            height: 217px;
            border: 1px solid #293883;
        }

        #drupaleasy ul
        {
            list-style-image: url("img/drupal.png");
        }
```

این هم شد خروجی کار من

DrupalEasy.ir
طراحی کاملا استاندارد برای اوبونتو
آموزش طراحی قالب دروپال - قسمت دوم
استفاده از گرادیانت سی اس اس 3 حتی در IE6 !
نگاهی به پایگاه داده دروپال
سی اس اس 3 (CSS3) در اختیار شماست
more

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۶/۰۶ ۹:۳۶

جالبه که گوگل امکان cross site ajax request رو می‌ده.

نویسنده: امیرحسین
تاریخ: ۱۳۹۱/۰۶/۰۸ ۱۳:۱۹

بله این امکان رو یاهو هم می‌دهد.
ولی استفاده از گوگل خیلی خوش دست‌تر و راحت‌تر هست.

نویسنده: علیرضا صبوری
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱:۱۵

سلام مطلب مفیدی بود ولی فکر میکنم بیشتر این قابلیت مربوط با کتابخانه گوگل هستش .. نه jquery . چون با جاواسکریپت هم به سادگی همین کار رو میشه کرد. موفق باشید

عنوان: **آپلود فایل ها با استفاده از PlUpload در Asp.Net Mvc**
نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۰۶ ۱۳:۴۳
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.Net MVC, jQuery, Ajax

امروزه بازار برنامه‌های تماما ajax و بدون Postback شدن صفحه بسیار داغ میباشد که از این موارد میتوان به برنامه‌های تحت وب گوگل اشاره کرد. (gmail ، googlePlus ، Google Reader)
در این میان یکی از دغدغه‌های توسعه دهندگان وب ، آپلود فایل‌ها به صورت آنی (مثل attach files گوگل) میباشد. برای حل این مسئله ، ابزارها و پلاگین‌های متعددی وجود دارد که در اینجا به 10 تا از پلاگین‌های Jquery اشاره شده است.
به شخصه با پلاگین Uploadify کار کرده ام و از استفاده از آن راضی هستم ولی همین دیشب برای قسمتی از یک پروژه نیاز به ابزاری جهت آپلود فایل‌ها با امکانات مورد نظرم داشتم که به PlUpload برخورد کردم.

از امکاناتی که این ابزار در اختیار شما قرار میدهد :
- یک اینترفیس زیبا جهت آپلود و افزودن فایل ها
- پشتیبانی از زبان‌های مختلف و همین طور زبان فارسی
- امکان استفاده از قالب Jquery UI
- Drag&Drop برای مرورگرهایی که از Html5 پشتیبانی میکنند

حال که با امکانات این ابزار بیشتر آشنا شدید بریم سراغ استفاده از این ابزار در asp.net mvc :)
ابتدا پروژه را از اینجا دانلود کنید. سپس یک پروژه‌ی جدید mvc 3 بسازید (از نوع Internet Application و با نام دلخواه). سپس پوشه‌ی plupload را در قسمت سلوشن برنامه کپی کنید.
حال در فایل Views->Shared->_Layout.cshtml ، تگ head را جهت افزودن امکانات پلاگین این گونه تغییر دهید :

```
<title>@ViewBag.Title</title>

    <link href="@Url.Content("~/Content/Site.css")" rel="stylesheet" type="text/css" />
    <link href="../../plupload/js/jquery.plupload.queue/css/jquery.plupload.queue.css" rel="stylesheet"
/>
    <script src="@Url.Content("~/Scripts/jquery-1.7.1.min.js")" type="text/javascript"></script>
    <script type="text/javascript" src="http://bp.yahooapis.com/2.4.21/browserplus-min.js"></script>

    <script src="../../plupload/js/plupload.full.js"></script>
    <script src="../../plupload/js/jquery.plupload.queue/jquery.plupload.queue.js"></script>
    <script src="../../plupload/js/i18n/fa.js"></script>
```

نکته : فایل fa.js که جهت استفاده از زبان فارسی در اینترفیس آپلود فایل‌ها میباشد، که وجود آن در آدرس واضح میباشد.
سپس به فایل Views->Home->Index.cshtml بروید و آن را این گونه دوباره نویسی کنید :

```
@{
    ViewBag.Title = "Uploading Files using PlUpload";
}
<h2>@ViewBag.Message</h2>

@using (Html.BeginForm("Post", "home", FormMethod.Post,
    new { enctype = "multipart/form-data" }))
{
    <div id="uploader">
        <p>You browser doesn't have Flash, Silverlight, Gears, BrowserPlus or HTML5 support.</p>
    </div>
}

<script>
    $(function () {

        $("#uploader").pluploadQueue({
            // General settings
            runtimes: 'html5,gears,flash,silverlight,browserplus,html4',
            url: '@Url.Action("Upload" , "Home")',
            max_file_size: '10mb',
            chunk_size: '1mb',
```

```
            unique_names: true,

            // Resize images on clientside if we can
            resize: { width: 320, height: 240, quality: 90 },

            // Specify what files to browse for
            filters: [
                { title: "Image files", extensions: "jpg,gif,png" },
                { title: "Zip files", extensions: "zip" }
            ],

            // Flash settings
            flash_swf_url: '/plupload/js/plupload.flash.swf',

            // Silverlight settings
            silverlight_xap_url: '/plupload/js/plupload.silverlight.xap'
        });
    });
</script>
```

توضیحات و نکات :

- جهت آپلود فایل‌ها تگ "enctype = "multipart/form-data را فراموش نکنید.

- در قسمت مقداردهی به ویژگی‌های Plupload ، قسمت runtime به صورت ترتیبی کار میکند لذا اگر اولی پشتیبانی نشود سراغ دومی میرود و اگر دومی نشود سومی و ... در صفحه‌ی اول سایت PlUpload ، موارد پشتیبانی شده توسط تکنولوژی‌ها آورده شده است لذا این ترتیب را ترتیب مناسبی میبینم و اگر اولین مورد html5 باشد امکان Drag&Drop وجود خواهد داشت.

خود سایت PlUpload داکیومنت خیلی خوبی جهت توضیح موارد مختلف دارد لذا توضیح دوباره لازم نیست.

همان طور که در ویژگی url مشاهده میکنید به کنترلر Home و اکشن متود Upload اشاره شده است که طرز کار به این گونه است که هر بار که یک فایل آپلود میشود درخواستی به این آدرس و محتوای فایل در قسمت Request.Files ارسال میشود و همین طور نام فایل که unique ارسال میشود و chunk که تیکه‌های فایل است(پست میشود).

پس اکشنی با نام Upload در کنترلر HomeController بسازید :

```
[HttpPost]
        public ActionResult Upload(int? chunk, string name)
        {
            var fileUpload = Request.Files[0];
            var uploadPath = Server.MapPath("~/App_Data");
            chunk = chunk ?? 0;
            using (var fs = new FileStream(Path.Combine(uploadPath, name), chunk == 0 ? FileMode.Create
: FileMode.Append))
            {
                if (fileUpload != null)
                {
                    var buffer = new byte[fileUpload.InputStream.Length];
                    fileUpload.InputStream.Read(buffer, 0, buffer.Length);
                    fs.Write(buffer, 0, buffer.Length);
                }
            }
            return Content("chunk uploaded", "text/plain");
        }
```

توضیحات : ابتدا فایل مورد نظر از قسمت Request.Files واکشی میشود و سپس فایل را در پوشه App_Data ذخیره میکند. (یکی از چندین روش ذخیره سازی که مطالعه در این قسمت به خواننده واگذار میشود.)

حال برنامه را اجرا کنید و از این ابزار لذت ببرید:) پروژه‌ی ضمیمه شده

نکته : قسمت فارسی ساز اونو تغییر دادم چون که ترجمه‌ی فارسی خودش یه سری نقایصی داشت که گویا از کار با google translate به وجود اومده بود!

نظرات خوانندگان

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۰۶ ۱۵:۵۲

پروژه ضمیمه شد!

نویسنده: امیرحسین
تاریخ: ۱۳۹۱/۰۶/۰۶ ۱۶:۲۶

سلام
همه این ابزارها برای ارسال فایل به سرور از فلش استفاده می‌کنند.
و یه طورایی نمی‌شه گفت این کار با ای جکس انجام شده (ولی می‌شه گفت بدون پست بک)
موفق باشید.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۰۶ ۱۷:۲۷

فلاش یکی از روش هاس ... ولی در کل حرف شما درسته و من به همین خاطر از واژه‌ی " آپلود فایل‌ها به صورت آنی " استفاده کردم

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۱/۰۶/۰۶ ۲۲:۹

ممنون از آموزش شما.

من خودم از این پلاگین استفاده می‌کنم.پلاگینی برای jquery هست و بدون استفاده از فلش و خیلی راحت با استفاده از ajax فایل را آپلود می‌کنه.

بازم ممنون

نویسنده: سعید یزدانی
تاریخ: ۱۳۹۱/۱۱/۲۷ ۱:۴۴

تشکر بابت مطلب کاربردیتون
فقط یک مشکل داره اونم وقتی هست که فایل‌ها upload شدن . دیگه اجازه‌ی ulpoad دوباره نمیده

نویسنده: asal Mansuri
تاریخ: ۱۳۹۲/۰۱/۰۲ ۲۱:۴۶

سلام
من یک فایل آپلودر خیلی جمع و جور میخوام که فقط یک فایل رو آپلود کنه ولی ایجکسی باشه / اگه لطف کنین راهنمایی کنین که چطوری میتونم فایل آپلود رو در MVC بتونم به صورت ajax همراه با loading داشته باشم ممنون میشم .

نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۳:۱۲

سلام
چطور میشه از لیست اسامی فایل‌های ارسال شده، در هنگام ثبت اطلاعات مدل در سرور استفاده کرد؟

در واقع اطلاعات فایل ها، قسمتی از اطلاعات مدل من محسوب میشه و در هنگام ثبت در دیتابیس باید این لیست رو هم ثبت کنم. ولی نمیدونم چطور در کنترلر ثبت مدل به اسامی فایلها دسترسی داشته باشم.
مثلا مدل زیر :

```
public class album
{
public string  name;
public string des;
public string[] images;
}
```

من اطلاعات مدل رو در حالت آجاکس به صورت ajax.beginform ارسال میکنم.

ممنون میشم اگر دوستان پاسخ بدن

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۳:۵۳

در متد public ActionResult Upload بالا fileUpload.FileName اسم واقعی فایل هست؛ پارامتر name یک اسم موقتی.

نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۴:۱۶

خیلی ممنون از پاسختون
این قسمتی رو که گفتین درسته. من مرحله بعد از این رو مشکل دارم.
سوال اینه:
بعد از ارسال فایل‌ها به سرور، حالا نوبت ارسال اطلاعات مدل به سرور می‌رسه که این اسامی فایل‌ها هم قسمتی از اطلاعات مدل هست. چطور می‌شود در اکشن ثبت مدل، اسامی فایل‌ها اپلود شده را دریافت کرد و استفاده نمود.
یه راه حلی پیدا کردم که گفته بود بعد از اینکه کار آپلودر تموم شد اطلاعات فایل‌ها را به صورت فیلد مخفی در فرم ذخیره کن تا پست بک شود. نمیدونم آیا راه حل دیگه ای هست؟ این روش درست و اصولیه؟

```
uploader.bind('FileUploaded', function(up, file, info) {
  var obj = JSON.parse(info.response);
    $('form#quoteRequest').append('<input type="hidden" name="file_name" value="' +
obj.result.cleanFileName + '" />');
    //note obj.result.cleanFileName instead obj.cleanFileName
 });
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۴:۱۶

اینکار توصیه نمیشه. چون پس از اطمینان کامل از آپلود یک فایل باید نسبت به ذخیره اطلاعات آن اقدام کنید. الزامی نداره که ارسال فرم حتما به آپلود موفقیت آمیز فایل‌ها در این حالت خاص منجر بشه (چون مثلا از فلش یا سیلورلایت مجزای از فرم شما هم ممکن است برای ارسال داده‌های فایل‌ها استفاده کنه). یعنی آپلود فایل در اینجا بحث مجزایی از فرم شما است. ابتدا فایل‌ها را آپلود کنید؛ بعد اجازه بدید تا توضیحات مرتبط را ثبت کنند.
یا اینکه می‌شود در حین آپلود توسط افزونه هم یک سری توضیحات اضافی رو ارسال کرد:

```
// JS
up.settings.multipart_params = {
                    description: $("#imageDescription").val()
                 };
// C#
string description = context.Request.Form["description"];
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۴:۲۱

پیاده سازی دیگری از بحث PlUpload

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۴:۳۰

```
// JS
up.settings.multipart_params = {
                    description: $("#imageDescription").val(),
                    // other fields ....
                 };
// C#
string description = context.Request.Form["description"];
```

باید از multipart_params تنظیمات این افزونه برای ارسال اطلاعات اضافی استفاده کنید.

نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۴:۴۳

با تشکر از دوستان .
یک مثال عملی؛ یه نرم افزار **ارسال ایمیل تحت وب مثل جیمیل** رو در نظر بگیرید. یه **فرم ارسال ایمیل** داره که می‌خواد اطلاعات رو بفرسته به سمت سرور. یه سری فایل هم قراره بصورت آجاکسی پیوست بشه. مراحل کار بدین صورته که ابتدا فایل‌ها آجاکسی به سرور فرستاده می‌شوند و یه جایی ذخیره می‌شوند. بعد فرم اطلاعات شامل گیرنده ، متن نامه و غیره بصورت آجاکس ارسال میشه. وقتی در سمت سرور می‌خوایم ایمیل رو بفرستیم، نیاز داریم که نام فایل‌های پیوست شده در مرحله قبل رو هم داشته باشیم.
همونطور که قبلا خدمت دوستان عرض کردم؛ راه حلی که من بذهنم می‌رسه اینه که بعد از اینکه فایل‌ها در مرحله اول آپلود شد، اسم فایل‌ها رو توی فرم در فیلد مخفی تزریق کنیم. بعد که فرم پست شد، طبیعتا دسترسی به نام فایل‌ها داریم. کد ذکر شده هم کار تزریق در هنگام تمام شدن ارسال فایل‌ها انجام میداد.
خواستم بدونم آیا این روش درست هست؟ و آیا روش تمیزتری وجود داره؟
باز هم تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۴:۵۶

همان روشی که برای دریافت نام فایل‌های آپلود شده از سرور انتخاب کردید، مناسب است:

```
$("#uploader").pluploadQueue({
      // Post init events, bound after the internal events
      init: {
               FileUploaded: function(up, file, info) {
               // Called when a file has finished uploading
               }
      }
});
```

نویسنده: فرشته محسنیان
تاریخ: ۱۳۹۲/۰۶/۱۵ ۹:۳۶

ممنون از مطالب که گفتین. فقط سایتش دیگه اجازه دانلود نمیده، آیا از جایی دیگه میتونم بگیرم؟ و دوم اینکه mvc 4 رو هم ساپورت میکنه یا نه؟ مرسی

نویسنده: محسن خان

تاریخ: ۱۳۹۲/۰۶/۱۵ ۱۰:۲۰

plupload سورس باز هست . البته با مجوز GPL. یعنی کار خودتون رو باید سورس باز کنید (یا مجوزش رو بخرید). ضمنا MVC4 با MVC3 سازگار هست. فقط یک سری افزونه بیشتر داره.

نویسنده: جواد
تاریخ: ۱۳۹۳/۰۳/۲۱ ۹:۱

باسلام. من این کارهایی که گفتید کردم. روی vs مشکلی نداشتم. ولی وقتی نرم افزارمو پابلیش کردم و روی سرور گذاشتم متأسفانه هرکار می‌کنم خطا عدم پشتیبانی مرورگر میده. با فایرباگ چک کردم دیدم به فایلهای کتابخانه اش عدم دسترسی می‌ده. تمام تنظیمات iis رو هم چک کردم. نتونستم درستش کنم.
خطایی که میده اینطوریه:

"NetworkError: 404 Not Found - http://~/Scripts/plupload/js/jquery.ui.plupload/jquery.ui.plupload.js

نویسنده: ناهید
تاریخ: ۱۳۹۳/۰۵/۰۶ ۹:۴

سلام
یه سوال؟ وقتی بخوایم اطلاعات ذخیره شده (فایلی که قبلا آپلود کرده) رو نمایش بدیم چجوری فایل رو توی plUpload نمایش بدیم؟ خود سایت FilesAdded داره که کار نمیکنه.

عنوان: **مدیریت کوکی ها با jQuery**
نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۶/۱۲ ۳:۵
آدرس: www.dotnettips.info
برچسب‌ها: JSON, jQuery, jQuery-Tips

در گذشته نه چندان دور، کوکی‌ها نقش اصلی را در مدیریت کاربران ، و ذخیره اطلاعات کاربران ایفا می‌کردند. ولی بعد از کشف شدن باگ امنیتی (که ناشی از اشتباه برنامه نویس بود) در کوکی ها، برای مدتی کنار گذاشته شدند و اکثر اطلاعات کاربران در session های سمت سرور ذخیره می‌شد.
ذخیره اطلاعات زیاد و نه چندان مهم کاربران در session های سمت سرور ، بار زیادی را به سخت افزار تحمیل می‌کرد. بعد از این، برنامه نویسان به سمتی استفاده متعادل از هرکدام این‌ها (کوکی و سشن) رفتند.
اکثر دوستان با مدیریت سمت سرور کوکی‌ها آشنایی دارند ، بنده قصد دارم در اینجا با استفاده از یک پلاگین جی کوئری مدیریت کوکی‌ها را نمایش دهم.
در این برنامه ما از پلاگی jQuery.cookie استفاده می‌کنیم که شما می‌توانید با مراجعه به صفحه این پلاگین اطلاعات کاملی از این پلاگین به دست بیاورید.
کار با این پلاگین بسیار ساده است.
ابتدا فایل پلاگین را به صفحه خودتون اضافه می‌کنید.

```
<script src="/path/to/jquery.cookie.js"></script>
```

حالا خیلی راحت می‌توانید با این دستور یک مقدار را در کوکی قرار دهید.

```
$.cookie('the_cookie', 'the_value');
```

و برای گرفتن کوکی نوشته شده هم به این صورت عمل می‌کنید.

```
$.cookie('the_cookie'); // => "the_value"
```

همان طور که دیدید کار بسیار ساده ای است. ولی قدرت این پلاگین در option هایی است که در اختیار ما قرار می‌دهد.
مثلا شما می‌توانید انتخاب کنید این کوکی برای چند روز معتبر باشد ، و یا اطلاعات را به صورت json ذخیره و بازیابی کنید، و حتی option های دیگری برای بحث امنیت کوکی شما.
برای درک بهتر از قطعه کدی که کمی پیچیده‌تر از مثال منبع است، استفاده می‌کنیم.

به کد زیر توجه کنید :

JavaScript :

```
<script type="text/javascript">
        $(function () {
            $('#write').click(function () {
                $.cookie('data', '{"iri":"Iran","usa":"United States"}', { expires: 365, json: true });
                alert('Writed');
            });

            $('#show').click(function () {
                var obj = jQuery.parseJSON($.cookie('data'));
                alert(obj.iri);
            });

            $('#remove').click(function () {
                $.removeCookie('data');
            });
        })
    </script>
```

HTML :

```
<body>
    <a href="#" id="write">Write</a>
    <br />
    <a href="#" id="show">Show</a>
    <br />
    <a href="#" id="remove">Remove</a>
</body>
```

در اینجا ما سه لینک داریم که هر کدام برای ما عملی را نمایش میدهند.

توضیحات کد :

```
$('#write').click(function () {
                $.cookie('data', '{"iri":"Iran","usa":"United States"}', { expires: 365, json: true });
                alert('Writed');
            });
```

با کلیک بر روی لینک Write کوکی data با مقدار مشخص پر می‌شود.
دقت داشته باشید که این مقدار از نوع json انتخاب شده است و در انتها نیز این را مشخص کرده ایم ، همچنین اعلام کرده ایم که این کوکی برای 365 روز معتبر است.
حالا مرورگر خودتان را ببندید و دوباره باز کنید.
این بار بر روی Show کلیک می‌کنیم :

```
$('#show').click(function () {
                var obj = jQuery.parseJSON($.cookie('data'));
                alert(obj.iri);
```

با کلیک بر روی لینک Show مقدار از کوکی خوانده می‌شود و نمایش داده می‌شود. دقت کنید ، به دلیل اینکه مقدار ذخیره شده ما از نوع json است باید دوباره این مقدار را pars کنیم تا به مقادیر property آن دسترسی داشته باشیم.
همچنین شما می‌توانید خیلی راحت کوکی ساخته شده را از بین ببرید :

```
$('#remove').click(function () {
                $.removeCookie('data');
            });
```

و یا این که کوکی را برابر null قرار دهید.
نکته ای که باید رعایت کنید و در این مثال هم نیامده است ، این است که ، هنگامی که شما می‌خواهید object ی که با کد تولید کرده اید در کوکی قرار بدهید ، باید از متد JSON.stringify استفاده کنید و مقدار را به این صورت در کوکی قرار دهید.

```
$.cookie('data', JSON.stringify(jsonobject), { expires: 365, json: true });
```

که در اینجا jsonobject ، ابجکتی است که شما تولید کرده اید و قصد ذخیره آن را دارید.
من از این امکان در نسخه بعدی این پروژه استفاده کرده ام ، و به کمک این پلاگین ساده اما مفید ، وب سایت هایی که کاربر نتایج آن را مشاهده کرده است در کوکی کاربر ذخیره می‌کنم تا در مراجعه بعدی میزان تغییرات رنکینگ‌های وب سایت ای در خواست شده را ، به کاربر نمایش دهم. نسخه بعد all-ranks.com تا آخر هفته آینده در سرور اختصاصی (و نه این هاست رایگان (!)) قرار می‌گیرد و به مرور قسمت هایی که در این پروژه پیاده سازی شده (پلاگین‌های جی کوئری و کدهای سرور) در اینجا شرح می‌دهم.
امیدوارم تونسته باشم مطلب مفید و مناسبی به شما دوستان عزیزم انتقال بدم.

نظرات خوانندگان

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۶/۱۲ ۱۱:۵۵

خیلی ممنون ... این سبک مطلبارو دوست دارم :)

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۶/۱۲ ۱۲:۲

پس در این زمینه هم عقیده ایم :)
موفق باشید.

عنوان: ایجاد alert,confirm,prompt هایی متفاوت با jQuery Impromptu
نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۶/۱۸ ۱۸:۴۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, FireBug, jQuery

alert,confirm,prompt سه متد توکار JavaScript هستند که برای نمایش پیغام ، دریافت تایید و دریافت مقدار از کاربر هستند .

گرافیک این پیغام‌ها هم وابسته به مرورگر هستند و قابل تغییر نیستند . متن عنوان و دکمه‌ها هم با توجه به زبان سیستم عامل تعیین می‌شوند و قابل تغییر توسط برنامه نویس نیستند.

حال مواقعی پیش می‌آید که نیاز داریم پیغام هایی با گرافیک و عبارات متفاوت نمایش دهیم . برای رفع این نیاز می‌توانیم از پلاگین jQuery Impromptu استفاده کنیم . البته این پلاگین قابلیت‌های دیگری هم دارد که در این مقاله با آنها آشنا می‌شویم .

از ویژگی‌های این پلاگین می‌توان حجم کم (حدود 11 کیلوبایت) و قدرت شخصی سازی بالا اشاره کرد .

طرز استفاده به این شکل است :

```
$.prompt( msg , options )
```

msg می‌تواند یک string یا یک شئ از states باشد . string ارسال شده می‌تواند شامل کدهای html باشد .

یک شئ states هم شامل مجموعه ای از وضعیت‌های prompt است . برای مثال می‌توان یک prompt ایجاد کرد که مثل یک Wizard شامل چند مرحله باشد .

options هم تنظیماتی است که می‌توان مشخص کرد . تنظیماتی مثل : prefix,classes,persistent,timeout,...

msg :

گفتیم شئ states شامل وضعیت‌های مختلف prompt است . هر وضعیت (state) می‌تواند شامل بخش‌های زیر باشد :

html : مقدار Html وضعیت

buttons : یک شئ شامل متن و مقدار دکمه هایی که کاربر می‌تواند کلیک کند

focus : ایندکس دکمه‌ی focus شده در وضعیت

submit : تابعی که زمانی که یکی از دکمه‌های وضعیت انتخاب شود فراخوانی می‌شود .

اگر در این تابع false بازگشت داده شود یا متد preventDefault از event فراخوانی شود ، prompt باز می‌ماند . (روشی برای جلوگیری از بسته شدن prompt هنگام تغییر state یا اعتبار سنجی فرم)
همچنین شئ event شامل state (المنت state) و stateName (نام state) می‌باشد .

پیشفرض :

```
function(event, value, message, formVals){}
```

value مقدار دکمه ای است که بروی آن کلیک شده ، message مقدار html تعریف شده برای state است ، formVals هم در صورتی که در html تعریف شده برای state ، المنت‌های فرم وجود داشته باشد ، شامل نام/مقادیر آنها می‌باشد . (برای دریافت مقادیر فرم ، باید از نام المنت استفاده نمایید .)

position : مشخص کننده‌ی موقعیت state که شامل موارد زیر است :

```
position: { container: '#container', x: 0, y: 0, width: 0, arrow: 'lm' }
```

selector : **container** المنتی است که state باید در آن مکان قرار بگیرد .
x/y : موقعیت نسبی prompt نسبت به container
arrow : جهت نمایش فلش prompt است که می‌تواند یکی از این مقادیر باشد : tl, tc, tr, rt, tm, tb, br, bc, bl, lb, lm, lt.

نحوه تعریف یک wizard ساده با شئ states :

```
var tourSubmitFunc = function (e, v, m, f) {
    if (v === -1) {
        $.prompt.prevState();
```

```
        return false;
    } else if (v === 1) {
        $.prompt.nextState();
        return false;
    }
};

var states =
    {
        state0:
            {
                html: "State1",
                buttons: { Next: 1 },
                //position: { container: '#container', x: 10, y: 0, width: 350, arrow: 'lm' },
                submit: tourSubmitFunc
            },
        state1:
            {
                html: "State2",
                buttons: { Prev: -1, Next: 1 },
                submit: tourSubmitFunc
            },
        state2:
            {
                html: "State3",
                buttons: { Prev: -1, Done: 0 },
                submit: tourSubmitFunc
            }
    };

$.prompt(states);
```

تا به اینجا با پارامتر اول prompt آشنا شدیم و فهمیدیم که می‌توانیم یک رشته یا یک شئ states به عنوان message به prompt ارسال کنیم .

options :

اکنون با optionهای prompt (پارامتر دوم) آشنا خواهیم شد .

توجه کنید که زمانی که یک رشته به prompt ارسال کنید ، مقادیر buttons,focus,submit از این تنظیمات دریافت می‌شود .
به عبارت دیگر ، زمانی که یک شئ states به prompt ارسال کنید ، از مقادیر فوق که در تنظیمات است ، استفاده نمی‌شود .

loaded

یک تابع که زمانی که prompt کامل بارگزاری شده فراخوانی می‌شود .

```
$.prompt("Message",
    {
        loaded: function() {
            alert("Prompt Loaded !");
        }
    });
```

submit

یک تابع که زمانی که یکی از دکمه‌های state کلیک شود ، فراخوانی می‌شود .
(زمانی اتفاق میوفتد که یک رشته به عنوان متن به prompt ارسال کرده باشید و زمانی که یک شئ از states ارسال می‌کنید ، هنگام کلیک دکمه‌های آنها ، این تابع فراخوانی نمی‌شود .)
پیشفرض :

```
function(event){}
```

statechanging

یک تابع که زمانی که یک state در حال تعویض شدن هست فراخوانی می‌شود .

پیشفرض :

```
function(event, fromStateName, toStateName){}
```

برای لغو تغییر state ، مقدار return false کنید یا متد preventDefault از event را فراخوانی کنید .

statechanged

یک تابع که زمانی که یک state در حال تعویض شدن هست فراخوانی می‌شود .

پیشفرض :

```
function(event, toStateName){}
```

callback

یک تابع که زمانی که (یکی از دکمه‌های prompt کلیک شود و) prompt بسته شود ، فراخوانی می‌شود .

پیشفرض :

```
function(event[, value, message, formVals]){}
```

سه پارامتر آخر تنها زمانی که یک دکمه‌ی prompt کلیک شده باشد موجود هستند .

buttons

یک شئ شامل مجموعه ای از دکمه‌ها .

پیشفرض :

```
{ Ok : true }
```

شکل دیگر تعریف دکمه به این شکل است :

```
[
    {title: 'Hello World',value:true},
    {title: 'Good Bye',value:false}
]
```

prefix

یک پیشوند برای همه cssها و idهای المنت‌های html که توسط prompt ایجاد می‌شود .

پیشفرض : jqi

classes

یک css class که به بالاترین سطح prompt داده می‌شود .

در حالت پیشفرض مقداری ندارد .

focus

ایندکس دکمه‌ی focus شده

پیشفرض : 0

zIndex

zIndex اعمال شده بروی prompt .
پیشفرض : 999

useiframe
استفاده از یک iframe برای overlay در IE6
پیشفرض : false

top
فاصله‌ی prompt از بالای صفحه
پیشفرض : 15%

opacity
میزان شفافیت لایه‌ی ای که صفحه را پوشانده است .
پیشفرض : 0.6

overlayspeed
سرعت نمایش افکت fadeIn , fadeOut لایه‌ی پوشاننده .
مقادیر قابل قبول : "slow", "fast", number(milliseconds)
پیشفرض : "slow"

promptspeed
سرعت نمایش prompt .
مقادیر قابل قبول : "slow", "fast", number(milliseconds)
پیشفرض : "fast"

show
نام یک متد jQuery برای animate کردن نمایش prompt .
مقادیر قابل قبول : "slideDown","fadeIn","show", ...
پیشفرض : "promptDropIn"

persistent
بسته شدن prompt زمانی که بروی لایه‌ی fade کلیک شود .
false : بسته شدن
پیشفرض : true

timeout
مدت زمانی که پس از آن ، prompt بصورت خودکار بسته می‌شود . (milliseconds)
پیشفرض : 0

: **returns**

مقدار بازگشتی متد prompt ، یک شئ jQuery ، شامل همه‌ی محتویات تولید شده توسط prompt است .

: **Events using Bind**

استفاده از Eventها با بایند کردن

promptloaded

معادل loaded در optionها .

promptsubmit

هنگام submit شدن (کلیک شدن یکی از دکمه‌های) state فعال می‌شود .

promptclose

معادل callback در optionها .

promptstatechanging

معادل statechanging در optionها .

promptstatechanged

معادل statechanged در optionها .

ظرز بایند کردن یک event به شئ prompt :

```
var myPrompt = $.prompt( msg , options );
myPrompt.bind('promptloaded', function(e){});
```

Helper Functions :

jQuery.prompt.setDefaults(options)

تعیین مقادیر پیشفرض برای همه‌ی promptها .

```
jQuery.prompt.setDefaults({
prefix: 'myPrompt',
show: 'slideDown'
});
```

jQuery.prompt.setStateDefaults(options)

تعیین مقادیر پیشفرض برای stateها .

```
jQuery.prompt.setStateDefaults({
buttons: { Ok:true, Cancel:false },
focus: 1
});
```

jQuery.prompt.getCurrentState()

یک شئ jQuery از state جاری برمی‌گرداند .

jQuery.prompt.getCurrentStateName()

نام state جاری را برمی‌گرداند .

jQuery.prompt.getStateContent(stateName)

یک شئ jQuery از state مشخص شده برمی‌گرداند .

jQuery.prompt.goToState(stateName, callback)

state مشخص شده را در prompt نمایش می‌دهد .

callback تابعی است که بعد از تغییر state فراخوانی می‌شود .

jQuery.prompt.nextState(callback)

prompt را به state بعدی منتقل می‌کند .

jQuery.prompt.prevState(callback)

prompt را به state قبلی منتقل می‌کند .

jQuery.prompt.close()

prompt را می‌بندد .

حال که با این پلاگین آشنا شدیم ، سه دستور جاوا اسکریپتی که در ابتدای مقاله ذکر کردیم را با این پلاگین پیاده سازی می‌کنیم .

alert

```
$.prompt("یک پیام تستی",
    {
        prefix: 'dnt',
        buttons: { 'تایید': true }
    });
```

نتیجه :

confirm

```
$.prompt("درخواست تایید - موافقید ؟",
    {
        prefix: 'dnt',
        buttons: { 'تایید': true, 'انصراف': false }
    });
```

نتیجه :

prompt

یک فرم html مخفی برای نمایش در prompt :

```
<div class="prompt-content" style="display: none;">
    <span>نام خود را وارد نمایید : </span>
    <span>
        <input type="text" name="name" />
    </span>
</div>
```

نمایش prompt :

```
$.prompt(
    $(".prompt-content").html(),
    {
        prefix: 'dnt',
        buttons: { 'تایید': true, 'انصراف': false }
    });
```

نتیجه :

در این سه مثال آخر ، از یک css متفاوت استفاده کردیم . این پلاگین یک سری کلاس دارد که نام این کلاس‌ها از ترکیب مقدار prefix که در option مشخص کردیم حاصل می‌شود .

برای مثال اگر مقدار prefix را برابر با dnt قرار بدهیم ، برای استایل دهی متن پیام باید از کلاس div.dnt .dntmessage استفاده کنید .

همانطور که در سه مثال آخر مشاهده کردید ، تغییری در استایل prompt داشتیم که با تغییر دادن استایل‌های مورد نظر انجام شد .

برای تهیه‌ی یک قالب سفارشی باید selector المنت هایی که نیاز به تغییر دارند را از قالب پیشفرض پیدا کنید ، سپس قسمت prefix از selector را به نام قالب خودتان تغییر بدید و استایل را ویرایش کنید . سپس هنگام ایجاد prompt ، مقدار prefix را برابر نام قالب قرار دهید .

مثلا می‌خواهید یک قالب به نام dnt داشته باشید . می‌خواهید متن قالب راست به چپ باشد .

```
div.dnt .dntmessage {
     color: #444444;
     line-height: 20px;
     padding: 10px;
/*       Edited      */
     direction:rtl;
     text-align:right;
}
```

البته راه بهتری هم هست که نیاز به آشنایی با فایرباگ دارد . در این روش ابتدا کل قالب jqi (موجود در قالب پیشفرض) را در برنامه خود کپی می‌کنیم ، مقادیر jqi را با نام قالب جایگزین می‌کنیم ، مقدار prefix را در prompt برابر با نام قالب قرار می‌هیم . اکنون در FireFox یک prompt ایجاد می‌کنیم و توسط فایرباگ استایل هایی که با نام قالب بروی prompt اعمال شده‌اند را مطابق سلیقه تغییر می‌دهیم . در مرحله آخر به تب CSS در فایرباگ می‌رویم و کل استایل‌های مربوط به قالب را کپی و جایگزین استایل قبلی در برنامه می‌کنیم .

قالبی که بنده برای سه دستور فوق استفاده کردم (dnt) ، به این شکل است :

```
/*    Start : DotNetTips Theme     */

.dntfade {
     background-color: #AAAAAA;
     position: absolute;
}

div.dnt {
     background-color: #FFFFFF;
     border-radius: 10px 10px 10px 10px;
     border: 1px solid #FFFFFF;
     box-shadow: 0px 0px 10px 1px #6D6D6C;
     font-family: tahoma;
     font-size: 11px;
     padding: 7px;
     position: absolute;
     text-align: left;
     width: 400px;
}

div.dnt .dntcontainer {
     font-size: small;
}

div.dnt .dntclose {
     color: #BBBBBB;
     cursor: pointer;
     font-weight: bold;
     position: absolute;
     top: 4px;
     width: 18px;
}

div.dnt .dntmessage {
     color: #444444;
     line-height: 20px;
     padding: 10px;
}

div.dnt .dntbuttons {
```

```
     background-color: #F4F4F4;
     border: 1px solid #EEEEEE;
     padding: 5px 0px;
     text-align: right;
}

div.dnt button {
     background-color: #2F6073;
     border: 1px solid #F4F4F4;
     color: #FFFFFF;
     font-size: 12px;
     font-weight: bold;
     margin: 0px 10px;
     padding: 3px 10px;
}

div.dnt button:hover {
     background-color: #728A8C;
}

div.dnt button.dntdefaultbutton {
     background-color: #0099CC;
}

.dnt_state {
     direction: rtl;
     text-align: right;
}

.dnt_state button {
     font-family: tahoma;
}

.dntwarning .dnt .dntbuttons {
     background-color: #CCDDFF;
}

.dnt .dntarrow {
     border: 10px solid transparent;
     font-size: 0px;
     height: 0px;
     line-height: 0;
     position: absolute;
     width: 0px;
}

.dnt .dntarrowtl {
     border-bottom-color: #FFFFFF;
     left: 10px;
     top: -20px;
}

.dnt .dntarrowtc {
     border-bottom-color: #FFFFFF;
     left: 50%;
     margin-left: -10px;
     top: -20px;
}

.dnt .dntarrowtr {
     border-bottom-color: #FFFFFF;
     right: 10px;
     top: -20px;
}

.dnt .dntarrowbl {
     border-top-color: #FFFFFF;
     bottom: -20px;
     left: 10px;
}

.dnt .dntarrowbc {
     border-top-color: #FFFFFF;
     bottom: -20px;
     left: 50%;
     margin-left: -10px;
}

.dnt .dntarrowbr {
     border-top-color: #FFFFFF;
     bottom: -20px;
```

```
     right: 10px;
}

.dnt .dntarrowlt {
     border-right-color: #FFFFFF;
     left: -20px;
     top: 10px;
}

.dnt .dntarrowlm {
     border-right-color: #FFFFFF;
     left: -20px;
     margin-top: -10px;
     top: 50%;
}

.dnt .dntarrowlb {
     border-right-color: #FFFFFF;
     bottom: 10px;
     left: -20px;
}

.dnt .dntarrowrt {
     border-left-color: #FFFFFF;
     right: -20px;
     top: 10px;
}

.dnt .dntarrowrm {
     border-left-color: #FFFFFF;
     margin-top: -10px;
     right: -20px;
     top: 50%;
}

.dnt .dntarrowrb {
     border-left-color: #FFFFFF;
     bottom: 10px;
     right: -20px;
}

/*    End : DotNetTips Theme     */
```

برای مشاهده‌ی مثال‌های بیشتر به صفحه‌ی اصلی jQuery Impromptu مراجعه نمایید .

نظرات خوانندگان

نویسنده: بهمن خلفی
تاریخ: ۱۳۹۱/۰۶/۱۹ ۱۰:۳۶

با سلام و تشکر از شما

نمونه پست ارسالی شما در این آدرس نیز موجود است لطفا تفاوت کارایی‌های این دو را بفرمائید و از طرفی اگر امکان دارد نحوه استفاده آن در دات نت را برای تائید و دریافت ورودی توضیح دهید باتشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۹ ۲۳:۵۷

- تفاوت مهم آن در حجم کمتر است. jQuery-UI نسبتا حجم بالایی دارد و اگر صرفا قرار است پیام کوتاهی به کاربری نمایش داده شود، روش فوق حداقل از لحاظ حجم (1lK) مقرون به صرفه‌تر است.
- استفاده از نتایج آن هم مطابق توضیحات مفصل مطلب جاری، در قسمت callback آن باید صورت گیرد. برای مثال

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۶/۲۰ ۱۴:۵۹

ممنون،
یه سوال بنده یه فرم ثبت نام دارم که با کلیک بر روی Botton اطلاعات در دیتابیس درج می‌شود حالا می‌خوام بعد از ثبت اطلاعات یه alert به کاربر نشون بده که اطلاعات با موفقیت ثبت شد ولی مشکل اینجاست که وقتی کاربر روی Button کلیک میکنه چون صفحه PostBack میشه دیگه پیغام به کاربر نمایش داده نمیشه،اگه امکان داره بنده رو راهنمایی بفرمائید.
دقیقا مثل همین قسمت ارسال نظر سایت.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۰ ۱۵:۵۹

این مطلب می‌تونه مفید و مرتبط باشه.

نویسنده: رضا
تاریخ: ۱۳۹۱/۰۷/۱۲ ۲:۳۲

سلام. یک linkbutton دارم تو گرید ویو که کار حذف رو انجام میده. در حالت پیش فرض با دستور

```
return confirm('')
```

و انتساب اون به خاصیت onClientClick میشد یک confirm ساده قبل حذف داشت. حالا وقتی میخام از این کنترل برای این کار استفاده کنم کار نمیکنه؟ میشه راهنمایی کنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۱۲ ۹:۴۶

مثالی که در این کامنت معرفی شد مرتبط به GridView هم هست (انتهای مقاله).

نویسنده: omid

تاریخ: ۱۳۹۱/۱۰/۱۲ ۸:۱۹

سلام
با تشکر از راهنمایی شما برای کنترل‌ها من از این کنترل می‌خواستم استفاده کنم ولی اجرا نشد اگه راهنماییم کنید ممنون میشم .

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>
<meta content="text/html; charset=UTF-8" http-equiv="Content-Type" />
<title></title>

<link href="jquery-impromptu.css" media="all" rel="stylesheet" type="text/css" />
<script src="jquery-1.8.3.min.js" type="text/javascript"></script>
<script src="jquery-impromptu.js" type="text/javascript"></script>
<script type="text/javascript">

$(function(){
$show.click(function(e){
$.prompt("Hello World!");
});
});
});

</script>
</head>

<body>
<button class="show">ShowPrompt</button>
</body>
</html>
```

ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۲ ۱۰:۲۴

- کد شما تعداد {} متوازنی ندارد. دو } بازه شده، سه تا بسته.
- به نظر با آخرین نگارش jQuery سازگار نیست. به صفحه خطای مرورگر مراجعه کنید تا خطای شناخته نشدن نوع مرورگر قابل مشاهده باشد. با نگارش‌های پایین‌تر مشکلی ندارد .

نویسنده: omid
تاریخ: ۱۳۹۱/۱۰/۱۲ ۱۰:۵۶

سلام
خیلی ممنون بابت راهنماییتون و آشنایی با سایت http://jsfiddle.net

نویسنده: neda
تاریخ: ۱۳۹۲/۰۱/۰۹ ۱۲:۲۶

ممنون از سایت خیلی خوب و مفیدتون
ممنون میشم اگه امکانش هست یه سمپل ساده و کوچیک از این کار هم در اختیار بگذارید
من از کدها استفاده کردم ولی موفق نشدم خروجی صحیح بگیرم
ممنونم

نویسنده: احمد احمدی
تاریخ: ۱۳۹۲/۰۱/۰۹ ۱۷:۱۷

```
<!DOCTYPE html>
<html>
```

```
<head>
<link rel="stylesheet" media="all" type="text/css" href="http://trentrichardson.com/Impromptu/jquery-impromptu.css" />
<script type="text/javascript" src="http://code.jquery.com/jquery-1.9.0.min.js"></script>
<script type="text/javascript" src="http://trentrichardson.com/Impromptu/jquery-impromptu.js"></script>
</head>
<body>
<button class="show">ShowPrompt</button>
<script type="text/javascript">
$(function(){
   $(".show").click(function(e){
      $.prompt("Hello World!");
   });
});
</script>
</body>
</html>
```

نویسنده: سمیرا
تاریخ: ۱۳۹۲/۰۳/۰۸ ۱۳:۴۴

سلام؛ وقتی با استفاده از jQuery Impromptu یک inputbox ایجاد می‌کنیم و کاربر اطلاعاتی رو توی inputbox وارد می‌کنه، چجوری می‌تونیم مقداری که کاربر وارد کرده رو برگردونیم و ازش استفاده کنیم ؟

نویسنده: مژده
تاریخ: ۱۳۹۲/۰۳/۰۸ ۱۴:۲۳

سلام
من هم همین مشکلو داشتم
از این روش استفاده کردم جواب داد :

```
var myval = m.find('#name').val();
alert(myval);
```

نویسنده: احمد احمدی
تاریخ: ۱۳۹۲/۰۳/۰۸ ۱۸:۴۴

سلام؛

طبق مطلب ارائه شده در مقاله:

تابع submit :

```
function(event, value, message, formVals){}
```

value مقدار دکمه ای است که بروی آن کلیک شده ، message مقدار html تعریف شده برای state است ، formVals هم در صورتی که در html تعریف شده برای state ، المنت‌های فرم وجود داشته باشد ، شامل نام/مقادیر آنها می‌باشد . (برای دریافت مقادیر فرم ، باید از نام المنت استفاده نمایید .)

خب حالا مشابه مثال زیر عمل کنید:

```
<div class="prompt-content" style="display: none;">
    <span>ایمیل خود را وارد نمایید : </span>
    <span>
        <input type="text" name="user_email" />
    </span>
```

```
</div>
```

```
$.prompt(
    $(".prompt-content").html(),
    {
        submit: function (e, v, m, f) {
            var userEmail = f["user_email"];
            console.log(userEmail);
        }
    });
```

نویسنده: مژده
تاریخ: ۱۳۹۲/۰۳/۰۹ ۲۱:۱۳

سلام
خسته نباشید
من توی سایتم از ajax استفاده کردم و برای حذف یه رکورد اول از کاربر تاییدیه گرفتم . اگه کاربر روی دکمه‌ی بله کلیک کنه ، عملیات شروع میشه و رکورد حذف میشه .
آخر سر توی تابعی که نتیجه برمیگرده توش (handleServerResponse)، بر اساس نتیجه‌ی برگشتی پیغام مناسبی بایدنمایش داده بشه. اما هیچ پیغامی نشون داده نمیشه
خیلی چیزا رو آزمایش کردم و آخر سر به این نتیجه رسیدم که چون دو تا پیغام میخواد پشت سر هم نمایش داده بشه ، این مشکل پیش میاد :(
حتی وقتی که پیغام تایید حذف رو برداشتم ، دیدم که پیغام دوم میاد !
واااقعا نمیدونم علتش چیه ! ممنون میشم کمکم کنید.
اینم قسمتی از کدمه :

```
function deleteFile(location, filename)//حذف فیزیکی و منطقی فایل
{
    $.prompt("آیا برای حذف فایل موجود اطمینان دارید ؟", {
        title: '<img src="Images/warning.png" />',
        buttons: { "بله": true, "خیر": false },
        focus: 2,
        submit: function (e, v, m, f) {
            if (v == true) {
                var getdate = new Date();  //Used to prevent caching during ajax call
                if (xmlhttp) {
                    $("p#vtip").fadeOut("slow").remove(); //محو شدن tooltip
                    var id = document.getElementById("id").value;
                    var i = '3';
                    params = "id=" + id + "&i=" + i + "&filename=" + filename + "&location=" +
location;
                    xmlhttp.open("POST", "FetchData/dbSearchDocument1.php", true);
                    xmlhttp.onreadystatechange = handleServerResponse4;
                    xmlhttp.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
                    xmlhttp.send(params);
                }
            }
            else {
            }
        }
    });
}
//-----------------------------------------------------------------------
function handleServerResponse4() {
    if ((xmlhttp.readyState == 1) || (xmlhttp.readyState == 2) || (xmlhttp.readyState == 3))//loading
    {
        document.getElementById("content").innerHTML = "<img src='Images/bigloading.gif'/>";
    }
    else if (xmlhttp.readyState == 4) //ready
    {
        if (xmlhttp.status == 200) {
            if (xmlhttp.responseText == 1) {
                $.prompt(". حذف فایل با موفقیت انجام شد ", {
```

```
                    title: '<img src="Images/warning3.png" />',
                    buttons: { "بستن": true }
                });
            }
            else {
                $.prompt("! حذف فایل با خطا مواجه شده است", {
                    title: '<img src="Images/warning2.png" />',
                    buttons: { "بستن": true }
                });
            }
        }
        else {
            alert("Error during AJAX call. Please try again");
        }
    }
}
//-----------------------------------------------------------------------
```

نویسنده: مژده
تاریخ: ۱۳۹۲/۰۳/۰۹ ۲۱:۵۴

من الان یک بار دیگه این مطلبو خوندم و به نظرم رسید که باید از callback استفاده کنم.
اما حتی این توی سایتم جواب نمیده :

```
function mycallbackfunc(v,m){
      $.prompt('i clicked ' + v);
}
$.prompt('Example 8',{ callback: mycallbackfunc });

});
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۰ ۰:۱۶

زمانیکه از jQuery استفاده می‌کنید، دیگر نباید از Ajax خام استفاده کنید. خود jQuery تابع سازگار با انواع و اقسام مرورگرها رو برای کار با Ajax داره. این مثال تست شده و مشکلی نداره:

```
using System.Web.Mvc;

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

        [HttpPost]
        public ActionResult Index(string postId, string filename)
        {
            return Content("ok");
        }
    }
}
```

```
@{
    ViewBag.Title = "Index";
    var postUrl = Url.Action(actionName: "Index", controllerName: "Home");
}
<h2>
    Index</h2>
<input type="button" id="btnDelete" value="delete" />
```

```
<div id="infoDiv">
    Some Info ....
</div>
@section JavaScript
{
    <script type="text/javascript">
        $(function () {
            $('#btnDelete').click(function () {
                $.prompt("آیا برای حذف اطلاعات موجود اطمینان دارید ؟",
             {
                 title: 'Title',
                 buttons: { "بله": true, "خیر": false },
                 focus: 2,
                 submit: function (e, v, m, f) {
                     //debugger;
                     if (!v) {
                         return;
                     }
                     var postId = 10;
                     var fileName = "test.jpg"
                     $.ajax({
                         type: "POST",
                         url: '@postUrl',
                         data: JSON.stringify({ postId: postId, fileName: fileName }),
                         contentType: "application/json; charset=utf-8",
                         dataType: "json",
                         complete: function (xhr, status) {
                             var data = xhr.responseText;
                             if (status === 'error' || !data || data == "nok") {
                                 $.prompt("حذف فایل با خطا مواجه شده است !",
                                    {
                                        title: 'Title',
                                        buttons: { "بستن": true }
                                    });
                             }
                             else {
                                 $("#infoDiv").fadeOut("slow").remove();
                             }
                         }
                     });
                 }
             });
            });
        });
    </script>
}
```

نویسنده: مژده
تاریخ: ۱۳۹۲/۰۳/۱۱ ۱۱:۳۵

سلام
ممنون که جواب دادید. خیلی لطف کردید
من تونستم با استفاده از یه js دیگه ، با callback مشکلمو حل کنم . اما حالا این js ، عنوان (title) نداره ! :D

از این سایت استفاده کردم :
http://www.shiguenori.com/material/jquery.impromptu/

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۱ ۱۱:۴۴

کدی که نوشته شد بر اساس نسخه GitHub بود.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۰۶ ۲۱:۱۱

سلام ، اگه بخوایم مقدار ورودی‌های اکشنمون به صورت داینامیک باشه چیکار باید کنیم؟
منظورم به عنوان مثال مقادیر این خط در مثال شماست

```
var postId = 10;
var fileName = "test.jpg"
```

چنین مدلی دارم که میخوام دکمه‌ی حذفش به صورت Jquery Ajax باشه

```
@foreach (var item in Model) {
    <tr>
        <td>
            @Html.DisplayFor(modelItem => item.Title)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.IsRead)
        </td>
        <td>
            @Html.ActionLink("حذف پیام", MVC.User.ActionNames.DeleteMessageSend, MVC.User.Name, new{
id=item.Id }, new { @class="btn btn-danger" })
        </td>
    </tr>
}
```

ولی مقدار id رو باید به اکشن بفرستم ! اونم تو یه حلقه Foreach . باید کدهای جی کوئری تو حلقه باشه؟ به نظرتون راه حل چیه؟ ممنون

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۰۶ ۲۱:۳۶

ببخشید تو این مطلب توضیح دادید.

عنوان: **بار کردن ساعت و تاریخ فعلی سرور با JQuery Ajax**
نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۷:۱۴
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, jQuery, Ajax

در این مطلب می‌خواهم شما را با نحوه بار گزاری ساعت و تاریخ سیستم سرور با استفاده از JQuery Ajax آشنا کنم.
در بعضی از سایتها با استفاده از جاوا اسکریپت تاریخ و ساعت جاری سیستم کلاینت به او نشان داه می‌شود.
این روش یک مزیت دارد: اول اینکه این کدها سمت کلاینت اجرا میشن و برای سرور بار اضافی ایجاد نمیکنن.
و یک عیب هم دارد: در صورتی که ساعت و تاریخ روی سیستم کلاینت تنظیم نباشد، همین ساعت و تاریخ نادرست برای او نمایش داده می‌شود. همین عیب می‌تواند باعث افت کیفیت وب سایت شود.

اما راهی هست که تاریخ و ساعت سیستم سرور برای کاربر نشان داده شود و آن هم استفاده از JQuery Ajax هست. به صورتی که هر ثانیه درخواستی برای یک handler فرستاده می‌شود و آن handler نیز ساعت و تاریخ روی سرور را باز می‌گرداند و این مقدار بازگشته شده را می‌توان در تگی از صفحه وب نمایش داد.

مثال: ابتدا یک صفحه aspx می‌سازیم و تگ زیر را در آن قرار می‌دهیم:

```
<p id="datetime"></p>
```

ساعت و تاریخ بار شده از سرور در این تگ باید نشان داده شود.

سپس کدهای اسکریپت زیر را می‌نویسیم:

```
var auto_referesh = setInterval
 (
     function()
    {
         $.post
         (
            "GetDateTime.ashx",
             function (result)
            {
                $('#datetime').html(result);
            }
        );
     }, 1000


 );
```

با نوشتن این کدها هر ثانیه یک بار، بوسیله Ajax درخواستی برای یک handler به اسم GetDateTime.ashx فرستاده می‌شود.
وظیفه این handler برگرداندن تاریخ و ساعت فعلی سیستم سرور است. بعد از دریافت مقدار این مقدار از این handler، آنرا در تگ با شناسه datetime قرار می‌دهیم.

کد استفاده شده در handler هم به این صورت است:

```
<%@ WebHandler Language="C#" Class="GetDateTime" %>

using System;
using System.Web;

public class GetDateTime : IHttpHandler {

    public void ProcessRequest (HttpContext context) {
        context.Response.ContentType = "text/plain";
        context.Response.Write(DateTime.Now.ToString());
    }

    public bool IsReusable {
```

```
        get {
            return false;
        }
    }

}
```

در انتها فایل ضمیمه این مثال را از این لینک دریافت کنید:

AjaxDateTime.zip

نظرات خوانندگان

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۰

سلام
روش شما خوبه اما عیبی که داره اینه که هر ثانیه ساعت از سرور میگیره و باعث سربار میشه
به نظر من روش بهتر اینه از همون کد طرف کلاینت استفاده بشه اما زمان اولیه از سرور خونده بشه بعد طرف کلاینت این ساعت خودمون اضافه کنیم و ساعت شبیه سازی کنیم
در صورتی هم که کاربر به صفحه دیگری بره یا صفحه رفرش کنه دوباره ساعت از سرور خونده میشه و الگوریتم ساعت طرف کلاینت به کار می‌افته
به نظر من این روش بهینه‌تر هست حالا باز ببینیم دوستان چه نظری دارن
موفق وموید باشید

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۳۷

سلام
بله این روش سربار اضافی داره. روشی که گفتید خیلی بهتر هست.
ولی خوب مسئله اینه که چطور میشه داده نوع DateTime در ASP.Net رو با jQuery خوند؟
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۵۵

روش دوم: خود وب سرور هم با درخواست‌های Head تاریخ رو ارسال می‌کنه. به این صورت هم قابل خواندن است:

```
<script src="Scripts/jquery.min.js" type="text/javascript"></script>
    <script type="text/javascript">
        $(document).ready(function () {
            try {
                var result = $.ajax({ 'type': 'HEAD', 'url': '/' }).success(function () {
                    var date1 = new Date(result.getResponseHeader('Date'));
                    alert(date1);
                });
            }
            catch (err) {
                //...
            }
        });
    </script>
```

و مهم‌ترین مزیتش این است که با تمام وب سرورهای استاندارد کار می‌کنه و فرقی نمی‌کنه کد شما PHP است یا ASP.NET.
+ این رو هم باید درنظر داشت که حین پردازش تاریخ دریافتی از وب سرور باید مسایل GMT را هم لحاظ کرد تا تاریخ و زمان دریافتی با زمان ایران تطابق پیدا کند.

نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۲:۵

سلام
همونطور که جناب فتح اللهی فرمودند، بهتر هست که یک بار زمان را از سرور دریافت کنید و سپس بروزرسانی را بوسیله جاوا اسکریپت انجام دهید.
بنده در ارتباط با این بحث، یک کلاس ساده نوشتم که می‌تونه عملیات بروزرسانی را انجام بده.

```
var MyTime = function () {
    var date;
    var tag;

    var init = function (hour, minute, seconds, tagId) {
        var constructor = getConstructorString(hour, minute, seconds);
        date = new Date(constructor);
        tag = document.getElementById(tagId);
        //console.log('MyTime : Init(%s, %s, %s, %s)', hour, minute, seconds, tagId);
    };

    var run = function () {
        update();
        window.setInterval(update, 1000);
        //console.log('MyTime : Run');
    };

    var update = function updateClock() {
        var h = date.getHours();
        var m = date.getMinutes();
        var s = date.getSeconds();

        s++;
        if (s == 60) { m++; s = 0; };
        if (m == 60) { h++; m = 0; };
        if (h == 13) h = 1;

        var constructor = getConstructorString(h, m, s);
        date = new Date(constructor);

        h = (h < 10) ? ("0" + h) : h;
        m = (m < 10) ? ("0" + m) : m;
        s = (s < 10) ? ("0" + s) : s;

        tag.innerHTML = h + ":" + m + ":" + s;
        //console.log('MyTime : update');
    };

    var getConstructorString = function (hour, minute, seconds) {
        //console.log('MyTime : getConstructorString');
        return '01/01/2000 ' + hour + ':' + minute + ':' + seconds;
    };

    return {
        Init: init,
        Run: run
    };
}
```

روش کار به این صورت هست که شما یک بار متد Init را به همراه پارامترهای (ساعت، دقیقه، ثانیه، آی‌دی تگ) فراخوانی می‌کنید، در نهایت برای اجرای ساعت، یک بار هم متد Run را فراخوانی می‌کنید.

مثال :

```
myTime = new MyTime();
myTime.Init(12, 59, 50, 'clock');
myTime.Run();
```

امیدوارم مفید بوده باشه .

عنوان: ساخت DropDownList های مرتبط به کمک jQuery Ajax در MVC
نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۷/۲۷ ۲۱:۵۸
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net MVC, jQuery, jQuery-Tips, Ajax, MVC

در پروژه‌های زیادی با مواردی مواجه می‌شویم که دو انتخاب داریم ، و یک انتخاب زیر مجموعه انتخاب دیگر است ، مثلا رابطه بین استان‌ها و شهرها ، در انتخاب اول استان انتخاب می‌شود و مقادیر DropDownList شهرها حاوی ، شهرهای آن استان می‌شود.
در پروژه‌های WebForm این کار به کمک Update Panel انجام می‌شد ، ولی در پروژه‌های MVC ما این امکان را نداریم و خودمان باید به کمک موارد دیگر این درخواست را پیاده سازی کنیم.
در پروژه ای که فعلا مشغول آن هستم ، با این مورد مواجه شدم ، و دیدم شاید بد نباشد راهکار خودم را با شما در میان بگذارم.
صورت مسئله :
کاربری نیاز به وارد کردن اطلاعات یک شیرالات بهداشتی (Tap) را دارد ، از DropDownList اول نوع آن شیر (Type) را انتخاب می‌کند ، و از DropDownList دوم ، که شامل گروه‌های آن نوع است ، گروه(Category) مورد نظر را انتخاب می‌کند. (در انتهای همین مطلب ببینید)
ساختار Tableهای دیتابیس به این صورت است:

رابطه ای از جدول Type به جدول Category.

به کمک Scaffolding یک کنترلر برای کلاس Tap (شیر آب) می‌سازیم ، به طور عادی در فایل Create.chtml مقدار گروه را به صورت DropDown نمایش می‌دهد، حال ما نیاز داریم که خودمان DropDown را برای Type ایجاد کنیم و بعد ارتباط این‌ها را بر قرار کنیم.

تابع اولی Create را این طوری ویرایش می‌کنیم :

```
public ActionResult Create()
        {
            ViewBag.Type = new SelectList(db.Types, "Id", "Title");
            ViewBag.Category = new SelectList(db.Categories, "Id", "Title");
            return View();
        }
```

همان طور که مشخص است ، علاوه بر مقادیر Category که خودش ارسال می‌کند ، ما نیز مقادیر نوع‌ها را به View مورد نظر ارسال می‌کنیم.

برای نمایش دادن هر دو DropDownList ویو مورد نظر را به این صورت ویرایش می‌کنیم :

```
<div>
            نوع
        </div>
        <div>
            @Html.DropDownList("Type", (SelectList)ViewBag.Type, "-- انتخاب ---", new { id = "rdbTyoe"
})
            @Html.ValidationMessageFor(model => model.Category)
        </div>

        <div>
            دسته بندی
        </div>
        <div>
            @Html.DropDownList("Category", (SelectList)ViewBag.Category, "-- انتخاب ---", new { id =
"rdbCategory"})
            @Html.ValidationMessageFor(model => model.Category)
        </div>
```

همان طور که مشاهده می‌کنید ، در اینجا DropDownList مربوط به Type که خودمان سمت سرور ،مقادیر آن را پر کرده بودیم نمایش می‌دهیم.

خب شاید تا اینجای کار ، ساده بود ولی میرسیم به اصل مطلب و ارتباط بین این دو DropDownList. (~~قبل از این قسمت حتما نگاهی به ساختار DropDownList یا همان تگ select بیندازید ، اطلاعات جی کوئری شما در این قسمت خیلی کمک حال شما است~~)

برای این کار ما از jQuery استفادی می‌کنیم ، **کار به این صورت است که** هنگامی که مقدار DropDownList اول تغییر کرد :
ما Id آن را به سرور ارسال می‌کنیم.
در آنجا Category هایی که دارای Type با Id مورد نظر هستند را جدا می‌کنیم
فیلدهای مورد نیاز یعنی Id و Title را می‌گیریم
و بعد به کمک Json مقادیر را بر می‌گردانیم

و مقادیر ارسالی از سرور را در optionهای DropDownList دوم (گروه‌ها) قرار می‌دهم

در ابتدا متدی در سرور می‌نویسیم ، کار این متد (همان طور که گفته شد) گرفتن Id از نوع Type استو برگرداندن Categoryهای مورد نظر. به این صورت :

```
public ActionResult SelectCategory(int id)
        {
            var categoris = db.Categories.Where(m => m.Type1.Id == id).Select(c => new { c.Id, c.Title
});
            return Json(categoris, JsonRequestBehavior.AllowGet);
        }
```

ابتدا به کمک Where مقادیر مورد نظر را پیدا می‌کنیم و بعد به کمک Select فیلد هایی مورد استفاده را جدا می‌کنیم و به کمک Json اطلاعات را بر می‌گردانیم. توجه داشته باشید که مقادیری که در categoris قرار می‌گیرد ، شبیه به مقادیر json هستند و ما می‌تانیم مستیما از این مقادیر در javascript استفاده کنیم.
در کلایت (همان طور که گفته شد) به این نیاز داریم که هنگاهی که مقدار در لیست اول تغییر کرد ، ای دی آن را به تابع بالا بفرستیم و مقادیر بازگشتی را در DropDownList دوم قرار دهیم : به این صورت :

```
$('#rdbTyoe').change(function () {
                jQuery.getJSON('@Url.Action("SelectCategory")', { id: $(this).attr('value') }, function
(data) {
```

```
                    $('#rdbCategory').empty();
                    jQuery.each(data, function (i) {
                        var option = $('<option></option>').attr("value",
data[i].Id).text(data[i].Title);
                        $("#rdbCategory").append(option);
                    });
                });
            });
```

توضیح کد :

با کمک تابع .change() از تغییر مقدار آگاه می‌شویم ، و حالا می‌توانیم مراحل کار را انجام دهیم. حالا وقت ارسال اطلاعات (Id مورد گزینه انتخاب شده در تغییر مقدار) به سرور است که ما این کار را به کمک jQuery.getJSON انجام می‌دهیم ، در تابع callback باید کار مقدار دهی انجام شود.

به ازای رکورد هایی که برگشت شده است که حالا در data قرار دارد به کمک تابع each لوپ می‌زنیم و هربار این کار را تکرار می‌کنیم.

ابتدا یک تگ option می‌سازیم

مقادیر مربوطه شامل Id که باید در attribute مورد نظر value قرار گیرد و متن آن که باید به عنوان text باشد را مقدار دهی می‌کنیم

option آماده شده را به DropDownList دومی (Category) اضافه می‌کنیم.

توجه داشته باشید که هنگامی که کار دریافت اطلاعات با موفقیت انجام شد ، مقادیر داخل DropDownList دومی ، مربوط به Category را به منظور بازیابی دوباره در

```
$('#rdbCategory').empty();
```

خالی می‌کنیم.

نتیجه کار برای من می‌شود این :

قبل از انتخاب :

بعد از انتخاب :

موفق و پیروز باشید

نظرات خوانندگان

نویسنده: فرشاد کریمی
تاریخ: ۱۳۹۱/۰۷/۲۷ ۲۳:۲۹

خیلی خوب و جامع بود

نویسنده: Hamid NCH
تاریخ: ۱۳۹۱/۰۷/۲۹ ۱۵:۲۶

مرسی مفید بود.

نویسنده: امیرسهیل
تاریخ: ۱۳۹۱/۰۹/۲۱ ۱۳:۲

سلام
ممنونم ، خیلی مطلب خوب مفیدی بود
یکی بزرگترین مشکلات ما رو حل کردید ، ما از روش شما در چند پروژه استفاده کردیم.

نویسنده: ابوالفضل
تاریخ: ۱۳۹۱/۱۰/۰۶ ۱۳:۱۸

سلام
برای من کار نکرد. به عبارتی متد getjson اجرا نمی‌شد.
تا قبل از متد jquery.getjson اجرا میشه. alert رو اجرا کرد. بعد دیگه اجرا نمیشه. تو firebug هم اساسا درخواستی json دیده نمیشه.
jquery-1.8.3

نویسنده: ابوالفضل
تاریخ: ۱۳۹۱/۱۰/۰۶ ۱۴:۱۰

مشکلم حل شد.
متد اجرا می‌شد. ولی در قسمت کنترلر حالت چرخشی پیش می‌اومد که نمی‌تونه به json تبدیل بشه. با new حل شد.

```
var Cities =from c in Cities
Select new {Id=c.Id, Name=c.Name};
return Json(Cities  , JsonRequestBehavior.AllowGet);
```

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۰/۰۷ ۹:۴۴

بدون new هم می‌شه خروجی linq گرفت؟

نویسنده: ابوالفضل
تاریخ: ۱۳۹۱/۱۰/۰۷ ۹:۵۶

بله، میشه خروجی گرفت. ولی همونطور که گفتم اگر آبجکت 1 به آبجکت 2 ارتباط داشته باشه و آبجکت 2 هم متقابلا از یک مسیر دیگه به آبجکت 1 ارتباط داشته باشه، در اینجا چرخه پیش میاد و json نمی‌تونه اون رو تبدیل کنه.
بنابراین با new یک آبجکت مختصر برای خروجی تولید کردم که شامل خصوصیات دیگر که باعث چرخش می‌شوند نباشد.

البته یه جایی خوندم در نگارش‌های جدید json.net حل شده این مشکل.

موفق باشید

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۰/۰۷ ۱۰:۱۴

در مثال شما که نشانی از اشیاء تو در تو نیست. حتی در سؤال شما هم نبود. خروجی return json با اشیاء تو در تو مشکلی ندارد. فقط خروجی آن‌را باید بتوانید سمت کلاینت پردازش کنید.

نویسنده: ابوالفضل
تاریخ: ۱۳۹۱/۱۰/۰۷ ۱۰:۲۵

اشیاء تو در تو با اشیاء چرخشی فرق میکنه دوست عزیز. من گفتم خروجی جیسون با اشیاء چرخشی مشکل دارد. اینجا

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۰/۰۷ ۱۰:۴۳

اون شیءایی که لینک دادی تو در تو هست و ارجاع حلقوی هم دارد. این نوع سؤال پرسیدن در کل صحیح نیست. افراد از راه دور که نمی‌تونن صفحه مونتیور شما رو مشاهده کنند. نه خطایی رو نوشتی و نه کلاس‌های مدلت مشخص بود.

نویسنده: سعید یزدانی
تاریخ: ۱۳۹۱/۱۱/۰۶ ۲۳:۷

با تشکر
میشه درباره‌ی selector این خط بیشتر توضیح بدبد

```
var option = $('<option></option>').attr("value", data[i].Id).text(data[i].Title);
```

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۱۱/۰۷ ۱:۷

ببین دوست من ، ابتدا نگاهی به ساختار dropdown لیست‌ها داشته باش.
مقداری که باید توی dropdown نشان داده بشند چیزی به این صورته :

```
<option value="value">Name</option>
```

مقداری که به سرور ارسال می‌شه مقدار داخل value هست و چیزی که نمایش داده می‌شه جای Name قرار می‌گیره
حالا ما ابن جا می‌خواهیم این قسمت‌ها رو خودمان درست کنیم.
در یک حلقه each به اضای مقادیری که از سرور گرفتیم loop می زنیم option را می‌سازیم و به dropdown اضافه می‌کنیم.
در مورد این قسمت هم باید بگم

```
.attr("value", data[i].Id).text(data[i].Title)
```

attr برای ایجاد کردن یک اتریبیوت به یک المان هست و از text هم برای مقداری دهی به خود المان به کار می‌رود.
مثلا برای یک رکورد که داری ای دی 100 و عنوان AmirHossein هست این option اینطوری ساخته می‌شه:

```
<option value="100">AmirHossein</option>
```

نویسنده: سعید یزدانی
تاریخ: ۱۳۹۱/۱۱/۰۷ ۱۹:۱۹

مرسی از جواب کاملتون

نویسنده: ziya
تاریخ: ۱۳۹۲/۰۲/۰۶ ۱۷:۴۵

1- من dropdownlist رو درست کردم ولی لازمه که توی کنترلرم مقدارش رو بدونم تا بتونم توی دیتابیسم ذخیره کنم ... چه جوری بدونم مقدارش چیه؟
2- با تگ input دوتا radio button درست کردم به مقدارهای این دو هم چه جوری از کنترلر دسترسی پیدا کنم (یعنی چطور متوجه شم که کاربر کودوم رو انتخاب کرده که براساس انتخابش باید توی جدولهای جداگونه ذخیره بشه)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۶ ۲۱:۴۵

- در مورد radio button list مطلب داریم در سایت.
- برای استفاده مقدماتی از DropDownList یک مثال کامل در سایت ASP.NET هست.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۲/۰۲/۰۶ ۲۱:۵۶

سلام
1 - به کامنت بالایی مراجعه کنید و ساختار DropDownList رو مطالعه کنید. همان طور که می‌بینید مقداری که به سرور ارسال می‌شود همان مقدار داخل Value است.
این شما هستید که DropDown رو می‌سازید، همان مقداری که در value قرار می‌دهید ارسال می‌شود.
2 - برای این هم شما رو ارجاع می‌دم به ساختار Radio button .
نمونه:

```
<form>
<input type="radio" name="sex" value="male">Male<br>
<input type="radio" name="sex" value="female">Female
</form>
```

شما در این جا دو radio دارید که دارای نام یکسال هستند و مقادیر Value آن‌ها تفاوت دارد.
در اینجا هم شما می‌توانید در کنترلر خود با name ان radio مقدار آن را در یافت کنید.
به عنوان مثال شما در کنترلر یکی از مقدارهای male و female را دریافت می‌کنید.

نویسنده: nasim
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۸:۴۷

سلام
من از linq, SelectListItem استفاده کردم.در view هم ازSelectListItem. ولی بقیه دقیقا مثل شما ولی جواب نمیده.مشکل چیه؟
لطفا راهنمایی کنید

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۹:۳۱

یه مطلب خوب در مورد دیباگ کدهای جی‌کوئری چند وقت پیش در سایت مطرح شده. شاید بتونه به شما در پیدا کردن مشکل کمک کنه.

نویسنده: nasim

تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۵:۲۹

من این کارو امتحان کردم مشکلی نداره.میدونید کلا کد جی کوئری من رو اجرا نمیکنه.با یک alert امتحان کردم.اگه کنترلر از نو scaffolding نباشه کار نمیکنه؟من زیاد بلد نیستم

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۶:۴۷

این قسمت jQuery.getJSON رو با Url درستی مقدار دهی کردید؟ این آدرس رو باید بر اساس برنامه خودتون تغییر بدید.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۶:۵۳

سلام
اگر امکان داره دقیق‌تر سوال رو بیان کنید و یا کدی که نوشتید اینجا بگذارید.

نویسنده: nasim
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱:۳۲

سلام
من از دو فیلد از 2 دیتابیس رو با دو dropdown نمایش دادم به صورت زیر:

```
public ActionResult Create()
        {
           Models.category cat = new Models.category();
            List<SelectListItem> list = cat.GetList().Select(p => new SelectListItem { Text =
p.category1, Value = p.ID.ToString() }).ToList();

            list[0].Selected = true;
            ViewData["Category1"] = list;

            Models.subcategory subcat = new Models.subcategory();


            List<SelectListItem> list1 = subcat.GetList().Select(p => new SelectListItem { Text =
p.subcategory1, Value = p.ID.ToString() }).ToList();

            list1[0].Selected = true;
            ViewData["Category2"] = list1;

            return View();
        }
-------------------------razor view-----------------------------------------------
  <div>
         @{ List<SelectListItem> lstCategories = (List<SelectListItem>)ViewData["category1"]; }
            @Html.DropDownList("dbcat", lstCategories, new { id="dbcat" })
        </div>
        <div>
            @{ List<SelectListItem> lstsubCategories = (List<SelectListItem>)ViewData["Category2"]; }
            @Html.DropDownList("dbsubcat", lstsubCategories, new { id="dbsubcat"})
```

قسمت کنترلر برای ارتباط دو dropdownlist

```
public ActionResult SelectCategory(int id)
        {


           studentDataContext  db= new studentDataContext();
            var subcat = db.subcategories.Where(m => m.ID == id).Select(c => new
            {
                c.ID,
                c.subcategory1
            });
            return Json(subcat, JsonRequestBehavior.AllowGet);
        }
```

وقسمت کد جی کوئری:

```
<script src="~/Scripts/jquery-1.7.1.min.js"></script>
```

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

    $('#dbcat').change(function () {

    jQuery.getJSON('@Url.Action("SelectCategory")', { id: $(this).attr('value') }, function (data) {
            $('#dbsubcat').empty(); jQuery.each(data, function (i) {
                var option = $('<option></option>')
             .attr("value", data[i].Id).text(data[i].Title); $("#dbcat").append(option);
            });
        });
   });

</script>
```

قسمت کد جی کوئری من اجرا نمیشه.حتی alert داخل

```
$('#dbcat').change(function ()
```

گذاشتم اجرا نشد.لطفا راهنمایی کنید
با تشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۸ ۹:۱۳

```
$(function () {
  // اینجا اسکریپت‌های جی‌کوئری باید تعریف شوند
});
```

قسمت کد جی‌کوئری رو باید در document ready قرار بدید

نویسنده: زرین
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۴:۲۸

سلام ،با تشکر از آموزشهای فوق العاده تون از درس1 تا این قسمت‌ها
من 2 ماهی هست که شروع کردم به یادگیری mvc و امروز رسیدم به این قسمت ، برنامه رو مثل شما نوشتم و یه کم مشکل دارم: قسمتی که باید dropdownList دوم که گروه است رو پر کنه کار نمی‌کنه! وارد کد jquery میشه و حتی متد SelectCategory رو هم فراخوانی میکنه اما dropdownList دومم رو خالی بر میگردونه!

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۴:۴۸

```
Select(c => new { c.Id, c.Title })
```

خروجی شما باید همین‌ها باشه در متد انتخاب گروه، البته مطابق کدهای جی‌کوئری نوشته شده

نویسنده: زرین
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۵:۱۰

ممنونم از پاسختون؛ جواب داد (-:

نویسنده: بهادر
تاریخ: ۱۳۹۲/۰۸/۱۹ ۲:۳۶

با سلام
من از کد زیر استفاده کردم تا موارد تکراری حذف بشه اما dropdown بدون value هست در هر آیتمش.

```
ViewBag.city = new SelectList(_db.regions.Select(r => r.region_main).Distinct(), "region_main");
```

لطفا راهنمایی بفرمایید
با تشکر

نویسنده: داود
تاریخ: ۱۳۹۳/۰۲/۲۵ ۱۱:۲۸

با سلام؛ من شبیه همچین کاری می‌خوام انجام بدم با این تفاوت که وقتی با استفاده از dropdownlistFor یکی رو انتخاب می‌کنیم property مخصوص به اون سمت viewmodel نیز تغییر کنه.
مکان dropdownlist خارج از beginform هستش و با binding ام وی سی مقدار set نمی‌شه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۵ ۱۲:۴۱

```
$('#dbcat').change(function () {
    var selectedItem = $(this).val();
    $.post( "-----url----", { item: selectedItem } );
});
```

+ عنصر خارج از فرم را نمی‌شود توسط submit معمولی به سرور ارسال کرد؛ مگر اینکه از Ajax استفاده کنید و عناصر مورد نیاز را از قسمت‌های مختلف صفحه جمع‌آوری و به سرور ارسال کنید؛ مانند کدهای فوق. یا اینکه در رخداد change، یک فیلد مخفی داخل فرم را با مقدار value انتخابی مقدار دهی کنید. به این ترتیب چون این فیلد مخفی، داخل فرم هست، قابلیت ارسال به سرور را از طریق دکمه‌ی استاندارد و غیر Ajax ایی submit، خواهد یافت.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۴/۱۹ ۱۴:۱۴

یک تجربه :
این سلکتور $('this).attr('value) برای کار با value در Option نال برمیگردوند و من من اینجوری استفاده کردم

```
$('#Cities').change(function () {
                jQuery.getJSON('@Url.Action("SelectTown")', { id: $(this).val() }, function (data) {
                    $('#Towns').empty();
                    jQuery.each(data, function (i) {
                        var option = $('<option></option>').val(data[i].ID).text(data[i].Name);
                        $("#Towns").append(option);
                    });
                });
            });
```

منظورم استفاده از .val به جای .attr('value') هستش. نمیدونم شاید بخاطر نسخه جدیدتر Jquery هست.

نویسنده: مهیاس هانی
تاریخ: ۱۳۹۳/۱۰/۰۳ ۹:۱۱

با سلام؛ من راجب dropDown در پیج Edit.cshtml سوال دارم. چطور می‌تونم SelectedItem رو تنظیم کنم، منظورم اینکه وقتی صفحه ویرایش لود میشه، منوی کشویی روی هموم آیتمی باشه که کاربر قبلا وارد کرده و در داخل DataBase ذخیره شده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۳ ۹:۲۰

از مقدار دهی چهارمین پارامتر SelectList برای تعیین selectedValue استفاده کنید:

```
public SelectList(IEnumerable items, string dataValueField, string dataTextField, Object selectedValue)
```

```
// in controller
ViewBag.Countries = new SelectList(countries.GetCountries(), "id", "countryName", "82" /* selected id
*/);

// in view
@Html.DropDownListFor(model => model.CountryId, (IEnumerable<SelectListItem>)ViewBag.Countries, new {
@class = "form-control" })
```

عنوان: **وب‌پارت اسلاید متحرک در شیرپوینت 2010**
نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۱/۰۸/۱۳ ۱۸:۵۰
آدرس: www.dotnettips.info
برچسب‌ها: SharePoint, WebPart, jQuery, Content Slider

در خلال تعدادی از پروژه‌هایی که در زمینه توسعه شیرپوینت 2010 درگیر بودم، یکی از وب‌پارت‌هایی که اکثر مشتریان درخواست می‌کردند وب پارت اسلاید متحرک بود. به نظر من داشتن این وب‌پارت ظاهرا خیلی برای مشتریان مهم بنظر می‌رسید، بهمین سبب در زیر به پیاده‌سازی آن اشاره می‌کنم.

ساخت لیست سفارشی

ابتدا یک لیست سفارشی بنام ContentSlides ایجاد می‌کنیم و ستونی از نوع Rich HTML به آن اضافه می‌کنیم.

ایجاد یک پروژه شیرپوینتی از نوع Visual Web Part

سپس یک پروژه شیرپوینت از نوع Visual Web Part در سایتی که لیست فوق در آن قرار دارد می‌سازیم.

در این مرحله نامگذاری‌های پیش‌فرض ویژوال استودیو که برای ویژوال وب‌پارت در نظر گرفته را به نام مناسب تغییر می‌دهیم.

افزودن پلاگین AnythingSlider

ابتدا پلاگین AnythingSlider را از این آدرس دریافت نمایید. سپس فایلهای anythingslider.css, anythingslider-ie.css, jquery.min.js, jquery.anythingslider.js و "default.png" را به ویژوال وب‌پارت اضافه می‌کنیم. برای اضافه کردن فایلهای این پلاگین، ابتدا فولدر "Layouts" به پروژه اضافه می‌نماییم و سپس فایلهای این پلاگین در این فولدر قرار می‌دهیم.

سپس در خط 129 و 155 و 181 فایل *anythingslider.css*

```
background: url("../images/default.png") no-repeat;
```

را به

```
background: url(“default.png") no-repeat;
```

تغییر دهید و شکل پروژه به صورت زیر خواهد شد.

در ادامه، بر روی "ContentSliderVisualWebPartUserControl.ascx" دابل کلیک کنید و کد زیر را به آن اضافه می‌کنیم.

```
<SharePoint:CssRegistration ID="AnythingSliderCssRegistration" runat="server"
Name="/_layouts/ContentSliderWebPart/anythingslider.css"></SharePoint:CssRegistration>
<SharePoint:CssRegistration ID="AnythingSliderCssRegistrationIE7" runat="server"
Name="/_layouts/ContentSliderWebPart/anythingslider.css" ConditionalExpression="lte IE
7"></SharePoint:CssRegistration>

<SharePoint:ScriptLink ID="JqueryScriptLink" runat="server"
Name="/_layouts/ContentSliderWebPart/jquery.min.js"></SharePoint:ScriptLink>
<SharePoint:ScriptLink ID="AnythingSliderScriptLink" runat="server"
Name="/_layouts/ContentSliderWebPart/jquery.anythingslider.js"></SharePoint:ScriptLink>
```

توجه کنید در کد بالا ما از کنترل‌های "SharePointCssRegistration" و "SharePointScriptLink" برای اضافه نمودن فایلهای

Css و JavaScript مورد نیاز و هم چنین از خصیصه "ConditionalExpression" برای اضافه کردن فایل "anythingslider-ie.css" برای مرورگر اینترنت اکسپلورر 7 به بالا استفاده کردیم.

ایجاد کوئری برای لیست سفارشی

برای ایجاد کوئری از کتابخانه jQuery بنام SPService که از این آدرس قابل دریافت است، استفاده می‌کنیم. فایل "jquery.SPServices.min.js" موجود در این کتابخانه را همانند سایر فایل‌های اضافه شده در قسمت قبل را به پروژه اضافه می‌کنیم.

برای استفاده از کتابخانه، کد زیر را در user control قسمت فبلی و در ادامه کدهای قبلی وارد می‌کنیم.

```
<SharePoint:ScriptLink ID="SPServicesScriptLink" runat="server"
Name="/_layouts/ContentSliderWebPart/jquery.SPServices.min.js"></SharePoint:ScriptLink>
```

و برای ایجاد کوئری کد زیر در user control وارد می‌کنیم.

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

$(document).ready(function () {
$().SPServices({
operation: "GetListItems",
async: false,
listName: "ContentSlides",
CAMLViewFields: "<ViewFields<FieldRef Name='SlidesContent' /></ViewFields>",
completefunc: function (xData, Status) {
$(xData.responseXML).SPFilterNode("z:row").each(function () {
var liHtml = "<li>" + $(this).attr("ows_SlidesContent") + "</li>";
$("#slider").append(liHtml);
});
}
});

/*-- Initialize AnythingSlider plugin --*/
    $('#slider').anythingSlider();

});
</script>

<ul id="slider" />
```

وظیفه این کد انجام کوئری بر روی لیست "ContentSlides" و ایجاد ساختار یک لیست جهت نمایش اسلایدها می‌باشد که آیتم‌های این لیست مقادیر ستون
"Slides Content" می‌باشد، شناسه این لیست "Slider" است.

توجه کنید: ما می‌توانیم CAML Queryهای پیشرفته‌ای برای اعمال فیلتر مناسب و چیدمان اسلایدها استفاده کنیم.

در ادامه، برای مقدار دهی اولیه به پلاگین بوسیله فراخوانی تابع

```
$('#slider').anythingSlider();
```

انجام می‌شود.

در پایان براحتی با Deploy نمودن Solution، امکان استفاده از وب پارت مهیاست.

برای دریافت کد این پروژه از این آدرس استفاده کنید.

نظرات خوانندگان

نویسنده: حسین
تاریخ: ۱۳۹۱/۱۲/۲۱ ۱۸:۴۷

با سلام خدمت امیر آقای عزیز
من فایل پروژه شما رو دانلود کردم و طبق مراحلی که توضیح دادین پیش رفتم اما متاسفانه موقع اضافه کردن وب پارت به صفحه پیغام خطای زیر را می‌دهد .لطفاً راهنمایی بفرمایید
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۲ ۰:۱۱

نکته‌ای عمومی در مورد تمام محصولات مایکروسافت:
«اگر کپی مطابق اصل خطای مشاهده شده را در جستجوی گوگل paste کنید و بر روی دکمه جستجو کلیک نمائید، در همان صفحه اول به جواب خواهید رسید»
این خطاها و این جملات بارها در سایت‌های مختلف تکرار شدن. چون تغییری هم نخواهند کرد، به عنوان یک مرجع رفع اشکال مورد استفاده قرار می‌گیرند.
مثلا همین عبارت بالا را که جستجو کنید به این راه حل خواهید رسید.

نویسنده: حسین
تاریخ: ۱۳۹۱/۱۲/۲۲ ۱۶:۴۰

با سلام و عرض تشکر از آقای نصیری عزیز

متاسفانه طبق اون آدرسی که شما گفتید عمل کردم ولی نشد.

عنوان: **نمایش رکوردها به ترتیب اولویت به کمک jQuery UI sortable در ASP.NET MVC**
نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۰:۴۷
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net MVC, jQuery, jQuery Ajax, UI, MVC

همان طور که می‌دانید کاربرد پذیری در خیلی از پروژه‌ها حرف اول رو می‌زند و کاربر دوست دارد کارهایی که انجام می‌دهد خیلی راحت و با استفاده از موس باشد.یکی از کار هایی که در اکثر پروژه‌ها نیاز است ، چیدمان ترتیب رکوردها است. ما می‌خواهیم در این پست ترتیبی اتخاذ کنیم که کاربر بتواند رکوردها را به هر ترتیبی که دوست دارد نمایش دهد.

از توضیحاتی که قبلا دادم مشخص است که این کار احتمالا در ASP.NET WebForm کار سختی نیست ولی این کار باید در MVC از ابتدا طراحی شود.

طرح سوال : یک سری رکورد از یک Table داریم که می‌خواهیم به ترتیب وارد شدن رکوردها نباشد و ترتیبی که ما می‌خواهیم نمایش داده شود.

پاسخ کوتاه : خب باید ابتدا یک فیلد (برای اولویت بندی) به Table اضافه کنیم بعد اون فیلد رو بنا به ترتیبی که دوست داریم رکوردها نمایش داده شود پر کنیم (Sort می کنیم) و در آخر هم هنگام نمایش در View رکوردها را بر اساس این فیلد نمایش می‌دهیم.

(این پست هم در ادامه پست قبلی در همان پروژه است و از همان Table ها استفاده شده است)

اضافه کردن فیلد :

ابتدا یک فیلد به Table مورد نظر اضافه می‌کنیم. من اسم این فیلد رو Priority گذاشتم. Table من چنین وضعیتی دارد.

AMIRHOSSEIN-PC.Khazar - dbo.Types

Column Name	Data Type	Allow Nulls
Id	int	☐
Title	nvarchar(200)	☑
Description	text	☑
ImagePath	varchar(500)	☑
Priority	int	☑
IsActive	bit	☐
CreatedOn	datetime	☐
CreatedBy	uniqueidentifier	☐
ModifiedOn	datetime	☑
ModifiedBy	uniqueidentifier	☑
IsDeleted	bit	☐
		☐

افزودن فایل‌های jQuery UI :

در این مرحله شما نیاز دارید فایل‌های مورد نیاز برای Sort کردن رکوردها را اضافه کنید. شما می‌توانید فقط فایل‌های مربوط به Sortable را به صفحه خودتان اضافه کنید و یا مثل من فایل هایی که حاوی تمام قسمت‌های jQuery UI هست را اضافه کنید.

من برای این کار از Section استفاده کردم ، ابتدا در Head فایل Layout دو Section تعریف کردم برای CSS و JavaScript . و فایل‌های مربوط به Sort کردن را در صفحه ای که باید عمل Sort انجام بشود در این Section ها قرار دادم.

فایل Layout

```
<head>
    <meta charset="utf-8" />
    @RenderSection("meta", false)
    <title>@ViewBag.Title</title>
    <link href="@Url.Content("~/Content/~Site.css")" rel="stylesheet" type="text/css" />
    <link href="@Url.Content("~/Content/redactor/css/redactor.css")" rel="stylesheet" type="text/css"
/>
    <link href="@Url.Content("~/Content/css/bootstrap-rtl.min.css")" rel="stylesheet" type="text/css"
/>
    @RenderSection("css", false)
    <script src="@Url.Content("~/Scripts/jquery-1.8.2.min.js")" type="text/javascript"></script>
    <script src="@Url.Content("~/Scripts/modernizr-1.7.min.js")" type="text/javascript"></script>
    <script src="@Url.Content("~/Content/js/bootstrap-rtl.js")" type="text/javascript"></script>
    <script src="@Url.Content("~/Content/redactor/redactor.min.js")" type="text/javascript"></script>
    <script src="@Url.Content("~/Scripts/jquery.unobtrusive-ajax.min.js")"
type="text/javascript"></script>
    @RenderSection("js", false)

</head>
```

فایل Index.chtml در پوشه کنترلر Type

```
@model IEnumerable<KhazarCo.Models.Type>
@{
    ViewBag.Title = "Index";
    Layout = "~/Areas/Administrator/Views/Shared/_Layout.cshtml";
}
@section css
{<link href="@Url.Content("~/Content/themes/base/jquery-ui.css")" rel="stylesheet" type="text/css" />
}
@section js
{
    <script src="@Url.Content("~/Scripts/jquery-ui-1.9.0.min.js")" type="text/javascript"></script>
}
```

در آخر فایل Index.chtml به اینصورت شده است:

```
<h2>
    نوع ها</h2>
<p>
    @Html.ActionLink("ایجاد یک مورد جدید", "Create", null, new { @class = "btn btn-info" })
</p>
<table>
    <thead>
        <tr>
            <th>
                عنوان
            </th>
            <th>
                توضیحات
            </th>
            <th>
                فعال
            </th>
            <th>
            </th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model.OrderBy(m => m.Priority))
        {
            <tr id="@item.Id">
                <td>
                    @Html.DisplayFor(modelItem => item.Title)
                </td>
                <td>
                    @(new HtmlString(item.Description))
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.IsActive)
                </td>
                <td>
                    @Html.ActionLink("ویرایش", "Edit", new { id = item.Id }, new { @class = "btnEdit
label label-warning" })
                    |
                    @Html.ActionLink("مشاهده", "Details", new { id = item.Id }, new { @class =
"btnDetails label label-info" })
                    |
                    @Html.ActionLink("حذف", "Delete", new { id = item.Id }, new { @class = "btnDelete
label label-important" })
                </td>
            </tr>
        }
    </tbody>
</table>
```

** توجه داشته باشید که من به هر tr یک id اختصاص داده ام که این مقدار id همان مقدار فیلد Id همان رکورد هست ، ما برای مرتب کردن به این Id نیاز داریم (خط 25).

افزودن کدهای کلاینت :

حالا باید کدی بنویسم که دو کار را برای ما انجام دهد : اول حالت Sort پذیری را به سطرهای Table بدهد و دوم اینکه هنگامی که ترتیب سطرهای تغییر کرد ما را با خبر کند:

```
<script type="text/javascript">
    $(function () {
        $("table tbody").sortable({
            helper: fixHelper,
            update: function (event, ui) {
                jQuery.ajax('@Url.Action("Sort", "Type", new { area = "Administrator" })', {
                    data: { s: $(this).sortable('toArray').toString() }
                });
            }
        }).disableSelection();
    });
    var fixHelper = function (e, ui) {
        ui.children().each(function () {
            $(this).width($(this).width());
        });
        return ui;
    };
</script>
```

توضیح کد :

در این کد ما حالت ترتیب پذیری را به Table می دهیم و هنگامی که عمل Update در Table انجام شد تابع مربوطه اجرا می‌شود. ما در این تابع، ترتیب جدید سطرها را می‌گیریم (** به کمک مقدار Id که به هر سطر دادیم ، این مقدار Id برابر بود با Id خود رکورد در Database) و به کمک jQuery.ajax به تابع Sort از کنترلر Type در منطقه (Administrator) area ارسال می‌کنیم و در آنجا ادامه کار را انجام میدهیم.

تابع fixHelper هم به ما کمک می‌کند که هنگامی که سطرها از جای خود جدا می‌شوند ، دارای عرض یکسانی باشند و عرض آن‌ها تغییری نکند.

افزودن کد Server :

حالا باید تابع Sort که مقادیر را به آن ارسال کردیم بنویسم. من این تابع را بر اساس مقداری که از کلاینت ارسال می‌شود اینگونه طراحی کردم .

```
public EmptyResult Sort(string s)
{
    if (s != null)
    {
```

s "1,2,3,4,6,5"

```
public EmptyResult Sort(string s)
        {
            if (s != null)
            {
                var ids = new List<int>();
                foreach (var item in s.Split(','))
```

```
                {
                    ids.Add(int.Parse(item));
                }
                int intpriority = 0;

                foreach (var item in ids)
                {
                    intpriority++;
                    db.Types.Single(m => m.Id == item).Priority = intpriority;
                }
                db.SaveChanges();
            }
            return new EmptyResult();
        }
```

در ابتدا مقادیر Id که از کلاینت به صورت String ارسال شده است را می‌گیریم و بعد به همان ترتیب ارسال در لیستی از int قرار می‌دهیم ids.

سپس به اضای هر رکورد Type مقدار اولویت را به فیلدی که برای همین مورد اضافه کردیم Priority اختصاص می‌دهیم . و در آخر هم تغییرات را ذخیره می‌کنیم. (خود کد کاملا واضح است و نیاری به توضیح بیشتر نیست)

حالا باید هنگامی که لیست Type ها نمایش داده می‌شود به ترتیب (OrderBy) فیلد Priority نمایش داده شود پس تابع Index را اینطور تغییر می‌دهیم.

```
public ViewResult Index()
        {
            return View(db.Types.Where(m => m.IsDeleted == false).OrderBy(m => m.Priority));
        }
```

این هم خروجی کار من:

نوع ها

ایجاد یک مورد جدید

عنوان	توضیحات	فعال	
مدل اهرمی	این مدل از طرح هاي جديد و نوين بهره مي برد.	☑	ویرایش | مشاهده | حذف
مدل کلاسیک	طرح هاي قديمي شيرآلات	☑	ویرایش | مشاهده | حذف
سینک شاوری	توضيحاتي در مورد سينک هاي شاوري	☑	ویرایش | مشاهده | حذف
دسته ها و پوسته ها	توضيحاتي در مورد دسته ها و پوسته ها	☑	ویرایش | مشاهده | حذف
علم های دوش	توضيحاتي در مورد علم هاي دوش	☑	ویرایش | مشاهده | حذف
سایر محصولات		☑	ویرایش | مشاهده | حذف

این عکس مربوط به است به قسمت مدیریت پروژه شیرآلات مرجان خزر .

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۱:۱۲

« این کار احتمالا در ASP.NET WebForm کار سختی نیست »

اتفاقا کار ساده‌ای نیست و همین مراحل باید طی شود. ضمن اینکه کار با jquery ajax در آنجا به این یک دستی نیست. نیاز است در code behind فرم، متد وب سرویس مانندی به صورت استاتیک تعریف شود (که خودش سبب می‌شود تا دسترسی به اعتبار سنجی توکار مبتنی بر فرم‌ها محدود شود) یا اینکه از یک هندلر مجزا بجای یک اکشن متد کمک گرفته شود ... خلاصه خیلی داستان دارد.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۱:۱۶

من خودم این کار رو در webform انجام ندادم و خیلی هم با سختی هاش آشنایی ندارم.
ولی فکر کردم شاید با update panel بشه این کار رو راحت انجام داد.
خب نظرتون راجب به راه حل من چیه؟

راه حل خوبی ارائه دادم؟ متن قابل فهم بود؟

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۱:۲۰

بسیار عالی.

فقط در مورد update panel ... تنها کاری که انجام می‌دهد کپسوله کردن ارسال مقادیر به سرور است به صورت ajax سازگار با کنترل‌های دارای view state . بیشتر از این کاری انجام نمی‌دهد. مابقی آن اگر یک سری کنترل در toolkit آن موجود باشد برای این کارها یا خیر. این toolkit هم محدود است و آنچنان به روز نمی‌شود.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۱:۲۳

ممنونم از پاسختون
از اینکه سوییچ کردم روی MVC خوشحالم :)
البته یکی از دلایلی هم که این کار رو کردم همین راحتی استفاده Ajax بود.

نویسنده: امیر آشنا
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۲:۴

سلام
مطلب مفیدی بود ولی می‌تونستید بیشتر توضیح بدید.
برای ما افراد مبتدی درک بعضی قسمت‌ها کمی مشکل هست.
ولی باز هم ممنونم

نویسنده: امیرحسین مرجانی

تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۲:۷

سلام
بفرمایید کدوم قسمت‌ها تا بیشتر توضیح بدم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۲:۲۱

پیشنیازهای مطلب جاری:
- در مورد RenderSection : (^)
- مقدمه‌ای بر jQuery Ajax در MVC : (^)
- db.SaveChanges : کل مباحث Entity framework سایت (^)
- EmptyResult و کلا خروجی‌های اکشن متدها: (^)
- علت استفاده از Sort@ Url.Action در حین آدرس دهی: (^)

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۲:۲۳

واقعا تشکر می‌کنم بابت این لیستی که ارائه کردید.
بحث تکمیل شد
به نظر شما باید بیشتر به جزییات اهمیت داده می‌شد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۲:۴۱

ضرورتی نداره. چون واقعا به اندازه لیستی که عنوان شد نیاز به پیشنیاز درک این مطالب هست و فرصت تکرار آن‌ها نیست. این مطالب جدید، یک سری مطالب تکمیلی هستند نه مطالب پایه و از صفر.

نویسنده: محسن موسوی
تاریخ: ۱۳۹۱/۰۹/۰۹ ۰:۲۴

با استفاده از Web API Controller تقریبا به همین راحتی امکانپذیره. تغییر زیادی هم نیاز نداره.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۹/۰۹ ۰:۳۷

ممنونم آقای موسوی
قابل توجه آقا سعید (احتمالا این مطلب برای شما مفیده)

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۹/۰۹ ۰:۵۲

- نیاز به vs2012 داره.
- نمیشه یک فرم رو strongly typed تعریف کرد مثل مثال بالا:

```
@model IEnumerable<KhazarCo.Models.Type>
```

نویسنده: محسن موسوی
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۰:۲۱

بحث در مورد UI نبود.بحث در مورد استفاده کردن بود.
در ضمن در Web Formها نیز از امکاناتی شبیه به MVC می‌توان بهره برد.(^)

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۰:۴۶

فقط mvc هست که امکان استفاده از چند view engine را با هم دارد.

صفحه‌ای که لینک دادید مربوط است به web pages asp.net و نه web forms. این web pages برای کار با webmatrix طراحی شده.

نویسنده: محسن موسوی
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۱:۴

- استفاده از این قابلیت در Web Page ها توسط VS
- استفاده از این قابلیت در Web Form ها توسط VS
هر دو مورد توسط VS استفاده شده است.

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۱:۴۵

```
<iframe src="/twitter" frameborder="0" height="400" width="270" scrolling="no"></iframe>
```

فلسفه و کاربرد web pages متفاوت است. در مورد وب فرم‌ها هم نمی‌تونید از razor استفاده کنید چون در یک فایل نمی‌شود از دو موتور view استفاده کرد. موتور view وب فرم‌ها، نامش همین web forms view engine است و قابل تعویض هم نیست (برخلاف MVC). در مقاله‌ای که لینک دادید، داره از یک *iframe* استفاده می‌کنه برای الحاق razor.

به علاوه در مورد Web API که کلا مطلب جدیدی نیست. نسخه ساده شده WCF است.

نویسنده: محسن موسوی
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۲:۲۷

دوست عزیز
بحث ما در مورد توانایی انجام موارد فوق الذکر بود. نه اینکه حالا چون ...
به توانایی‌های MVC شکی نیست.مسئله اصلی اینه که آیا پست جاری را میتوان به راحتی با WebPage و یا WebForm انجام داد؟!
چه از طریق قابلیت‌های ASP.NET Web Form و ASP.NET Web Page راحته.
استفاده از Razor چه از طریق Jquery و یا Iframe (نظر قبلی)و یا به طور مستقل و یا روش‌های دیگر در ASP.NET Web Form و ASP.NET Web Page
و استفاده معمولی با توانایی‌های DataBind
یا بطور کامل از Razor View Engine در Web Page
در نهایت کار سختی نیست.

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۲:۴۳

razor بی‌جهت اینجا طرح شد. من در مورد strongly typed view سطری رو نوشتم. این نوع viewها مستقل از نوع view

engine هستند. در mvc حتی با موتور web forms هم میشه یک چنین viewهای تحت نظر کامپایلری رو با پشتیبانی کامل از intelisense داشت.

عنوان: **ایجاد قسمت‌های Toggle در سایت با jQuery**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۰/۲۷ ۲۱:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: MVC, jQuery, ASP.Net 4.5, Design

البته قبلش بگم که عنوان بهتری به ذهنم نرسید.
بسیاری از مواقع پیش می‌آید که در سایت خود بخواهیم کادری داشته باشیم که با کلیک بروی آن ظاهر و با کلیک دوباره بروی آن محو شود. مانند تصویر زیر

سپس با کلیک بروی قسمت مشخص شده از تصویر بالا تصویر مانند زیر ظاهر شود.

در این نوشته قصد داریم کادری به این صورت حالا به هر منظوری طراحی نماییم.

برای کار سه قسمت کد داریم:

کدهای طراحی قسمت مورد نظر در صفحه وب
نوشتن کدهای CSS مربوطه
نوشتن کدهای jQuery

در مرحله اول ابتدا صفحه وب خود را به نحو زیر ایجاد می‌نماییم.

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>کادر لغزان با jQuery</title>
    <script src="Scripts/jquery-1.7.1.min.js"></script>
    <link href="CSS/site.css" rel="stylesheet" />
</head>
<body>
    <div id="loginPanel">
        <div style="height: auto;" id="login">
            <div>
                <div>
                    <br />
                    محتویات دلخواه خود را در این قسمت قرار دهید
                </div>
            </div>
            <div><a href="#" id="closeLogin"></a></div>
```

```
        </div>
        <div id="container">
            <div id="top">
                <!-- login -->
                <ul>
                    <li> </li>
                    <li><a id="toggleLogin" href="#">پانل باز شو</a></li>
                </ul>
                <!-- / login -->
            </div>
            <!-- / top -->

        </div>
    </div>
    <div id="main">
        محتویات سایت در این قسمت قرار می‌گیرد
    </div>
    </body>
</html>
```

در صفحه ایجاد شده قسمتی را برای نگهداری پانل مورد نظر قرار دادیم و در div با نام loginContent مواردی را که می‌خواهیم در پانل مربوطه نمایش داده شود، قرار می‌دهیم، نگهدارنده کل قسمت مربوطه (کادر لغزان می‌باشد)، و قسمت قسمتی است که دکمه یا عنوان مورد نظر برای باز شدن یا بستن کادر استفاده می‌شود.

در مرحله دوم کدهای CSS بخش‌های مورد نظر (جهت رنگ و تصاویر و شکل و شمایل کادر مورد نظر) را مانند زیر ایجاد می‌کنیم.

```
body {
     margin:0;
     padding:0;
     width:100%;
     background: #e9e9e9 url(images/header_bg.gif) top repeat-x;
     direction: rtl;
}
html {
     padding:0;
     margin:0;
}

#main {margin-top: 100px;}

#loginPanel {
    margin: 0px;
    position: absolute;
    overflow: hidden;
    height: auto;
    z-index: 3000;
    width: 100%;
    top: 0px;
    color: #fff;
}
#top {
    background: url(images/login_top.jpg) repeat-x 0 0;
    height: 38px;
    position: relative;
}
#top ul.login {
    display: block;
    position: relative;
    float: right;
    clear: right;
    height: 38px;
    width: auto;
    margin: 0;
    right: 150px;
    color: white;
    text-align: center;
    background: url(images/login_r.png) no-repeat right 0;
    padding-right: 45px;
}
#top ul.login li.left {
    background: url(images/login_l.png) no-repeat left 0;
    height: 38px;
    width: 45px;
    padding: 0;
    margin: 0;
    display: block;
    float: left;
```

```
}
#top ul.login li {
    text-align: left;
    padding: 0 6px;
    display: block;
    float: left;
    height: 38px;
    background: url(images/login_m.jpg) repeat-x 0 0;
}
#top ul.login li a {
    color: #fff;
    text-decoration: none;
}
#top ul.login li a:hover {
    color: #ff0000;
    text-decoration: none;
}
#login {
    width: 100%;
    color: white;
    background: #1E1E1E;
    overflow: hidden;
    position: relative;
    z-index: 3;
    height: 0px;
}
#login a {
    text-decoration: none;
    color: #fff;
}
#login a:hover {
    color: white;
    text-decoration: none;
}
#login .loginContent {
    width: 900px;
    height: 80px;
    margin: 0 auto;
    padding-top: 25px;
    text-align: right;
}
#login .loginClose {
    display: block;
    position: absolute;
    right: 15px;
    top: 10px;
    width: 70px;
    text-align: left;
}
#login .loginClose a {
    display: block;
    width: 100%;
    height: 20px;
    background: url(images/button_close.jpg) no-repeat right 0;
    padding-right: 10px;
    border: none;
    color: white;
}
#login .loginClose a:hover {
    background: url(images/button_close.jpg) no-repeat right -20px;
}
.cen { text-align: center;}
.w_100p{ width: 100%;}
```

خوب تا اینجای کار فقط کادر با قالب مورد نظر ایجاد شد، برای اینکه عمل مورد نظر انجام شود با استفاده از تکنیک‌های jQuery به صورت زیر کار را به پایان می‌رسانیم. در انتهای صفحه اسکریپت زیر را قبل از قسمت </body> می‌نویسیم.

```
<script type="text/javascript">
        $(document).ready(function () {
            $("#login").hide(0);
            $("#toggleLogin").click(function () {
                $("#login").slideToggle("slow");
            });
            $("#closeLogin").click(function () {
                $("#login").slideUp("slow");
            });
        });
```

```
</script>
```

با نوشتن این اسکریپت بعد از لود صفحه مورد نظر ابتدا کادر ما مخفی می‌شود، سپس برای دکمه (یا هر المانی که می‌خواهیم با کلیک روی آن کادر بسته یا باز شود) کد کلیک می‌نویسیم که با کلیک بروی آن عمل اسلاید (باز یا بسته شدن) رخ دهد. در نهایت در رویداد کلیک لینک close تعیین می‌کنیم که کادر به آرامی بسته شود.

مثال کامل از ^ قابل دانلود است

لینک‌های کمکی جهت آشنایی بیشتر با توابع استفاده شده: slideUp

slideToggle

نظرات خوانندگان

نویسنده: مرتضی مختاری
تاریخ: ۱۳۹۱/۱۲/۲۱ ۲۰:۲۸

سلام آقای فتح الله ممنون از پستی که قرار دادید بنده کد شما رو تویه سایتم قرار دادم ولی یک مشکلی که وجود داره وقتی رویه دکمه ارسال که درون یک آپدیت پنل هستش کلیک میکنم قسمت toggle بسته میشه چطوری میشه این رو موقع post back باز نگه داشت .با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۱ ۲۳:۵۵

مراجعه کنید به مطلب نمایش پیغامی به کاربر در هنگام استفاده از MS Ajax و همچنین استفاده‌ی همزمان از آپدیت پنل ASP.Net و پلاگین‌های جی‌کوئری

عنوان: **استفاده از WCF Data Services در برنامه های وب به کمک JQuery**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۱/۰۵ ۱۷:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: Web Service, jQuery, OData, WCF Data Services, JSON

در مطالب قبلی با پروتکل OData و WCF Data Service و قراردادهای کوئری نویسی آن آشنا شدید. حال می‌خواهیم با استفاده از Jquery به داده‌های وب سرویس WCF Data Service دسترسی یابیم. اما پیش نیازهای لازم است

پیش نیاز اول : دسترسی به خروجی Json وب سرویس WCF Data

خروجی پیش فرض وب سرویس WCF Data Services ساختار Xml دارد پس می‌بایست وب سرویس را متوجه سازیم که ما با خروجی Json نیاز داریم. از نسخه 5 به بعد اگر MaxProtocolVersion را بر روی V3 قرار دهیم دیگر با Accept Header برابر application/json کار نخواهد کرد و می‌بایست از application/json;odata=verbose استفاده نمود یا نسخه پروتکل را بر روی V2 یا پایین‌تر تنظیم کنید. علاوه بر آن کتابخانه‌های و قطعه کدهای تهیه شده است که با پارامتر $format این کار را برای ما انجام می‌دهد در زیر آدرس دو نمونه آورده شده است.

DataServicesJSONP

WCF Data Services Toolkit

قطعه کد اول یک Attribute است که با اضافه کردن آن به بالای کلاس WebService و استفاده از پارامتر $format=json در آدرس وب سرویس این کار را برای ما انجام می‌دهد.

```
[JSONPSupportBehavior]
public class Northwind : DataService<NorthwindEntities>
```

و نمونه آدرس

```
http://localhost:8358/Northwind.svc/Products?$format=json
```

دومی کتابخانه ای است که مانند روش اول عمل می‌کند اما به جای ارث برای از کلاس DataService می‌بایست از کلاس ODataService استفاده نماییم.

نکته: در صورتی که بخواهیم از نسخه V3 استفاده نماییم Accept Header را باید به application/json;odata=verbose تغییر دهیم

```
public class Northwind : ODataService<NorthwindEntities>
```

استفاده از WCF Data Services به کمک Jquery

تابع getJSON مخصوص درخواست‌های است خروجی بصورت json برگردانده می‌شود اما با نسخه V3 سازگار نمی‌باشد و از روش پارامتر $format می‌توان استفاده نمود

```
$.getJSON("Northwind.svc/Products?$format=json", function (data) {
                $.each(data.d, function (i, item) {
                    $("<p/>").html(item.Product_Name + " " + item.Unit_Price).appendTo("#products");
                });
            });
```

همچنین از تابع Ajax که امکانات بیشتری را در اختیارمان قرار می‌دهد به راحتی می‌توان استفاده نمود به مثال زیر دقت کنید:

```
$.ajax('Northwind.svc/Products', {
                dataType: "json",
                beforeSend: function (xhr) {
                    xhr.setRequestHeader("Accept", "application/json;odata=verbose");
                    xhr.setRequestHeader("MaxDataServiceVersion", "3.0");
                },
                success: function (data) {
                    $.each(data.d, function (i, item) {
                        $("<p/>").html(item.Product_Name + " " +
item.Unit_Price).appendTo("#products");
                    });
                }
            });
```

با اسفاده از beforeSend مقدار Accept Header و MaxDataServiceVersion را تعیین نموده ایم.

بنابراین به کمک قرارداده‌های کوئری نویسی که در مطالب قبلی گفته شد می‌توان با استفاده از Url تابع Ajax به داده مورد نظر خود رسید.

نظرات خوانندگان

نویسنده: مرادی
تاریخ: ۱۳۹۱/۱۱/۰۵ ۱۷:۴۲

با سلام، بهتر نیست از jay data یا از breeze.js استفاده کنید ؟

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۱/۰۵ ۱۸:۳۴

در مطلب بعدی به این دو اشاره خواهم کرد قطعا امکانات بیشتری در اختیارمان قرار می‌دهد

نویسنده: abdali
تاریخ: ۱۳۹۲/۰۴/۱۱ ۱۵:۱۹

لطف میکنید کد رو قرار بدین ، چند بار امتحان کردم با خطا مواجه شدم . مرسی

عنوان: آموزش (jQuery) جی کوئری #1
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۱/۲۲ ۲۳:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, jQuery, Web Design

با سلام خدمت دوستان عزیز
تصمیم گرفتم در طی چندین پست در حد توانم به آموزش jQuery بپردازم. (مطالب نوشته شده برداشت ازادی از کتاب jQuery in action است)

جی کوئری (jQuery) چیست؟

jQuery یک کتابخانه بسیار مفید برای جاوا اسکریپت است. بسیار ساده و کارآمد است و مشکل جاوا اسکریپت را برای تطابق با مرورگرهای اینترنتی مختلف برطرف نموده است؛ یادگیریjQuery بسیار آسان است. در جی کوئری کد جاوا اسکریپت از فایل HTML جدا شده و بنابراین کنترل کدا و بینه‌سازی آنا بسیار ساده‌تر خواد شد. توابعی برای کار با AJAX فرام نموده و در این زمینه نیز کار را بسیار ساده کرده است. در جی کوئری می‌توان از خصوصیت فراخوانی زنجیره‌ای متدا استفاده نمود و این باعث می‌شود چندین کد فقط در یک سطر قرار گیرد و در نتیجه کد بسیار مختصر گردد. در مقایسه با سایر ابزارهایی که تاکید عمده‌ای بروی تکنیک‌های هوشمند جاوا اسکریپت دارند، هدف جی کوئری تغییر تفکر سازندگان وب سایت‌ها، به ایجاد صفحه‌هایی با کارکرد بالا می‌باشد. به جای صرف زمان برای مقابله با پیچیدگی‌های جاوا اسکریپت پیشرفته، طراحان می‌توانند با استفاده از زمان و دانش خود در زمینه‌ی CSS، HTML، XHTML و جاوا اسکریپت‌های ساده، عناصر صفحه را مستقیما دستکاری کنند و از همین طریق تغییرهای گشترده و سریعی انجام دهند.

نکته : برای استفاده از جی کوئری باید HTML و CSS و جاوا اسکریپت آشنایی داشته باشید.

چگونه از جی کوئری استفاده کنیم؟

برای استفاده از جی کوئری باید ابتدا فایل آن را از سایت آن دانلود کرده و در پروژه خود استفاده نمایید. البته روش‌های دیگری برای استفاده از این فایل وجود دارد که در آینده بیشتر با آن آشنا خواهیم شد. برای استفاده از این فایل در پروژه باید به شکل زیر آن را به صفحه HTML خود معرفی کنیم.

```
<html>
   <head>
       <script type="text/javascript" src="jquery-1.9.1.min.js"></script>
   </head>
   <body>
   </body>
</html>
```

سپس بعد از معرفی خط فوق در قسمت head صفحه باید کدهای خود را در یک تگ script بنویسیم.

کوتاه کردن کد : هر زمان شما خواسته باشید کارکرد یک صفحه وب را پویاتر کنید، در اکثر مواقع به ناچار این کار از طریق عناصری بروی صفحه انجام داده اید که با توجه به انتخاب شدن آنها، صفحه کارکردی خاص خواهد داشت. مثلا در جاوا اسکریپت اگر بخواهیم عنصری را که در یک radioGroup انتخاب شده است را برگردانیم باید کدهای زیر را بنویسیم:

```
var checkedValue;
var elements = document.getElementByTagName ('input');
for (var n = 0; n < elements.length; n++) {
       if (elements[n].type == 'radio' && elements[n].name == 'myRadioGroup' && elements[n].checked) {
              checkedValue = elements[n].value;
       }
}
```

اما اگر بخواهیم همین کد را با جی کوئری بنویسیم:

```
var checkedValue = $ ('[name="myRadioGroup"]:checked').val();
```

ممکن است مثال بالا کمی گنگ باشد نگران نباشید در آینده با این دستورات بیشتر آشنا خواهیم شد.
قدرت اصلی جی کوئری برگفته از انتخاب‌کننده‌ها (Selector) هاست، انتخاب‌کننده ، یک عبارت است که دسترسی به عنصری خاص بر روی صفحه را موجب می‌شود؛ انتخاب‌کننده این امکان را فراهم می‌سازد تا به سادگی عنصر مورد نظر را مشخص و به آن دسترسی پیدا کنیم که در مثال فوق، عنصر مورد نظر ما گزینه انتخاب شده از myRadioGroup بود.
Unobtrusive JavaScript : اگر پیش از پیدایش CSS در کار ایجاد صفحه‌های اینترنتی بوده‌اید حتما مشکلات و مشقات آن دوران را به خاطر می‌آورید. در آن زمان برای فرمت‌دهی به اجزای مختلف صفحه ، به ناچار علائم فرمت‌دهی را به همراه دستورات خود اجزا، در صفحه‌های HTML استفاده می‌کردیم. اکنون بسیار بعید به نظر می‌رسد کسی ترجیح دهد فرمت‌دهی اجزا را به همراه دستورهای HTML آن انجام دهد. اگر چه هنوز دستوری مانند زیر بسیار عادی به نظر می‌آید:

```
<button type="button" onclick="document.getElementById('xyz').style.color='red';">
        Click Me
</button>
```

نکته ای که در مثال فوق حائز اهمیت است، این است که خصوصیات ظاهری دکمه ایجاد شده از قبیل فونت و عنوان دکمه، از طریق تگ <font> و یا پارامترهای قابل استفاده در خود دستور دکمه تعیین نشده است، بلکه CSS وظیفه تعیین آنها را دارد. اما اگرچه در این مثال فرمت‌دهی و دستور خود دکمه از یکدیگر جدا شده‌اند؛ شاهد ترکیب این دکمه با رفتار آن هستیم. در جی کوئری می‌توانیم رفتار را از اجزا به آسانی جدا کنیم.

مجموعه عناصر در جی کوئری :

زمانی که CSS به عنوان یک تکنولوژی به منظور جداسازی طراحی از ساختار به دنیای صفحه‌های اینترنتی معرفی شد، می‌بایست راهی برای اشاره به اجزای صفحات از طرف فایل CSS نیز معرفی می‌شد. این امر از طریق **انتخاب‌کننده‌ها (Selector)** صورت پذیرفت.

برای مثال انتخاب‌کننده زیر، به تمام عناصر <a> اشاره دارد که در یک عنصر <p> قرار گرفته‌اند:

```
p a
```

جی کوئری نیز از چنین انتخاب‌کننده‌هایی استفاده می‌کند، البته نه تنها از انتخاب‌کننده‌هایی که هم اکنون در CSS موجود می‌باشند، بلکه برخی از انتخاب‌کننده‌هایی که هنوز در تمام مرورگرها پشتیبانی نمی‌شوند.
برای انتخاب مجموعه‌ای از عناصر از یکی از دو Syntax زیر استفاده می‌کنیم.

```
$(Selector)
یا
jQuery(Selector)
```

ممکن است در ابتدا ()$ کمی نا معمول به نظر آید، اما اکثر کسانی که با جی کوئری کار می‌کنند از اختصار و کوتاهی این ساختار استفاده می‌کنند.
مثال زیر نمونه‌ای دیگر است که در آن مجموعه‌ای از تمام لینک‌هایی که درون تگ <p> قرار دارند را انتخاب می‌کند:

```
$("p a")
```

تابع ()$ که در حقیقت نام خلاصه‌ای برای ()jQuery می‌باشد، نوع خروجی مخصوصی دارد که شامل یک آرایه از اشیایی می‌شود که انتخاب‌کننده آن را برگزیده است. این نوع خروجی این مزیت را دارد که شمار زیادی متد از پیش تعریف شده را داراست که به سادگی قابل اعمال می‌باشند.
در اصطلاح برنامه نویسی به چنین توابعی که گروهی از عناصر را جمع می‌کنند، Wrapper می‌گویند زیرا تمام عناصر مطلوب را تحت

یک شی بسته‌بندی می‌کند. در جی‌کوئری به آنها Wrapped Set یا jQuery Wrapper می‌گویند و به متدهایی که قابل اعمال بروی اینها به نام jQuery Wrapper Methodes شناخته می‌شوند.
در مثال زیر می‌خواهیم تمام عناصر در صورتی که دارای کلاس notLongForThisWorldباشند را مخفی (با فید شدن) کنیم.

```
$("div.notLongForThisWorld").fadeOut();
```

یکی از مزیت‌های اکثر متدهای قابل اجرا بروی مجموعه عناصر انتخاب شده آن است که خروجی خود آنها مجموعه‌ای دیگر است. به این معنا که خروجی این متد، آماده اعمال یک متد دیگر است.
فرض کنید در مثال بالا بخواهیم پس از مخفی کردن هر بخواهیم یک کلاس به نام removedبه آن بیافزاییم. به این منظور می‌توان کدی مانند زیر نوشت:

```
$("div.notLongForThisWorld").fadeOut().addClass("removed");
```

این زنجیره متدها می‌توانند به هرتعداد ادامه پیدا کند.

چند نمونه انتخاب کننده:

نتیجه		انتخاب کننده
تمام <p>های زوج را انتخاب می‌کند		`$('p:even')`
سطر اول هر جدول را انتخاب می‌کند		`$("tr:nth-child(1)");`
هایی که مستقیما در <body> تعریف شده باشند را انتخاب می‌کند.		`$("body > div");`
لینک هایی که به یک فایل pdf اشاره دارند را انتخاب می‌کند.		`$("a[href$=pdf]");`
تمام هایی که مستقیما در <body> معرفی شده اند و دارای لینک می‌باشند را انتخاب می‌کند.		`$("body > div:has(a)")`

ادامه مطالب در پست‌های بعدی تشریح خواهد شد.

جهت مطالعه بیشتر می‌توانید از این منابع ^ و ^ و ^ و ^ و ^ استفاده کنید.
موفق و موید باشید

نظرات خوانندگان

نویسنده: امیر
تاریخ: ۱۳۹۱/۱۱/۲۳ ۱۱:۲۶

مرسی .اقا خوب ادامه بده

نویسنده: آرآر
تاریخ: ۱۳۹۱/۱۲/۰۱ ۱۳:۳

ممنون. حتما ادامه بدید

عنوان: آموزش (jQuery) جی کوئری #2
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۰۶ ۱۲:۱۰
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, jQuery, Web Design, Learning jQuery

در ادامه مطلب قبلی آموزش (jQuery) جی کوئری #1 به ادامه بحث می‌پردازیم.

توابع سودمند

با وجود آنکه انتخاب کردن و ایجاد مجموعه ای از عناصر صفحه یکی از معمول‌ترین و پراستفاده‌ترین کاربردهای تابع ()$ محسوب می‌شود، این تابع توانایی‌های دیگری نیز دارد. یکی از مفیدترین آنها استفاده شدن به عنوان فضای نام گروهی برای توابع سودمند می‌باشد. تعداد زیادی تابع سودمند با استفاده از $ به عنوان فضای نام قابل دسترسی می‌باشند که اکثر نیازهای یک صفحه را پاسخگو می‌باشند در این پست برخی از آنها را معرفی می‌کنیم در پست‌های آینده سعی می‌کنیم توابع سودمند بیشتری را شرح دهیم.

فراخوانی و استفاده از این توابع در ابتدا ممکن است کمی عجیب به نظر برسد. به مثال زیر دقت کنید که تابع سودمند () trim را فراخوانی کرده ایم.

```
$.trim(someString);
```

در صورتی که نوشتن علامت **$** برای شما عجیب به نظر می‌رسد می‌توانید شناسه دیگر با نام **jQuery** به کار ببرید. کد زیر دقیقا مانند بالا عمل می‌کند شاید درک آن راحت‌تر هم باشد.

```
jQuery.trim(someString);
```

بدیهی است که از **jQuery** یا **$** تنها به عنوان فضای نامی که تابع **trim()** در آن تعریف شده اند، استفاده شده باشد.

نکته : اگر چه در نوشته‌های آنلاین jQuery، این عناصر به عنوان توابع سودمند در معرفی شده اند اما در حقیقت آنها متدهایی برای تابع **()$** می‌باشند.

عملکرد صفحه آماده (The document ready handler)

هنگامی که از Unobtrusive JavaScriptاستفاده می‌کنیم، رفتار از ساختار جدا می‌شود، بنابراین برای انجام عملیات روی عناصر صفحه باید منتظر بمانیم تا انها ایجاد شوند. برای رسیدن به این هدف، ما نیاز به راهی داریم که تا زمان ایجاد عناصر DOM روی صفحه منتظر بماند قبل از آن عملیات را اجرا کند.

به طور معمول از onload برای نمونه‌های window استفاده می‌شود، که پس از لود شدن کامل صفحه ، دستورها قابل اجرا می‌باشند. بنابراین ساختار کلی آن کدی مانند زیر خواهد بود:

```
window.onload = function() {
    $("table tr:nth-child(even)").addClass("even");
};
```

نوشتن کد به صورت بالا سبب می‌شود که کد پس از بارگذاری کامل صفحه اجرا شود. متاسفانه، مرورگرها تا بعد از ساخته شدن عناصر صفحه صبر نمی‌کنند، بلکه پس از ساخت درخت عناصر صفحه منتظر بارگذاری کامل منابع خارجی صفحه مانند تصاویر نیز می‌مانند و سپس آنها را در پنجره مرورگر نمایش می‌دهند. در نتیجه بازدید کننده زمان زیادی منتظر می‌ماند تا رویداد **onload** تکمیل شود.

حتی بدتر از آن، زمانی است که اگر به طور مثال یکی از تصاویر با مشکل مواجه شود که زمان قابل توجهی صرف بارگذاری آن

شود، کاربر باید تمام این مدت را صبر کند تا پس از آن بتواند با این صفحه کار کند. این نکته می‌تواند دلیلی برای استفاده نکردن از Unobtrusive JavaScriptبرای شروع کار باشد.

اما راه بهتری نیز وجود دارد، می‌توانیم تنها زمانی که قسمت ساختار عناصر صفحه ترجمه شده و HTML به درخت عناصر تبدیل می‌شود، صبر کنیم . پس از آن کد مربوط به رفتارها را اجرا کنیم. رسیدن به این روش برای استفاده از Cross-Browser کمی مشکل است، اما به لطف jQuery و قدرت آن، این امر به سادگی امکان پذیر است و دیگر نیازی به منتظر ماندن برای بارگذاری منابع صفحه مانند تصاویر و ویدیوها نمی‌باشد. Syntax زیر نمونه ای از چنین حالتی است:

```
$(document).ready(function() {
   $("table tr:nth-child(even)").addClass("even");
});
```

ابتدا صفحه مورد نظر را به تابع **$()** ارسال کرده ایم، سپس هر زمان که آن صفحه آماده شد (Ready) ، تابع ارسال شده به آن اجرا خواهد شد. البته می‌توان کد نوشته شده بالا را به شکل مختصرتری هم نوشت:

```
$(function() {
    $("table tr:nth-child(even)").addClass("even");
});
```

با ارسال تابع به **$()**، ما مرورگر را مجبور می‌کنیم که برای اجرای کد تا زمانی که DOM کامل لود شود (فقط DOM لود شود) منتظر بماند. حتی بهتر از آن ما می‌توانیم از این تکنیک چندین با در همان سند HTML استفاده کرده و مرورگر تمامی تابع‌های مشخص شده توسط ما را به ترتیب اجرا خواهد کرد. (یعنی من در دیک صفحه می‌توانم چنین بار تابع **ready()** را فراخوانی کنم). در مقابل روش **OnLoad** پنجره فقط اجازه اجرای یکبار تابع را به ما می‌دهد.

این هم یکی دیگر از کارکردهای دیگر تابع **$()** می‌باشد. حال به یکی دیگر از امکاناتی که این تابع برای ما فراهم می‌کند دقت کنید.

ساختن اجزای DOM (ساختن عناصر صفحه)

یکی دیگر از کارهایی که تابع **$()** می تواند برای ما انجام دهد ایجاد کردن عناصر صفحه است. به این منظور ورودی تابع **$()** را یک رشته که حاوی دستور HTML مربوط به ساخت یک عنصر می‌باشد، قرار می‌دهیم. برای مثال دستور زیر یک تگ p ایجاد می‌کند:

```
$("<p>Hi there!</p>")
```

اما ایجاد یک عنصر DOM یا (سلسله مراتب عناصر DOM) برای ما به تنهایی سودمند نیست، و هدف ما چیز دیگری است. ایجاد اشیا صفحه توسط **$()** زمانی برای ما مفید خواهد بود که بخواهیم به هنگام ساخت، تابعی بروی آن اعمال کنیم یا به محض ساخت آن را به تابعی ارسال کنیم به کد زیر دقت کنید:

```
<html>
   <head>
        <title>Follow me!</title>
        <script type="text/javascript" src="../scripts/jquery-1.2.js"></script>
        <script type="text/javascript">
           // بودن صفحه عنصر مورد نظر ایجاد می‌شود Reday در زمان
           $(function(){
              $("<p>Hi there!</p>").insertAfter("#followMe");
           });
        </script>
    </head>
<body>
     <p id="followMe">Follow me!</p>
</body>
</html>
```

در کد بالا زمانی که صفحه مورد نظر Ready شد تابع مورد نظر ما اجرا شده و در عناصر صفحه بعد از عنصری که id آن followMe می‌باشد یک عنصر p را ایجاد می‌کند. که خروجی آن شبیه تصویر زیر خواهد بود.

مزیت دیگر jQuery این است که در صورتی که امکانی را ندارد شما به آسانی می‌توانید آن را توسعه داده و برای آن پلاگین طراحی کنید.

برای پایان دادن به این پست همانطور که دیدیم jQuery قادر به انجام کارهای زیر است:
انتخاب عناصر و ایجاد مجموعه ای از آنها که آماده اعمال متدهای مختلف می‌باشند.
استفاده به عنوان یک فضای نام برای توابع سودمند.
ایجاد اشیا مختلف HTML بروی صفحه.
اجرای کد به محض آماده شدن اشیای صفحه.

موفق وموید باشید

عنوان: JQuery Plugins #1
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۰۷ ۱۶:۲۵
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, jQuery-Tips, Plugin

جی‌کوئری به عنوان مهم‌ترین و پرکاربردترین کتابخانه جاوا اسکریپتی، حالا در اکثر سایت‌های اینترنتی استفاده می‌شود و هر روز به قابلیت‌ها و امکانات آن اضافه می‌گردد. اما بیش از خود این کتابخانه، پلاگین‌های آن است که تحول عظیمی را در طراحی وب سایت‌ها ایجاد نموده است. از انواع اسلایدها، تصاویر، منوها، Tooltip ها، نمودارها، انیمیشن، جداول و هزاران پلاگین دیگر، همه و همه کدهای جاوا اسکریپتی است که با استفاده از جی کوئری به صورت پلاگین نوشته شده است و امکان استفاده مجدد را به ما می‌دهد.

از کجا شروع کنیم

برای نوشتن پلاگین یک تابع با نام خاصیتی جدید را به jQuery.fn اضافه می‌نماییم.

```
jQuery.fn.myPlugin = function() {
//محتویات پلاگین را اینجا می‌نویسیم
};
```

اما، برای اینکه بتوانیم از میانبر $ در پلاگین استفاده نماییم و تداخلی با سایر کتابخانه‌ها نداشته باشد، از الگوی (IIFE Immediately Invoked Function Expression D) به صورت زیر استفاده می‌نماییم:

```
(function( $ ) {
  $.fn.myPlugin = function() {
  //محتویات پلاگین را اینجا می‌نویسیم
    };
})( jQuery );
```

محتوای پلاگین

حال می‌توانیم در تابع، کدهای پلاگین خود را بنویسیم. برای دسترسی به شیء پاس داده شده به پلاگین، از کلمه کلیدی this استفاده کرده و لازم نیست از $(this) استفاده نماییم. در زیر یک پلاگین ساده تهیه شده است که با رفتن ماوس بر روی یک متن، خطی زیر آن می‌کشد:

```
(function($){
    $.fn.underline= function() {
        this.hover(function(){
            $(this).css( { text-decoration : underline })
        }, function(){
            $(this).css( { text-decoration : none } )
        });
    };
})(jQuery);

$("p").underline();
```

پلاگین بالا مقدار یا شیء ایی را بر نمی‌گرداند؛ اما اگر بخواهیم مقداری را برگردانیم از return استفاده می‌نماییم:

```
(function( $ ){

  $.fn.maxHeight = function() {

    var max = 0;

    this.each(function() {
      max = Math.max( max, $(this).height() );
    });

    return max;
```

```
  };
})( jQuery );
```

```
var tallest = $('div').maxHeight(); // بیشترین ارتفاع عنصر را برمی گرداند
```

حفظ خاصیت زنجیره‌ای پلاگین ها

در مثال بالا یک مقدار عددی برگردانده شده است؛ اما برای اینکه بتوانیم بصورت زنجیر وار خروجی پلاگین را به تابع یا هر پالاگین دیگری پاس دهیم از تابع each بصورت زیر استفاده می‌نماییم:

```
(function( $ ){

  $.fn.lockDimensions = function( type ) {

    return this.each(function() {

      var $this = $(this);

      if ( !type || type == 'width' ) {
        $this.width( $this.width() );
      }

      if ( !type || type == 'height' ) {
        $this.height( $this.height() );
      }

    });

  };
})( jQuery );
```

```
$('div').lockDimensions('width').css('color', 'red');
```

در پلاگین بالا با از تابع each برای روی this و برگرداندن آن با return برای حفظ خاصیت زنجیره‌ای پلاگین استفاده می‌نماییم. در تابع each می‌بایست از $(this) برای انجام عملیات بر روی شیء پاس داده شده استفاده کنیم. بدین صورت بعد از صدا زدن پلاگین، دوباره می‌توانیم از هر پلاگین یا تابع جی کوئری دیگری بر روی خروجی استفاده نماییم.

پیش فرض‌ها و تنظیمات

در پلاگین‌های پیشرفته‌تر می‌توانیم تنظیمات پیش فرضی را برای پلاگین در نظر بگیریم و این تنظیمات را به عنوان پارامتر ورودی از کاربر دریافت نماییم. جی کوئری دارای تابعی به نام extend است که امکان گسترش و ترکیب دو شیء را امکان پذیر می‌سازد به مثال زیر توجه نمایید:

```
(function( $ ){

  $.fn.tooltip = function( options ) {

    var settings = $.extend( {
      'location'         : 'top',
      'background-color' : 'blue'
    }, options);

    return this.each(function() {

      // Tooltip plugin code here

    });

  };
})( jQuery );
```

```
$('div').tooltip({
  'location' : 'left'
});
```

در این مثال، شیء settings با دو خاصیت location و background-color تعریف شده که با شیء options که از ورودی پلاگین دریافت نموده‌ایم با استفاده از تابع extend ترکیب شده است. خاصیت‌های که تعیین نشده باشند با مقادیر پیش فرض آن‌ها تکمیل می‌گردد.

ادامه دارد...

نظرات خوانندگان

نویسنده: امیر
تاریخ: ۱۳۹۱/۱۲/۰۹ ۱۳:۵۴

مرسی عالی بود .استفاده کردم.فقط زمانی که سی اس اس میدی نباید اونطوری نوشته بشه

```
$(this).css( "text-decoration" ,"none"  )
```

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۰۹ ۱۶:۲۳

ممنون. در جاواسکریپ هر دو صحیح است

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۲/۰۹ ۱۶:۵۳

```
$(this).css( { text-decoration : underline })
```

فکر کنم منظور ایشون («اونطوری» که نوشته) وجود dash در نام متغیر بوده.

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۰۹ ۱۸:۲۸

منظورشون نحوه ست کردن css که هر دو روش استفاده می‌شود ولی در jquery 1.9 css به بعد کمی تغییر کرده است

عنوان: **آموزش (jQuery) جی کوئری #3**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۰۸ ۱۲:۳۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Web Design, Learning jQuery

در ادامه مطلب قبلی آموزش (jQuery) جی کوئری #2 به ادامه بحث می‌پردازیم.

انتخاب عناصر صفحه

در پستهای قبل (^ و ^) با بسیاری از توانایی‌ها و کارکردهای jQuery شامل توانایی‌های آن برای انتخاب عناصر موجود در صفحه تا تعریف توابع جدید و استفاده از آنها به محض آماده شدن صفحه آشنا شدیم.

در این پست و پست بعدی توضیحات تکمیلی در خصوص دو مورد از توانایی‌های jQuery و البته تابع ()$ خواهیم داشت که مورد اول، انتخاب عناصر صفحه با استفاده از انتخاب کننده‌ها و مورد دوم ایجاد عناصر جدید می‌باشد.

در بسیاری از مواقع برای تعامل با صفحه اینترنتی نیاز به تغییر دادن بخشی از یکی از اشیا موجود در صفحه داریم. اما پیش از آنکه قادر باشیم آنها را تغییر دهیم، ابتدا باید با استفاده از مکانیزمی شی مورد نظر را مشخص و سپس آن را انتخاب کنیم تا پس از آن قادر به اعمال تغییری در آن باشیم. بنابراین اجازه دهید تا به یک بررسی عمیق از راه‌های مختلف انتخاب عناصر صفحه و ایجاد تغییر در آنها بپردازیم.

-1انتخاب عناصر صفحه برای ایجاد تغییر

اولین قدم برای استفاده از هر گونه تابع jQuery، مشخص کردن و انتخاب عناصری است که می‌خواهیم تابع روی آن عناصر اعمال شود. گاهی اوقات انتخاب این مجموعه عناصر با یک توضیح ساده مشخص می‌شود، برای مثال "تمام عناصر پاراگراف موجود در صفحه". اما گاهی اوقات مشخص کردن این مجموعه نیاز به توضیح پیچیده‌تری دارد، برای مثال "تمام عناصر لیست در صفحه که دارای کلاس listElement هستند و لینکی دارند که اولین عضو آن لیست می‌باشد".
خوشبختانه jQuery یک مکانیزم بسیار قوی و قدرتمند ارایه کرده است که انتخاب هر عنصری از صفحه را به سادگی امکان پذیر می‌سازد. انتخاب کننده‌های jQuery از ساختار مربوط به CSS استفاده می‌کنند، بنابراین ممکن است شما هم اکنون با تعداد زیادی از آنها آشنا باشید . در ادامه شمار بیشتر و قدرتمندتری خواهید آموخت.
برای درک بهتر شما از مطالب مربوط به بخش انتخاب کننده ها، یک مثال آماده مختص به این مبحث، در قالب یک صفحه اینترنتی، را در فایل صفحه کارگاهی قرار داده ایم، این فایل در ادرس chapter2/lab.selector.htm قابل دسترسی می‌باشد. این مثال از پیش آماده و کامل (نوشته شده توسط نویسنده کتاب)، این امکان را به شما می‌دهد تا با وارد کردن یک رشته، به عنوان پارامتر انتخاب کننده، در همان زمان عنصر انتخاب کننده در صفحه را رویت کنید. زمانی که این صفحه را اجرا می‌کنید تصویری مانند زیر ظاهر خواهد شد.

برای درک بهتر مطالب این سلسله پست‌ها می‌توانید فایل‌های کتاب را از آدرس اصلی آن یا از این آدرس در همین سایت دانلود نمایید.

این صفحه سه پنجره مجزا دارد. در پنجره سمت چپ ، یک textBox و یک دکمه دیده می‌شود، که با وارد کردن یک انتخاب کننده در textBox و فشردن دکمه، عنصر مورد نظر در پنجره سمت راست انتخاب می‌شود. برای شروع در textBox عبارت li را بنویسید و دکمه Apply را کلیک کنید.

با انجام این عمل تصویر زیر باید خروجی شما باشد. می‌توانید حالت‌های دیگر را خودتان امتحان کنید.

Figure 2.2 A selector value of `li` matches all `<li>` elements when applied as shown by the display results.

1-1- استفاده از انتخاب کننده‌های ابتدایی CSS

برنامه نویسان وب برای اعمال فرمت‌های ظاهری گوناگون به بخش‌ها و عناصر مختلف یک صفحه اینترنتی، از ایک راه بسیار ساده، در عین حال قدرتمند و کارا استفاده می‌کنند که در تمام مرورگرهای مختلف نیز جوابگو باشد. این انتخاب کننده‌ها عناصر را بر اساس نام شناسه آنها، نام کلاس و یا ساختار سلسله مراتبی موجود در صفحه انتخاب می‌کنند.

در زیر به معرفی چند نمونه از این انتخاب کننده‌های ساده CSS می‌پردازیم:

a : تمام عناصر <a> را انتخاب می‌کند.

#specialID : عنصری را که دارای ID با عنوان specialID باشد انتخاب می‌کند.

.specialClass : عناصری را که دارای کلاس specialClass هستند انتخاب می‌کند.

a#specialID.specialClass : این عبارت عنصری را انخاب می‌کند که شناسه آن specialID باشد، به شرط آنکه این عنصر <a> باشد و دارای کلاس specialClass نیز باشد را انتخاب می‌کند.

p a.specialClass: تمام عناصر لینک (<a>) را که دارای کلاس specialClass باشند و درون یک عنصر پاراگراف (<p>) قرار گرفته باشند را انتخاب می‌کند.

این انتخاب کننده‌ها شاید ساده به نظر برسند، اما در بسیاری از مواقع پاسخگوی ما می‌باشند؛ به علاوه آنه که با ادغام این انتخاب کننده‌های ساده، ما می‌توانیم انتخاب کننده‌های پیچیده‌تر و تخصصی‌تر ایجاد کنیم.

نکته مثبت در مورد انتخاب کننده‌های CSS این است که از همین انتخاب کننده‌ها می‌توانیم در jQuery نیز استفاده کنیم. برای این کار تنها کافیست انتخاب کننده مورد نظر را به تابع ()$ ارسال کنیم. در زیر یک نمونه را مشاهده می‌کنید:

```
$("p a.specialClass")
```

به جز چند مورد خاص که استثنا وجود دارد، CSS3 و jQuery کاملا با هم سازگاری دارند. بنابراین انتخاب عناصر به این شکل طبیعی خواهد بود. به عبارتی دیگر هر عنصر که از این طریق توسط CSS انتخاب شود، همان انتخاب حاصل انتخاب کننده jQuery نیز خواهد بود. اما باید به این نکته توجه داشت که jQuery وابسته به CSS نیست و اگر مرورگری پیاده سازی استانداردی برای CSS نداشته باشد، انتخاب کننده jQuery به مشکل بر نمی‌خورد، بلکه jQuery انتخاب خود را به درستی انجام می‌دهد، چرا که jQuery از قوانین استاندارد W3C تبعیت می‌کند.

2-1- استفاده از انتخاب کننده‌های فرزند (Child) ، نگهدارنده (Container) و صفت (Attribute)

برای انتخاب کننده‌های پیشرفته تر، jQuery از جدیدترین مرورگرهایی که CSS را پشتیبانی می‌کنند، استفاده می‌کند که می‌توان به Mozilla Firefox, Internet Explorer 7, Safariو سایر مرورگرهای پیشرفته (مدرن) اشاره کرد. این انتخاب کننده‌های پیشرفته شما را قادر می‌سازند تا مستقیما فرزند یک عنصر را انتخاب کنید و یا از ساختار سلسله مراتبی عناصر صفحه، مستقیما به عنصر مورد نظر دسترسی داشته باشید و یا حتی تمام عناصری که یک صفت خاص را شامل می‌شوند، انتخاب کنید. گاهی اوقات انتخاب فرزندی از یک شی برای ما مطلوب است. برای مثال ممکن است ما به چند مورد از یک لیست احتیاج داشته باشیم، نه یک زیر مجموعه ای از آن لیست. به قطعه کد زیر که از صفحه کارگاهی این پست گرفته شده است دقت نمایید:

```
<ul>
   <li><a href="http://jquery.com">jQuery supports</a>
      <ul>
            <li><a href="css1">CSS1</a></li>
            <li><a href="css2">CSS2</a></li>
            <li><a href="css3">CSS3</a></li>
            <li>Basic XPath</li>
       </ul>
    </li>
    <li>jQuery also supports
        <ul>
             <li>Custom selectors</li>
             <li>Form selectors</li>
         </ul>
      </li>
</ul>
```

حال فرض کنید از این ساختار، لینک وب سایت jQuery مد نظر ماست و این کار بدون انتخاب سایر لینک‌های مربوط به CSS مطلوب است. اگر بخواهیم از دستورهای انتخاب کننده CSS استفاده کینم، دستوری به شکل ul.myList li a خواهیم داشت. اما متاسفانه این دستور تمام لینک‌های این ساختار را انتخاب میکند، زیرا همه آنها لینک هایی در عنصر li می‌باشند. با نوشتن این دستور در صفحه کارگاهی خروجی به شکل زیر خواهد بود:

Figure 2.3 All anchor tags that are descendents, at any depth, of an <li> element are selected
ul.myList li a

راه حل مناسب برای انتخاب چنین حالتی استفاده از **انتخاب فرزند** می باشد که به این منظور Parent (والد) و Child (فرزند)، به وسیله یک کاراکتر > از یکدیگر جدا می‌شوند:

```
p > a
```

این دستور تنها لینک (<a>) هایی را بر می‌گرداند که فرزند مستقیم یک عنصر <p> می‌باشند. بنابراین اگر در یک <p> لینکی در عنصر معرفی شده باشد، این لینک انتخاب نمی‌شود، چرا که فرزند مستقیم <p> به حساب نمی‌آید. در مورد مثال لینک‌های موجود در لیست، می‌توانیم دستور زیر را به منظور انتخاب لینک مورد نظرمان استفاده کنیم:

```
ul.myList > li > a
```

دستور انتخاب فوق از میان عناصر <ul>، عنصری را که دارای کلاس myList می‌باشد، انتخاب می‌کند و پس از آن لینکهایی (<a>) که فرزند مستقیم گزینه‌های آن هستند، برگردانده می‌شوند. همانگونه که در شکل زیر مشاهده می‌کنید لینک‌های زیرمجموعه عنصر <ul> انتخاب نمی‌شوند، زیرا فرزند مستقیم این عنصر محصوب نمی‌شوند.

Figure 2.4 With the selector `ul.myList > li > a`, only the direct children of parent nodes are matched.

انتخاب کننده‌های صفت نیز بسیار قدرتمند می‌باشند و ما را تواناتر می‌سازند، فرض کنید برای منظوری خاص قصد دارید به تمام لینک‌های موجود در صفحه که به مکانی خارج از این وب سایت اشاره دارند، رفتاری را اضافه کنید (مثلا مانند همین سایت به کنار آنها یک آیکن اضافه نمایید) . فرض کنید این کد (کد موجود در مثال کارگاهی) را در صفحه خود دارید:

```
<li><a href="http://jquery.com">jQuery supports</a>
    <ul>
          <li><a href="css1">CSS1</a></li>
          <li><a href="css2">CSS2</a></li>
          <li><a href="css3">CSS3</a></li>
          <li>Basic XPath</li>
     </ul>
</li>
```

موردی که یک لینک با اشاره به وب سایت خارجی را از سایر لینک‌ها متمایز می‌سازد، شروع شدن مقدار صفت href آن با http:// می‌باشد. انتخاب لینک هایی که مقدار href آنها با http:// آغاز می‌شود، به سهولت و از طریق دستور زیر صورت می‌پذیرد:

```
a[href^=http://]
```

این دستور باعث انتخاب تمام لینک هایی که مقدار صفت href آنها **دقیقا** با http:// آغاز می‌شود، می‌گردد. علامت ^ موجب می‌شود تابررسی، لزوما از ابتدای مقادیر صورت پذیرد و از آنجا که استفاده از این کاراکتر در سایر عبارات منظم به همین منظور صورت می‌پذیرد، به خاطر سپردن آن دشوار نخواهد بود.

می توانید این کد را در صفحه کار گاهی تست کنید.

راهای دیگری برای استفاده از انتخاب کننده‌های صفت وجود دارد.

```
form[method]
```

این دستور تمام عناصر <form> را که یک صفت method دارند را انتخاب می‌کند.

```
input[type=text]
```

این انتخاب کننده تمام عناصر input را که type آنها برابر text با شد انتخاب می‌کند.
دستور زیر مثالی دیگر برای بررسی یک مقدار بر اساس کاراکترهای نخست آن می‌باشد:

```
div[title^=my]
```

همانطور که از دستور فوق بر میآید، عناصر div که مقدار title آنها با رشته my اغاز می‌شود، هدف این انتخاب کننده خواهد بود. اما اگر بخواهیم تنها بر اساس کاراکترهای انتهایی انتخابی انجام دهیم، دستور مناسب چه خواهد بود؟ برای چنین منظوری مانند زیر عمل می‌کنیم:

```
a[href$=.pdf]
```

این دستور کاربرد زیادی برای شناسایی لنک‌های اشاره کننده به فایل‌های pdf دارد. ساختار زیر نیز زمانی استفاده می‌شود که یک عبارت منظم در جایی از یک صفت قرار گرفته باشد، خواه این عبارت از کاراکتر دوم آغاز شده باشد و یا از هرجای دیگر.

```
a[href*=jquery.com]
```

همانگونه که انتظار میرود این انتخاب کننده ، تمام لینک هایی که به وب سایت jQuery اشاره دارند را برمی گرداند.
فراتر از خصوصیات، بعضی مواقع ما می‌خواهیم بررسی کنیم که آیا یک عنصر شامل عنصر دیگری هست یا خیر. در مثال‌های قبلی فرض کنید ما می‌خواهیم بدانیم که آیا یک li شامل a هست یا خیر، jQuery با **استفاده از انتخاب کننده‌های Containerها** این را پشتیبانی می‌کند:

```
li:has(a)
```

این انتخاب کننده همه li هایی را برمی گرداند که شامل لینک (<a>) هستند. دقت کنید که این انتخاب گر مانند li a نیست، انتخاب گر دوم تمامی لینک هایی را که در li هستند بر میگرداند اما دستور بالا li هایی را بر میگرداند که دارای لینک (<a>) هستند.

تصویر زیر انتخاب گرهایی را نشان میدهد که ما می‌توانیم در jQuery استفاده نماییم.

Table 2.1 The basic CSS Selectors supported by jQuery

Selector	Description
*	Matches any element.
E	Matches all element with tag name E.
E F	Matches all elements with tag name F that are descendents of E.
E>F	Matches all elements with tag name F that are direct children of E.
E+F	Matches all elements F immediately preceded by sibling E.
E-F	Matches all elements F preceded by any sibling E.
E:has(F)	Matches all elements with tag name E that have at least one descendent with tag name F
E.C	Matches all elements E with class name C. Omitting E is the same as *.C.
E#I	Matches element E with id of I. Omitting E is the same as *#I.
E[A]	Matches all elements E with attribute A of any value.
E[A=V]	Matches all elements E with attribute A whose value is exactly V.
E[A^=V]	Matches all elements E with attribute A whose value begins with V.
E[A$=V]	Matches all elements E with attribute A whose value ends with V.
E[A*=V]	Matches all elements E with attribute A whose value contains V.

انشالله در پست‌های بعدی ادامه مباحث را بررسی خواهد شد.

نظرات خوانندگان

نویسنده: هادی
تاریخ: ۱۳۹۳/۰۸/۲۶ ۱۱:۳۵

با سلام

من می‌خوام value انتخاب شده یک تگ Select رو بخونم، این کار رو با دستورات زیر در مرور گر IE جواب گرفتم:

```
$("#ddlPriortiy option:selected").val()
or
 $("#ddlPriortiy").val()
```

ولی در مرورگر فایر فاکس به من مقدار undefined رو بر میگردونه!

مشکل کجاست؟

اینم بگم این مشکل وقتی رخ میده که من این کنترل رو بصورت داینامیک درست میکنم و در صفحه میرزم.

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۸/۲۶ ۱۳:۳۵

باید با jQuery live آشنا باشید. البته اسمش جدیدا شده on و live حذف شده، اما مفهومش یکی هست.

عنوان: JQuery Plugins #2
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۲۶ ۲۳:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, jQuery-Tips, Plugin

در قسمت اول آموزش jQuery Plugin #1 با نحوه ساخت اولیه پلاگین در جی کوئری آشنا شدید. در ادامه به موارد دیگری خواهم پرداخت.

فضای نام

در پلاگین شما، فضای نام، بخش مهمی از توسعه پلاگین می‌باشد. فضای نام در واقع تضمین می‌کند که پلاگین شما توسط دیگر پلاگین‌ها باز نویسی نشود یا با کدهای موجود در صفحه تداخل نداشته باشد. همچنین کمک می‌کند که توابع، رویدادها و داده‌های پلاگین خود را بهتر مدیر کنید.

توابع پلاگین

تحت هیچ شرایطی نباید یک پلاگین، در چندین فضای نام، به شی Jquery.fn اضافه گردند. به مثال زیر توجه نمایید:

```
(function( $ ){

  $.fn.tooltip = function( options ) {
    // این
  };
  $.fn.tooltipShow = function( ) {
    // تعریف
  };
  $.fn.tooltipHide = function( ) {
    // بد است
  };
  $.fn.tooltipUpdate = function( content ) {
    // !
  };

})( jQuery );
```

همین طور که در مثال بالا مشاهده می‌کنید، پلاگین به شکل بدی تعریف شده و هر تابع در یک فضای نام جداگانه تعریف گردیده‌است. برای این کار شما باید تمام توابع را در یک متغیر تعریف و آن را به پلاگین خود پاس دهید و توابع را با نام رشته ای صدا بزنید.

```
(function( $ ){

  var methods = {
    init : function( options ) {
      // این
    },
    show : function( ) {
      // تعریف
    },
    hide : function( ) {
      // خوب است
    },
    update : function( content ) {
      // !
    }
  };

  $.fn.tooltip = function( method ) {

    // منطق تابع را  از اینجا صدا زده ایم
    if ( methods[method] ) {
      return methods[ method ].apply( this, Array.prototype.slice.call( arguments, 1 ));
    } else if ( typeof method === 'object' || ! method ) {
      return methods.init.apply( this, arguments );
    } else {
      $.error( 'Method ' +  method + ' does not exist on jQuery.tooltip' );
    }
```

```
  };

})( jQuery );
```

توضیح : متغیر method اگر در متغیر methods توابع موجود باشد، تابع هم نام آن و در صورت داشتن پارامتر ورودی، به آن تابع پاس داده شده و برگردانده می‌شود (در واقع صدا زده می‌شود). در غیر اینصورت اگر نوع مقدار ورودی، object بود تابع init آن صدا زده می‌شود وگرنه پیام خطا ارسال می‌گردد.

برای استفاده از پلاگین بصورت زیر عمل می‌کنیم:

```
// تابع init صدا زده می‌شود
$('div').tooltip();
```

و

```
// تابع update با پارامتر صدا زده می‌شود
$('div').tooltip('update', 'This is the new tooltip content!');
```

این معماری به شما امکان کپسوله کردن توابع در پلاگین را می‌دهد.

رویداد ها

یکی از روش‌های کمتر شناخته شده انقیاد توابع در فضای نام، امکان انقیاد رویدادها است. اگر پلاگین شما یک رویداد را انقیاد نماید، این یک عمل و تمرین خوب استفاده از فضای نام می‌باشد. بدین ترتیب اگر لازم باشد که انقیاد یک رویدا را حذف نمایید، بدون تداخل با دیگر رویدادها و بدون اینکه در یک شی دیگر از این پلاگین، اختلالی ایجاد نماید می‌توان آن را حذف نمود. به مثال زیر توجه نمایید.

```
(function( $ ){

  var methods = {
     init : function( options ) {

       return this.each(function(){
         $(window).bind('resize.tooltip', methods.reposition);
       });

     },
     destroy : function( ) {

       return this.each(function){
         $(window).unbind('.tooltip');
       })

     },
     reposition : function( ) {
       // ...
     },
     show : function( ) {
       // ...
     },
     hide : function( ) {
       // ...
     },
     update : function( content ) {
       // ...
     }
  };

  $.fn.tooltip = function( method ) {

    if ( methods[method] ) {
      return methods[method].apply( this, Array.prototype.slice.call( arguments, 1 ));
    } else if ( typeof method === 'object' || ! method ) {
      return methods.init.apply( this, arguments );
    } else {
```

```
      $.error( 'Method ' +  method + ' does not exist on jQuery.tooltip' );
    }

  };

})( jQuery );
```

این همان مثال قبل است که وقتی پلاگین با تابع Init مقدار دهی اولیه می‌شود، تابع reposition به رویداد resize پنجره در فضای نام پلاگین tooltip انقیاد می‌شود. پس از آن اگر توسعه دهنده نیاز داشت تا tooltip را از بین ببرد، با صدا زدن تابع destroy می تواند بصورت امن انقیاد ایجاد شده را حذف نماید.

```
$('#fun').tooltip();
// مدتی بعد...
$('#fun').tooltip('destroy');
```

ادامه دارد...

عنوان: آموزش (jQuery) جی کوئری #4
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۲۷ ۱۱:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, jQuery, Web Design, Learning jQuery

در ادامه مطلب قبلی آموزش (jQuery) جی کوئری #3 به ادامه بحث می‌پردازیم.

با توجه به حالت‌های مختلف و گزینه‌های گوناگونی که انتخاب کننده‌ها در اختیار ما گذاشته اند، اگر هنوز دنبال قدرت بیشتری از انتخاب کننده‌ها هستید در ادامه به چند مورد از آنها اشاره خواهیم کرد.

3-1- انتخاب عناصر بر اساس موقعیت

گاهی اوقات انتخاب عناصر با توجه به مکان آنها و یا موقعیت مکانی آن‌ها نسبت به سایر اجزا صورت می‌پذیرد؛ برای مثال اولین لینک صفحه و یا اولین لینک هر پاراگراف و یا گزینه‌ی آخر از لیست، jQuery شیوه ای خاص را برای چنین انتخاب هایی ارایه کرده است. برای مثال دستور زیر اولین لینک موجود در صفحه را انتخاب می‌کند:

```
a:first
```

دستور زیر چکاری انجام می‌دهد؟

```
p:odd
```

دستور بالا تمامی پاراگراف‌های فرد را انتخاب می‌کند. روش‌های دیگری هم ممکن است بخواهیم استفاده کنیم؛ مثلا دستور زیر تمامی پاراگراف‌های زوج را انتخاب می‌کند:

```
p:even
```

یا با استفاده از دستور زیر میتوان آخرین فرزند یک والد را انتخاب کرد؛ در زیر آخرین <lii> فرزند یک <ul> انتخاب می‌شود. علاوه بر انتخاب کننده هایی که ذکر شد؛ تعداد قابل توجه دیگری نیز وجود دارند که در جدول 2-2 ذکر شده اند.

جدول 2-2: انتخاب گرهای پیشرفته موقعیت عناصر که توسط jQuery پشتیبانی می‌شوند

توضیح	فیلتر
اولین عنصر که با شرط ما مطابقت می‌کند را انتخاب می‌کند، li a:first اولین لینکی را که فرزند لیست به حساب می‌آیند؛ را بر می‌گرداند	:first
آخرین عنصری که با شرط ما مطابقت کند را انتخاب می‌کند. li a:last آخرین لینک از فرزندان لیست را برمی گرداند.	:last
اولین فرزند عنصر که با شرط ما مطابقت می‌کند را انتخاب می‌کند. li a:first-child اولین عنصر لینک از هر لیست را برمی گرداند.	:first-child
آخرین فرزند عنصر که با شرط ما مطابقت می‌کند را انتخاب می‌کند. li a:last-child اولین عنصر لینک از هر لیست را	:last-child

توضیح	فیلتر
برمی گرداند.	
تمام عناصری که پدر انها تنها همان فرزند را داد، برمی گرداند.	:only-child
nامین فرزند عنصری که با شرط ما مطابقت داشته باشد را انتخاب می‌کند. //comment li: nth-child:(2) دومین عنصر از هر لیست را برمی گرداند.	:nth-child(n)
فرزندان زوج یا فرد عنصر را انخاب می‌کند. li:nth-child(even) //comment تمام عناصر زوج لیست‌ها را بر می‌گرداند.	:nth-child(even یا odd)
nامین فرزند عنصری که از طریق فرمول ارایه شده به دست می‌آید را انتخاب می‌کند. اگر Y صفر باشد، نیازی به نوشتن آن نیست. //comment li:nth-child(3n) تمام عناصر ضریب 3 لیست‌ها را بر می‌گرداند، در حالی که // li:nth-child(5n+1) comment عناصری از لیست را برمیگرداند که بعد از عنصرهای ضریب 5 لیست‌ها قرار گرفته باشند.	:nth-child(Xn+Y)
تمام عناصر زوج یا فرد که با شرط ما مطابقت کنند را انتخاب می‌کند. li:even تمامی عناصر زوج لیست‌ها را بر می‌گرداند.	:even یا :odd
n امین عنصر انتخاب شده را برمی گرداند.	:eq(n)
عناصر بعد از n امین عنصر را بر میگرداند. (در واقع عناصری که بزرگتر از عنصر n ام هستند را بر می‌گرداند)	:gt(n)
عناصر قبل از n امین عنصر را بر میگرداند. (در واقع عناصری که کوچکتر از عنصر n ام هستند را بر می‌گرداند)	:lt(n)

پ.ن : در جدول بالا در توضیحات بعضی از انتخاب گرها //comment نوشه شده است، اینها جز دستور نبوده و فقط برای نمایش صحیح پردانتز در صفحه اینترنتی نوشته شده است، در عمل نیازی به اینها نیست.

نکته ای که در مورد انتخاب گرهای جدول بالا وجود دارد این است که در فیلتر nth-child: برای سازگاری با CSS، مقدار شمارشگر از 1 آغاز می‌شود، اما در سایر فیلترها از قاعده ای که اکثر زبانهای برنامه نویسی استفاده شده است و شمارشگر آنها از صفر اغاز می‌شود. برای درک این موضوع مثال زیر را در نظر بگیرید:

```
<table id="languages">
     <thead>
          <tr>
               <th>Language</th>
               <th>Type</th>
               <th>Invented</th>
          </tr>
     </thead>
<tbody>
<tr>
     <td>Java</td>
     <td>Static</td>
     <td>1995</td>
</tr>
<tr>
     <td>Ruby</td>
     <td>Dynamic</td>
     <td>1993</td>
</tr>
<tr>
     <td>Smalltalk</td>
```

```
     <td>Dynamic</td>
     <td>1972</td>
</tr>
<tr>
     <td>C++</td>
     <td>Static</td>
     <td>1983</td>
</tr>
</tbody>
</table>
```

حال می‌خواهیم از این جدول، محتویات تمام خانه هایی که نام یک زبان برنامه نویسی در آنهاست را انتخاب نماییم. از انجا که نام این زبانها در اولین ستون از هر سطر قرار دارد. می‌توانیم دستوری مانند زیر بنویسیم:

```
table#languages tbody td:first-child
```

و یا با استفاده از دستور زیر این کار را انجام دهیم:

```
table#languages tbody td:nth-child(1)
```

اما دستور اول مختصرتر و خواناتر است. پس از آن برای دسترسی به نوع هریک از زبانهای برنامه نویسی، دستور انتخاب کننده دوم را به صورت

```
:nth-child(2)
```

تغییر می‌دهیم، و همچنین با تغییر پارامتر 2 به 3 سالی که هر یک از زبانها ابداع شده اند ، انتخاب می‌شوند. بدیهی است در این حالت دو دستور

```
:nth-child(3)
یا
:last-child
```

با یکدیگر برابرند. اما هردوی آنها ستون آخر از هر سطر را انخاب می‌کنند، در شرایطی که بخواهیم آخرین خانه جدول انتخاب شود (خانه ای با مقدار 1983)، از **td:last** استفاده می‌کنیم. توجه کنید در حالی که دستور comment // **td:eq(2)** خانه ای با مقدار 1995 را انتخاب می‌کند، دستور comment // **td:nth-child(2)** تمام خانه‌های بیان کننده نوع زبانها را انتخاب می‌کند. بنابراین به خاطر داشته باشید که مقدار ابتدایی شمارشگر فیلتر **:eq** از صفر است و این مقدار برای فیلتر **:nth-child** یک تعیین شده است.

در پست بعدی انتخاب گرهای CSS و فیلترهای سفارشی jQuery را بررسی خواهیم کرد.

عنوان: **پلاگین DataTables کتابخانه jQuery - قسمت اول**
نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۲:۰
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, DataTables

DataTables پلاگینی برای کتابخانه jQuery است. این پلاگین امکانات پیشرفته ای برای یک جدول html که حاوی داده‌ها است اضافه می‌کند، و همچنین عملیات صفحه بندی، جستجو، مرتب سازی داده‌ها را در سمت کاربر انجام می‌دهد.

به طور خلاصه می‌توانید امکانات متعدد این پلاگین را در زیر مشاهده کنید:
صفحه بندی داده‌ها با تعداد رکوردهای قابل تغییر در هر صفحه (variable length pagination)
فیلتر کردن داده‌های بایند شده به جدول (on-the-fly filtering)

مرتب سازی داده‌ها بر اساس ستون‌های مختلف با قابلیت تشخیص نوع داده ستون (Multi-column sorting with data type detection)
تغییر اندازه ستون‌ها به صورت هوشمند (Smart handling of column widths)
نمایش داده‌ها در جدول از اکثر data sourceها (DOM، یک آرایه جاوا اسکریپتی، یک فایل، یا با استفاده از پردازش سمت سروری (سی شارپ، php و غیره))
قابلیت جهانی شدن یا منطبق شدن با زبان‌های مختلف دنیا (Fully Internationalisable)
قابلیت تعویض theme آن با استفاده از jQuery UI ThemeRoller
وجود داشتن 2900 آزمون واحد برای آن (backed by a suite of 2900 unit test)
وجود داشتن پلاگین‌های متعدد برای آن
و رایگان بودن آن
در این مقاله شما را به طور مقدماتی با این پلاگین آشنا خواهم کرد.
برای استفاده از این پلاگین ابتدا به اینجا مراجعه کرده و آنرا به همراه مثالهای آن که در یک فایل فشرده هستند را دانلود کنید.
بعد از دانلود و خارج کردن فایل دانلودی از حالت فشرده، وارد پوشه examples از آن که بشوید می‌توانید مثالهای متعدد در رابطه با این پلاگین را مشاهده نمائید.

مثال‌های این پلاگین یکی از بهترین منابع یادگیری آن هستند. در این سری از مقالات هم از روی همین مثالها پیش میرویم. برای این کار، بعد از مراجعه به پوشه examples فایل index.html را باز کنید و مثال اول را (Zero Configuration) کلیک کنید.

نتیجه حاصل از اجرای مثال Zero Configuration چیزی شبیه تصویر زیر است:

تصویر را شماره گزاری کرده ام تا بتوانم راحت‌تر آنرا برایتان تشریح کنم.

داده‌های درون جدول (10 تای اول) که در قسمت tbody جدول قرار دارند

قسمت thead جدول
قسمت tfoot جدول
اندازه صفحه (page size)
کادر جستجو که در کلیه ستون‌های جدول جستجویی را انجام می‌دهد و داده‌ها بر اساس آن فیلتر می‌شوند.

قابلیت مرتب سازی رکوردها بر اساس یک ستون خاص به صورت صعودی یا نزولی
اطلاعات مربوط به رکوردهای جاری و تعداد کل رکوردها
قابلیت تغییر صفحه با دکمه‌های previous و next

تشریح مثال Zero Configuration :

برای استفاده از این پلاگین، باید ارجاعی به کتابخانه jquery و نیز فایل jquery.dataTables.js وجود داشته باشد. این دو فایل در زیر پوشه media/js قرار گرفته اند.

```
<script type="text/javascript" language="javascript" src="../../media/js/jquery.js"></script>
<script type="text/javascript" language="javascript"
src="../../media/js/jquery.dataTables.js"></script>
```

و همچنین cssهای مربوطه به این پلاگین بدین صورت معرفی شده اند:

```
<style type="text/css" title="currentStyle">
    @import "../../media/css/demo_page.css";
    @import "../../media/css/demo_table.css";
</style>
```

در این مثال که ساده‌ترین مثال مربوط به این پلاگین است داده‌ها به صورت دستی در جدول قرار گرفته اند و روش‌های دیگر را به قسمت‌های بعد موکول می‌کنیم. اگر به source این مثال مراجعه کنید (از روی فایل اصلی و نه از طریق مرورگر) مشاهده می‌کنید که یک جدول html با id برابر با example وجود دارد که حاوی 57 سطر است (در قسمت tbody) که حاوی داده‌های جدول هستند. اما با مراجعه به source مثال از طریق مرورگر مشاهده می‌کنید تعداد این سطرها 10 تا هست و این بدین معنیه که پلاگین فقط تعداد رکوردهای مورد نیاز رو در قسمت tbody قرار می‌ده و از بقیه فاکتور می‌گیره و هر بار که کاربر به صفحه رو با دکمه‌های Previous و Next تغییر می‌ده این پلاگین قسمت tbody رو تغییر میده

این نکته هم جا نمونه که برای اعمال شدن پلاگین DataTables به یک جدول که به طور مثال id جدول example هست، به صورت زیر عمل می‌کنیم:

```
$(document).ready(function() {
    $('#example').dataTable();
} );
```

نظرات خوانندگان

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۰۵ ۱۴:۰

سلام مطلب جالبی بود
اما یک س.ال برای من پیش اومده، آیا این پلاگین قادر است داده‌های جدول را از سمت سرور فراخوانی کند؟ یعنی می‌تواند با تغییر صفحه داده‌های صفحه‌های بعدی یا قبلی را از سرور (از دیتابیس) فراخوانی کند؟

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۱/۰۷ ۱۸:۴

سلام، بله این امکان هست که با تعویض صفحه توسط کاربر درخواستی به سرور فرستاده بشه و داده‌ها از سرور دریافت بشن.
در قسمتهای بعدی به آنها اشاره خواهم کرد

نویسنده: کامی
تاریخ: ۱۳۹۲/۰۵/۲۶ ۰:۲۴

سلام
شما نظرتون در مورد پلاگین jTable چیه؟ ایا امکانات jTable بیشتری از dataTable نیست؟

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۵/۲۶ ۲۱:۵۷

سلام
بنده با jTable کار نکردم ، نمی‌دونم امکاناتش چی هست . البته تعصب خاصی روی datatable ندارم. اگه شما اطلاعاتی دارید خوشحال می‌شیم اونو با دیگران به اشتراک بزارید. :)

نویسنده: وحید م
تاریخ: ۱۳۹۲/۰۹/۰۷ ۹:۳۲

با سلام؛ به‌نظر شما چه کتابخانه ای برای گرید مناسب ،کم حجم و حرفه ای‌تر ؟
گرید توکارmvc یا- DataTable-Jqgrid-Kendo-telerik-awesome خیلی مهمه ممنون از شما.

عنوان: آموزش (jQuery) جی کوئری #5
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۲۳ ۱۸:۲۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Web Design, Learning jQuery

در ادامه مطلب قبلی آموزش (jQuery) جی کوئری #4 به ادامه بحث می‌پردازیم.
در پست قبل به بررسی **انتخاب عناصر بر اساس موقعیت** پرداختیم، در این پست به بحث " *استفاده از انتخاب کننده‌های سفارشی jQuery* " خواهیم پرداخت.

4-1- استفاده از انتخاب کننده‌های سفارشی jQuery

در پست‌های قبلی (^ و ^) تعدادی از انتخاب کننده‌های CSS که هر کدامشان موجب قدرت و انعطاف پذیری انتخاب اشیا موجود در صفحه می‌شوند را بررسی کردیم. با این وجود فیلترهای انتخاب کننده قدرتمندتری وجود دارند که توانایی ما را برای انتخاب بیشتر می‌کنند.
به عنوان مثال اگر بخواهید از میان تمام چک باکس ها، گزینه هایی را که تیک خورده اند انتخاب نمایید، از آنجا که تلاش برای مطابقت حالت‌های اولیه کنترل‌های HTML را بررسی می‌کنیم، jQuery انتخابگر سفارشی :checked را پیشنهاد می‌کند، که مجموعه از عناصر را که خاصیت checked آنها فعال باشد را برای ما برمی گرداند. براس مثال انتخاب کننده input تمامی المان‌های <input> را انتخاب می‌کند، و انتخاب کننده input:checked تمامی inputهایی را انتخاب می‌کند که checked هستند. انتخاب کننده سفارشی :checked یک انتخاب کننده خصوصیت CSS عمل می‌کند (مانند [foo=bar]). ترکیب این انتخاب کننده‌ها می‌تواند قدرت بیشتری به ما بدهد، انتخاب کننده هایی مانند radio:checked: و checkbox:checked: .
همانطور هم که قبلا بیان شد، jQuery علاوه بر پشتیبانی از انتخاب کننده‌های CSS تعدادی انتخاب کننده سفارشی را نیز شامل می‌شود که در جدول 3-2 شرح داده شده است.

جدول 3-2: انتخاب کننده‌های سفارشی jQuery

توضیح	انتخاب کننده
عناصری را انتخاب می‌کند که تحت کنترل انیمیشن می‌باشند. در پست‌های بعدی انیمیشن‌ها توضیح داده می‌شوند.	animated:
عناصر دکمه را انتخاب می‌کند، عناصری مانند input[type=submit]، input[type=reset]،) (input[type=button]، یا button)	button:
عناصر Checkbox را انتخاب می‌کند، مانند (input[type=checkbox]).	checkbox:
عناصر checkboxها یا دکمه‌های رادیویی را انتخاب می‌کند که در حالت انتخاب باشند.	checked:
عناصری ر انتخاب می‌کند که دارای عبارت foo باشند.	contains(foo) :
عناصر در حالت disabled را انتخاب می‌کند.	disabled:
عناصر در حالت enabledرا انتخاب می‌کند.	enabled:
عناصر فایل را انتخاب می‌کند، مانند (input[type=file]).	file:
عناصر هدر مانند h1 تا h6 را انتخاب می‌کند.	header:
عناصر مخفی شده را انتهاب می‌کند.	hidden:
عناصر تصویر را انتخاب می‌کند، مانند (input[type=image]).	image:

توضیح	انتخاب کننده
عناصر فرم مانند input ، select، textarea، button را انتخاب می‌کند.	input:
انتخاب کننده‌ها را برعکس می‌کند.	not(filter)
عناصری که فرزندی دارند را انتخاب می‌کند.	parent:
عناصر password را انتخاب می‌کند، مانند (input[type=password]).	password:
عناصر radio را انتخاب می‌کند، مانند (input[type=radio]).	radio:
دکمه‌های reset را انتخاب می‌کند، مانند (input[type=reset] یا button[type=reset]).	raset:
عناصری (عناصر option) را انتخاب می‌کند که در وضعیت selected قراردارند.	selected:
دکمه‌های submit را انتخاب می‌کند، مانند (input[type=submit] یا button[type=submit]).	submit:
عناصر text را انتخاب می‌کند، مانند (input[type=text]).	text:
عناصری را که در وضعیت visibleباشند انتخاب می‌کند.	visible:

بسیاری از انتخاب کننده‌های سفارشی jQuery بررسی شده برای انتخاب عناصر فرم ورود اطلاعات کاربر استفاده می‌شوند. این فیلترها قابلیت ادغام را دارند، برای مثال در زیر دستوری را به منظور انتخاب آن دسته از گزینه‌های Checkbox که تیک خورده اند و فعال هستند را مشاهده می‌کنید:

```
:checkbox:checked:enabled
```

این فیلترها و انتخاب کننده‌ها کاربردهای وسیعی در صفحات اینترنتی دارند، آیا آنها حالت معکوسی نیز دارند؟

استفاده از فیلتر not:

برای آنکه نتیجه انتخاب کننده‌ها را معکوس کنیم می‌توانیم از این فیلتر استفاده کنیم. برای مثال دستور زیر تمام عناصری را که checkBox نیستند را انتخاب می‌کند:

```
input:not(:checkbox)
```

اما استفاده از این فیلتر دقت زیادی را می‌طلبد زیرا به سادگی ممکن است با نتیجه ای غیر منتظره مواجه شویم.

استفاده از فیلتر has:

در اینجا دیدیم که CSS انتخاب کننده قدرتمندی را ارایه کرده است که فرزندران یک عنصر را در هر سطحی که باشند (حتی اگر فرزند مستقیم هم نباشند) انتخاب می‌کند. برای مثال دستور زیر تمام عناصر span را که در div معرفی شده باشند را انتخاب می‌کند:

```
div span
```

اما اگر بخواهیم انتخابی برعکس این انتخاب داشته باشیم، باید چه کنیم؟ برای این کار باید تمام divهایی که دارای عنصر span می‌باشد را انتخاب کرد. برای چنین انتخابی از فیلتر has: استفاده می‌کنیم. به دستور زیر توجه نمایید، این دستور تمام عناصر div

را که در آنها عنصر span معرفی شده است را انتخاب می‌کند:

```
div:has(span)
```

برای برخی انتخاب‌های پیچیده و مشکل، این فیلتر و مکانیزم بسیار کارا می‌باشد و به سادگی ما را به هدف دلخواه می‌رساند. فرض کنید می‌خواهیم آن خانه از جدول که دارای یک عنصر عکس خاص می‌باشد را پیدا کنیم. با توجه به این نکته که آن عکس از طریق مقدار src قابل تشخیص می‌باشد، با استفاده از فیلتر has: دستوری مانند زیر می‌نویسیم:

```
$('tr:has(img[src$="foo.png"])')
```

این دستور هر خانه از جدول را که این عکس در آن قرار گرفته باشد را انتخاب می‌کند.
همانگونه که دیدیم jQuery گزینه‌های بسیار متعددی را به منظور انتخاب عناصر موجود در صفحه برای ما مهیا کرده است که می‌توانیم هر عنصری از صفحه را انتخاب و سپس تغییر دهیم که تغییر این عناصر در پست‌های آینده بحث خواهد شد.

موفق و موید باشید.

نظرات خوانندگان

نویسنده: سید باقر شفیعی
تاریخ: ۱۳۹۲/۰۱/۲۸ ۱۳:۴۱

سلام مهندس
خیلی عالی بود - امیدارم وقت داشته باشی پست آموزشی بیشتری بذاری.
مرسی

نویسنده: رها
تاریخ: ۱۳۹۲/۱۰/۰۷ ۲۰:۳۰

سلام؛ من یه اسلاید شو ساده را از اموزشهای یه سایت انگلیسی زبان ساختم که مدتهاست دنبالش بودم
اما هنوز یه ایراد کوچولو داره و اونهم اینه که بعد از رسیدن به اخرین عکس برمیگرده به اول یعنی بصورت بک اسلاید میشه و اگر عکسها از سمت راست به چپ اسلاید میشوند وقتی به اخرین عکس میرسه تمام عکسها در کسری از ثانیه از چپ به راست برمیگردند. نمونه کد کوئری رو میزارم و ممنون میشم منو در این زمینه راهنمایی کنید که چطور کاری کنم با رسیدن به اخرین عکس به همون روش از سمت راست به چپ دوباره برگرده به عکس اول نه تمام عکسها رو از چپ به راست برگردونه ؟
اسکریپت فراخوانده شده :

```
< script src = "http://code.jquery.com/jquery-latest.js" ></ script >
```

کوئری نوشته شده :

```
<script type = "text/javascript" >
$(document).ready(function () {
   slideShow();
});

var n = 0;
function slideShow() {
   id = n % 5 + 1;
   leftpost = (1 - parseInt(id)) * 500 + "px";
   $("div.slider-item").animate({ left: leftpost }, 1500);
   n = n + 1;
   s = setTimeout("slideShow()", 3000);
}
 </ script >
```

فایل css :

```
<style type = "text/css" >
div#slider {
  width: 500px;
  height: 300px;
  margin: auto;
  overflow: hidden;
  border: 10px solid gray;
}
div#slider-mask {
  width: 500%;
  height: 100%;
}
div.slider-item {
  width: 20%;
  height: 100%;
  position: relative;
  float: left;
}
 </ style >
```

فایل html :

```
< div id = "slider" > < div id = "slider-mask" >
< div class = "slider-item" >< img src = "img1.jpg" alt = "1" />< / div >
< div class = "slider-item" >< img src = "img2.jpg" alt = "2" />< / div >
< div class = "slider-item" >< img src = "img3.jpg" alt = "3" />< / div >
 < div class = "slider-item" >< img src = "img4.jpg" alt = "4" />< / div >
< div class = "slider-item" >< img src = "img5.jpg" alt = "5" />< / div >
< / div > < / div >
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۰۸ ۰:۵

قسمت اسکریپتی رو اینطوری تغییر بدین

```
<script type="text/javascript">
$(document).ready(function () {
   slideShow();
});

var leftPos = 0;
var numberOfImages = 5;
var sliderWidth = 500;
var ltr = true;

function slideShow() {
   $("div.slider-item").animate({ left: leftPos + "px" }, 1500);

   if(ltr){
     leftPos -= sliderWidth;
   }
   else{
     leftPos += sliderWidth;
   }

   if((Math.abs(leftPos) == (numberOfImages-1) * sliderWidth) || (leftPos == 0)){
     ltr = !ltr;
   }

   //console.log({ leftPos:leftPos , ltr: ltr });
   s = setTimeout("slideShow()", 3000);
}
</script>
```

نویسنده: رها
تاریخ: ۱۳۹۲/۱۰/۲۲ ۱۲:۳۳

سلام؛ وقتی یه جی کوئری یا اسکریپت رو دانلود میکنیم و در ویژوآل استادیو باز میکنم بصورتی نوشته شده که تمام فایل در یک خط افقی هست و اگر بخواهیم ویرایشش کنیم نمیشه با اسکرول کردن موس در امتدادش حرکت کنیم. میخواستم ببینم آیا در ویژوآل استادیو ابزاری هست که اینچنین فایلها رو یکباره از حالت افقی و اینکه در یک خط هست هستند با طول زیاد رو بشکنه و بصورت زیر هم بنویسه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۲۲ ۱۲:۳۸

JavaScript Deobfuscator

عنوان: **ایجاد helper برای Nivo Slider در Asp.net Mvc**
نویسنده: علی حق جو
تاریخ: ۱۳۹۲/۰۱/۲۹ ۱۶:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net MVC, jQuery, NivoSlider, Mvc helper, MVC

کامپوننت‌های jQuery زیادی وجود دارند که توسط آنها میتوان تصاویر را بصورت زمانبندی شده و به همراه افکت‌های زیبا در سایت خود نشان داد. مانند اینجا .

در این قصد ایجاد helper برای کامپوننت NivoSlider را داریم.

1- یک پروژه Asp.net Mvc 4.0 ایجاد میکنیم.

2- سپس فایل jquery.nivo.slider.pack.js ، فایل‌های css مربوط به این کامپوننت و چهار تم موجود را از سایت این کامپوننت و یا درون سورس مثال ارائه شده دریافت می‌کنیم.

3- به کلاس BundleConfig رفته و کدهای زیر را اضافه میکنیم:

```
#region Nivo Slider

   bundles.Add(new StyleBundle("~/Content/NivoSlider").Include("~/Content/nivoSlider/nivo-
slider.css"));
   bundles.Add(new
```

```
StyleBundle("~/Content/NivoSliderDefaultTheme").Include("~/Content/nivoSlider/themes/default/default.cs
s"));
   bundles.Add(new
StyleBundle("~/Content/NivoSliderDarkTheme").Include("~/Content/nivoSlider/themes/dark/dark.css"));
   bundles.Add(new
StyleBundle("~/Content/NivoSliderLightTheme").Include("~/Content/nivoSlider/themes/light/light.css"));
   bundles.Add(new
StyleBundle("~/Content/NivoSliderBarTheme").Include("~/Content/nivoSlider/themes/bar/bar.css"));
   bundles.Add(new
ScriptBundle("~/bundles/NivoSlider").Include("~/Scripts/jquery.nivo.slider.pack.js"));

#endregion
```

سپس در فایل shared آنها را بصورت زیر اعمال میکنیم:

```
<!DOCTYPE html>
<html>
<head>
    <title>@ViewBag.Title</title>
    <script type="text/javascript" src="~/Scripts/jquery-1.9.1.min.js"></script>
    @Styles.Render("~/Content/NivoSlider",
"~/Content/NivoSliderDefaultTheme","~/Content/NivoSliderLightTheme")
    @Scripts.Render("~/bundles/NivoSlider")
</head>
<body>
    @RenderBody()
    @RenderSection("scripts", required: false)
</body>
</html>
```

4- یک پوشه با عنوان helper به پروژه اضافه میکنیم. سپس کلاس‌های زیر را به آن اضافه می‌کنیم :

```
public class NivoSliderHelper
 {
        #region Fields

        private string _id = "nivo1";
        private List<NivoSliderItem> _models = null;
        private string _width = "100%";
        private NivoSliderTheme _theme = NivoSliderTheme.Default;
        private string _effect = "random"; // Specify sets like= 'fold;fade;sliceDown'
        private int _slices = 15; // For slice animations
        private int _boxCols = 8; // For box animations
        private int _boxRows = 4; // For box animations
        private int _animSpeed = 500; // Slide transition speed
        private int _pauseTime = 3000; // How long each slide will show
        private int _startSlide = 0; // Set starting Slide (0 index)
        private bool _directionNav = true; // Next & Prev navigation
        private bool _controlNav = true; // 1;2;3... navigation
        private bool _controlNavThumbs = false; // Use thumbnails for Control Nav
        private bool _pauseOnHover = true; // Stop animation while hovering
        private bool _manualAdvance = false; // Force manual transitions
        private string _prevText = "Prev"; // Prev directionNav text
        private string _nextText = "Next"; // Next directionNav text
        private bool _randomStart = false; // Start on a random slide
        private string _beforeChange = ""; // Triggers before a slide transition
        private string _afterChange = ""; // Triggers after a slide transition
        private string _slideshowEnd = ""; // Triggers after all slides have been shown
        private string _lastSlide = ""; // Triggers when last slide is shown
        private string _afterLoad = "";

        #endregion

        string makeParameters()
        {
            var builder = new StringBuilder();
            builder.Append("{");
            builder.Append(string.Format("effect:'{0}'", _effect));
            builder.AppendLine(string.Format(",slices:{0}", _slices));
            builder.AppendLine(string.Format(",boxCols:{0}", _boxCols));
            builder.AppendLine(string.Format(",boxRows:{0}", _boxRows));
            builder.AppendLine(string.Format(",animSpeed:{0}", _animSpeed));
```

```
            builder.AppendLine(string.Format(",pauseTime:{0}", _pauseTime));
            builder.AppendLine(string.Format(",startSlide:{0}", _startSlide));
            builder.AppendLine(string.Format(",directionNav:{0}", _directionNav.ToString().ToLower()));
            builder.AppendLine(string.Format(",controlNav:{0}", _controlNav.ToString().ToLower()));
            builder.AppendLine(string.Format(",controlNavThumbs:{0}",
_controlNavThumbs.ToString().ToLower()));
            builder.AppendLine(string.Format(",pauseOnHover:{0}", _pauseOnHover.ToString().ToLower()));
            builder.AppendLine(string.Format(",manualAdvance:{0}",
_manualAdvance.ToString().ToLower()));
            builder.AppendLine(string.Format(",prevText:'{0}'", _prevText));
            builder.AppendLine(string.Format(",nextText:'{0}'", _nextText));
            builder.AppendLine(string.Format(",randomStart:{0}", _randomStart.ToString().ToLower()));
            builder.AppendLine(string.Format(",beforeChange:{0}", _beforeChange));
            builder.AppendLine(string.Format(",afterChange:{0}", _afterChange));
            builder.AppendLine(string.Format(",slideshowEnd:{0}", _slideshowEnd));
            builder.AppendLine(string.Format(",lastSlide:{0}", _lastSlide));
            builder.AppendLine(string.Format(",afterLoad:{0}", _afterLoad));
            builder.Append("}");
            return builder.ToString();
        }
        public NivoSliderHelper (
            string id,
            List<NivoSliderItem> models,
            string width = "100%",
            NivoSliderTheme theme = NivoSliderTheme.Default,
            string effect = "random",
            int slices = 15,
            int boxCols = 8,
            int boxRows = 4,
            int animSpeed = 500,
            int pauseTime = 3000,
            int startSlide = 0,
            bool directionNav = true,
            bool controlNav = true,
            bool controlNavThumbs = false,
            bool pauseOnHover = true,
            bool manualAdvance = false,
            string prevText = "Prev",
            string nextText = "Next",
            bool randomStart = false,
            string beforeChange = "function(){}",
            string afterChange = "function(){}",
            string slideshowEnd = "function(){}",
            string lastSlide = "function(){}",
            string afterLoad = "function(){}")
        {
            _id = id;
            _models = models;
            _width = width;
            _theme = theme;
            _effect = effect;
            _slices = slices;
            _boxCols = boxCols;
            _boxRows = boxRows;
            _animSpeed = animSpeed;
            _pauseTime = pauseTime;
            _startSlide = startSlide;
            _directionNav = directionNav;
            _controlNav = controlNav;
            _controlNavThumbs = controlNavThumbs;
            _pauseOnHover = pauseOnHover;
            _manualAdvance = manualAdvance;
            _prevText = prevText;
            _nextText = nextText;
            _randomStart = randomStart;
            _beforeChange = beforeChange;
            _afterChange = afterChange;
            _slideshowEnd = slideshowEnd;
            _lastSlide = lastSlide;
            _afterLoad = afterLoad;
        }
        public IHtmlString GetHtml()
        {
            var thm = "theme-" + _theme.ToString().ToLower();
            var sb = new StringBuilder();
            sb.AppendLine("<div style='width:" + _width + ";height:auto;margin:0px auto' class='" + thm
+ "'>");
            sb.AppendLine("<div id='" + _id + "' class='nivoSlider'>");
            foreach (var model in _models)
            {
                string img = string.Format("<img src='{0}' alt='{1}' title='{2}' />",
```

```
model.ImageFilePath, "", model.Caption);
                string item = "";
                if (model.LinkUrl.Trim().Length > 0 &&
                    Uri.IsWellFormedUriString(model.LinkUrl, UriKind.RelativeOrAbsolute))
                {
                    item = string.Format("<a href='{0}'>{1}</a>", model.LinkUrl, img);
                }
                else
                {
                    item = img;
                }
                sb.AppendLine(item);
            }
            sb.AppendLine("</div>");
            sb.AppendLine("</div>");

            sb.AppendLine("<script type='text/javascript'>");
            //sb.AppendLine("$('#" + _id + "').parent().ready(function () {");
            sb.Append("$(document).ready(function(){");
            sb.AppendLine("$('#" + _id + "').nivoSlider(" + makeParameters() + ");");
            //sb.AppendLine("$('.nivo-controlNav a').empty();");//semi hack for rtl layout
            //sb.AppendLine("$('#" + _id + "').nivoSlider();");
            sb.AppendLine("});");
            sb.AppendLine("</script>");

            return new HtmlString(sb.ToString());
        }
    }
```

```
public enum NivoSliderTheme
    {
        Default, Light, Dark, Bar,
    }
```

```
public class NivoSliderItem
    {
        public string ImageFilePath { get; set; }
        public string Caption { get; set; }
        public string LinkUrl { get; set; }
    }
```

-6سپس برای استفاده از این helper یک کنترلر به پروژه اضافه میکنیم مانند Home :

```
public class HomeController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }
        public virtual ActionResult NivoSlider()
        {
            var models = new List<NivoSliderItem>
            {
                new NivoSliderItem()
                    {
                        ImageFilePath =Url.Content("~/Images/img1.jpg"),
                        Caption = "عنوان اول",
                        LinkUrl = "http://www.google.com",
                    },
                new NivoSliderItem()
                    {
                        ImageFilePath =Url.Content("~/Images/img2.jpg"),
                        Caption = "#htmlcaption",
                        LinkUrl = "",
                    },
                new NivoSliderItem()
                    {
                        ImageFilePath =Url.Content("~/Images/img3.jpg"),
                        Caption = "عنوان سوم",
                        LinkUrl = "",
                    },
                new NivoSliderItem()
                    {
                        ImageFilePath =Url.Content("~/Images/img4.jpg"),
                        Caption = "عنوان چهارم",
                        LinkUrl = "",
```

```
                    },
                new NivoSliderItem()
                    {
                        ImageFilePath =Url.Content("~/Images/img5.jpg"),
                        Caption = "عنوان پنجم",
                        LinkUrl = "",
                    }
            };
            return PartialView("_NivoSlider", models);
        }
    }
```

7- سپس ویوی Home را نیز ایجاد میکنیم:

```
@{
    ViewBag.Title = "Index";
}

<h2>Nivo Slider Index</h2>

@{ Html.RenderAction("NivoSlider", "Home");}
```

8- یک پارشال ویو نیز برای رندر کردن NivoSliderهای خود ایجاد میکنیم:

```
@using NivoSlider.Helper
@model List<NivoSliderItem>
@{
    var nivo1 = new NivoSliderHelper("nivo1", Model.Skip(0).Take(3).ToList(), theme:
NivoSliderTheme.Default, width: "500px");
    var nivo2 = new NivoSliderHelper("nivo2", Model.Skip(3).Take(2).ToList(), theme:
NivoSliderTheme.Light, width: "500px");
}
@nivo1.GetHtml()

<div id="htmlcaption">
    <strong>This</strong> is an example of a <em>HTML</em> caption with <a href="#">a link</a>.
</div>

<br />
<br />
@nivo2.GetHtml()
```

نکته اول : اگر بخواهیم بر روی تصویر موردنظر متنی نمایش دهیم کافیست خاصیت Caption را مقداردهی کنیم. برای نمایش توضیحاتی پیچیده‌تر (مانند نمایش یک div و لینک و غیره) بعنوان توضیحات یک تصویر خاص باید خاصیت Caption را بصورت "#{نام عنصر}" مقداردهی کنیم و ویژگی class مربوط به عنصر موردنظر را با nivo-html-caption مقداردهی کنیم. (همانند مثال بالا)

نکته دوم : این کامپوننت به همراه 4 تم (deafult- light- bar- dark) ارائه شده است. موقع استفاده از تم‌های دیگر باید رفرنس مربوط به آن تم را نیز اضافه کنید.

امیدوارم مفید واقع شود. دریافت پروژه نمونه این مقاله

نظرات خوانندگان

نویسنده: bahman
تاریخ: ۱۳۹۲/۰۲/۰۸ ۱۱:۴۴

سلام تشکر میکنم .خیلی عالی بود.
فقط یه مشکلی هست اونم نبودن WebGrease درون DLLهای شماست.
چطوری میتونم پیداش کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۸ ۱۱:۴۷

این dll جزو فایل‌های استاندارد یک پروژه ASP.NET MVC **4** است. در اینجا پیوست نشده، چون نیازی نیست هربار تمام این فایل‌های تکراری را دریافت کنید.
فایل‌های آن در دو پوشه زیر بر روی سیستم شما موجود هستند: (اگر ASP.NET MVC4 را نصب کرده باشید)
C:\Program Files\Microsoft ASP.NET\ASP.NET MVC **4** \Packages
C:\Program Files\Microsoft ASP.NET\ASP.NET Web Pages\v **2.0** \Packages

نویسنده: bahman
تاریخ: ۱۳۹۲/۰۲/۰۸ ۱۱:۵۴

مشکل با nuget حل شد.

عنوان: **نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۴ ۱۲:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Debugging, MVC, FireBug, jQuery, jQuery Ajax

هر از چندگاهی سؤال «این مثال jQuery رو نمی‌تونم اجرا یا باز سازی کنم» در این سایت یا سایت‌های مشابه تکرار می‌شوند. بنابراین بهتر است نحوه عیب یابی برنامه‌های ASP.NET مبتنی بر jQuery را یکبار با هم مرور کنیم. در اینجا، مثال تهیه یک Image Slider را که پیشتر در سایت مطرح شده است، به نحوی دیگر بررسی خواهیم کرد:

1) فراموش می‌کنیم تا اسکریپت اصلی jQuery را به درستی پیوست و مسیردهی کنیم.
2) مسیر Generic handler دیگری را ذکر می‌کنیم.
3) مسیرهای تصاویری را که Image slider باید نمایش دهد، کاملا بی‌ربط ذکر می‌کنیم.
4) خروجی JSON نامربوطی را بازگشت می‌دهیم.
5) یکبار هم یک استثنای عمدی دستی را در بین کدها قرار خواهیم داد.

و ... بعد سعی می‌کنیم با استفاده از Firebug عیوب فوق را یافته و اصلاح کنیم؛ تا به یک برنامه قابل اجرا برسیم.

معرفی برنامه‌ای که کار نمی‌کند!

یک برنامه ASP.NET Empty web application را آغاز کنید. سپس سه پوشه Content، Scripts و Images را به آن اضافه نمائید. در این پوشه‌ها، اسکریپت‌های نمایش دهنده تصاویر، Css آن و تصاویری که قرار است نمایش داده شوند، قرار می‌گیرند:

سپس یک فایل default.aspx و یک فایل OrbitHandler.ashx را نیز به پروژه با محتویات ذیل اضافه کنید: (در این دو فایل، 5 مورد مشکل ساز یاد شده لحاظ شده‌اند)
محتویات فایل OrbitHandler.ashx.cs مطابق کدهای ذیل است:

```
using System.Collections.Generic;
using System.IO;
using System.Web;
using System.Web.Script.Serialization;

namespace OrbitWebformsTest
{
    public class Picture
    {
        public string Title { set; get; }
        public string Path { set; get; }
    }

    public class OrbitHandler : IHttpHandler
    {
        IList<Picture> PicturesDataSource()
        {
            var results = new List<Picture>();
            var path = HttpContext.Current.Server.MapPath("~/Images");

            foreach (var item in Directory.GetFiles(path, "*.*"))
            {
                var name = Path.GetFileName(item);
                results.Add(new Picture
                {
                    Path = /*"Images/" + name*/ name,
                    Title = name
                });
            }

            return results;
        }

        public void ProcessRequest(HttpContext context)
        {
            var items = PicturesDataSource();
            var json = /*new JavaScriptSerializer().Serialize(items)*/ string.Empty;
            throw new InvalidDataException("همینطوری");
            context.Response.ContentType = "text/plain";
            context.Response.Write(json);
        }

        public bool IsReusable
        {
            get { return false; }
        }
    }
}
```

در اینجا جهت سهولت دموی برنامه (و همچنین امکان باز تولید آن توسط خوانندگان)، از بانک اطلاعاتی استفاده نشده و عمدا از یک لیست جنریک تشکیل شده در حافظه کمک گرفته شده است. تصاویر برنامه در پوشه Images واقع در ریشه سایت، قرار دارند. بنابراین توسط متد PicturesDataSource، فایل‌های این پوشه را یافته و مطابق ساختار کلاس Picture بازگشت می‌دهیم. نهایتا این اطلاعات به ظاهر قرار است با فرمت JSON بازگشت داده شوند تا بتوان نتیجه را توسط افزونه Orbit استفاده کرد.

همچنین کدهای صفحه ASPX ایی که قرار است (به ظاهر البته) از این Generic handler استفاده کند به نحو ذیل است:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="default.aspx.cs"
Inherits="OrbitWebformsTest._default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <link href="Content/orbit-1.2.3.css" rel="stylesheet" type="text/css" />
    <script src="Script/jquery-1.5.1.min.js" type="text/javascript"></script>
    <script src="Scripts/jquery.orbit-1.2.3.min.js" type="text/javascript"></script>
</head>
<body>
    <form id="form1" runat="server">
    <div id="featured">
    </div>
    </form>
    <script type="text/javascript">
```

```
        $(function () {
            $.ajax({
                url: "Handler.ashx",
                contentType: "application/json; charset=utf-8",
                success: function (data) {
                    $.each(data, function (i, b) {
                        var str = '<img src="' + b.Path + '" alt="' + b.Title + '"/>';
                        $("#featured").append(str);
                    });
                    $('#featured').orbit();
                },
                dataType: "json"
            });
        });
    </script>
</body>
</html>
```

خوب! اگر پروژه را اجرا کنیم، کار نمی‌کند. یک مستطیل مشکی رنگ در کنار صفحه ظاهر شده و همین! حالا چکار باید کرد؟

مراحل عیب یابی برنامه‌ای که کار نمی‌کند!

ابتدا برنامه را در فایرفاکس باز کرده و سپس افزونه Firebug را با کلیک بر روی آیکن آن، بر روی سایت فعال می‌کنیم. سپس یکبار بر روی دکمه F5 کلیک کنید تا مجددا مراحل بارگذاری سایت تحت نظر افزونه Firebug فعال شده، طی شود.

اولین موردی که مشهود است، نمایش عدد 3، کنار آیکن فایرباگ می‌باشد. این عدد به معنای وجود خطاهای اسکریپتی در کدهای ما است.
برای مشاهده این خطاها، بر روی برگه Console آن کلیک کنید:

بله. مشخص است که مسیر دهی فایل jquery-1.5.1.min.js صحیح نبوده و همین مساله سبب بروز خطاهای اسکریپتی گردیده است. برای اصلاح آن سطر زیر را در برنامه تغییر دهید:

```
<script src="Scripts/jquery-1.5.1.min.js" type="text/javascript"></script>
```

پیشتر پوشه Script ذکر شده بود که باید تبدیل به Scripts شود.

مجددا دکمه F5 را فشرده و سایت را با تنظیمات جدید اجرا کنید. اینبار در برگه Console و یا در برگه شبکه فایرباگ، خطای یافت نشدن Generic handler نمایان می‌شوند:

برای رفع آن به فایل default.aspx مراجعه و بجای معرفی Handler.ashx، نام OrbitHandler.ashx را وارد کنید.
مجددا دکمه F5 را فشرده و سایت را با تنظیمات جدید اجرا کنید.

اگر به برگه کنسول دقت کنیم، بروز استثناء در کدها تشخیص داده شده و همچنین در برگه Response پاسخ دریافتی از سرور، جزئیات صفحه خطای بازگشتی از آن نیز قابل بررسی و مشاهده است.
اینبار به فایل OrbitHandler.ashx.cs مراجعه کرده و سطر throw new InvalidDataException را حذف می‌کنیم. در ادامه برنامه را کامپایل و مجددا اجرا خواهیم کرد.

با اجرای مجدد سایت، تبادل اطلاعات صحیحی با فایل OrbitHandler.ashx برقرار شده است، اما خروجی خاصی قابل مشاهده نیست. بنابراین بازهم سایت کار نمی‌کند.

برای رفع این مشکل، متد ProcessRequest را به نحو ذیل تغییر خواهیم داد:

```
public void ProcessRequest(HttpContext context)
        {
            var items = PicturesDataSource();
            var json = new JavaScriptSerializer().Serialize(items);
            context.Response.ContentType = "text/plain";
            context.Response.Write(json);
        }
```

برنامه را کامپایل کرده و اجرا می‌کنیم. برنامه اجرا می‌شود، اما باز هم کار نمی‌کند. مشکل از کجاست؟

بله. تمام تنظیمات به نظر درست هستند، اما در برگه شبکه فایرباگ تعدادی خطای 404 و یا «یافت نشد»، مشاهده می‌شوند.

مشکل اینجا است که مسیرهای بازگشت داده شده توسط متد Directory.GetFiles، مسیرهای مطلقی هستند؛ مانند c:\path\images\01.jpg و جهت نمایش در یک وب سایت مناسب نمی‌باشند. برای تبدیل آن‌ها به مسیرهای نسبی، اینبار کدهای متد تهیه منبع داده را به نحو ذیل ویرایش می‌کنیم:

```
IList<Picture> PicturesDataSource()
        {
            var results = new List<Picture>();
```

```
        var path = HttpContext.Current.Server.MapPath("~/Images");

        foreach (var item in Directory.GetFiles(path, "*.*"))
        {
            var name = Path.GetFileName(item);
            results.Add(new Picture
            {
                Path = "Images/" + name,
                Title = name
            });
        }

        return results;
    }
```

در این کدها فقط قسمت Path ویرایش شده است تا به مسیر پوشه Images واقع در ریشه سایت اشاره کند.
اینبار اگر برنامه را اجرا کنیم، بدون مشکل کار خواهد کرد.

بنابراین در اینجا مشاهده کردیم که اگر «برنامه‌ای مبتنی بر jQuery کار نمی‌کند»، چگونه باید قدم به قدم با استفاده از فایرباگ و امکانات آن، به خطاهایی که گزارش می‌دهد و یا مسیرهایی را که یافت نشد بیان می‌کند، دقت کرد تا بتوان برنامه را عیب یابی نمود.

سؤال مهم: اجرای کدهای jQuery Ajax فوق، چه تغییری را در صفحه سبب می‌شوند؟

اگر به برگه اسکریپت‌ها در کنسول فایرباگ مراجعه کنیم، امکان قرار دادن breakpoint بر روی سطرهای کدهای جاوا اسکریپتی نمایش داده شده نیز وجود دارد:

در اینجا همانند VS.NET می‌توان برنامه را در مرورگر اجرا کرده و تگ‌های تصویر پویای تولید شده را پیش از اضافه شدن به صفحه، مرحله به مرحله بررسی کرد. به این ترتیب بهتر می‌توان دریافت که آیا src بازگشت داده شده از سرور فرمت صحیحی دارد یا خیر و آیا به محل مناسبی اشاره می‌کند یا نه. همچنین در برگه HTML آن، عناصر پویای اضافه شده به صفحه نیز بهتر مشخص هستند:

Console HTML CSS Script DOM Net Cookies YSlow Pixel P... Firefi

Edit | body < html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
  <body>
    <form id="form1" action="default.aspx" method="post">
      <div class="aspNetHidden">
      <div class="orbit-wrapper" style="width: 48px; height: 19px;">
        <div id="featured" class="orbit" style="width: 48px; height: 19px;">
          <img alt="01.png" src="01.png" style="z-index: 1; display: block; left:
          -48px;"> jQuery15100924838756054342 2=Object { olddisplay="inline" }
          <img alt="02.jpg" src="02.jpg" style="display: block; left: -48px;
          z-index: 1;">
          <img alt="03.jpg" src="03.jpg" style="display: block; left: 0px; z-index:
          3;">
        </div>
```

نظرات خوانندگان

نویسنده: صادق
تاریخ: ۱۳۹۲/۰۲/۰۴ ۱۲:۵۱

ممنون آقای نصیری - برای کروم چطور، آیا developer tools کفایت میکند یا ابزار بهتری سراغ دارید؟
البته من گاهی که نیاز به دستکاری ریسپانس و ریکوئست باشه از فیدلر استفاده میکنم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۴ ۱۲:۵۵

تمام مواردی که در بحث جاری با فایرباگ مطرح شدن به صورت توکار در developer tools مربوط به کروم هم وجود دارند و مباحث آن تقریبا یکی است.

نویسنده: محسن جمشیدی
تاریخ: ۱۳۹۲/۰۲/۰۶ ۱۰:۴۹

یکی از مواردی هم که ممکن است آزار دهنده باشد خطای سینتکسی است که در FireFox با زدن Ctrl+Shift+J قابل مشاهده خواهد بود.

نویسنده: Abolfazl
تاریخ: ۱۳۹۲/۰۲/۰۸ ۲۳:۴۸

سلام آقای نصیری ..ممنونم ..خیلی عالی بود
من قبلا هم این مشکل را داشتم و نمی‌تونستم برطرفش کنم ..الان درست شد..مرسی

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۰۹ ۱۶:۴۲

```
$(document).ready(function () {
     $(':radio').click(function () {
         debugger; //فعال سازی دیباگر ویژوال استودیو
      });
});
```

یک نکته جانبی است برای فعال سازی دیباگر خود ویژوال استودیو در حین کار با جی‌کوئری

نویسنده: ahmad.valipour
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۲:۱۹

با سلام
سوالی برام پیش اومده.اگر ما 10 تا عکس تو پوشه image داشته باشیم ولی فقط 5 تا اونها رو در صفحه ارجاع بهشون داشته باشیم، فقط اون 5 تا به طرف کلاینت میرن.درسته؟
پس چرا وقتی با firebug مسیر عکس رو تغییر میدیم به عکس دیگه، میشناسه اون عکس رو ،و نشون میده؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۲:۲۵

همکاری متقابل موتور مرورگر و فایرباگ. فایرباگ درخواست می‌کنه، مرورگر دریافت و ارائه.
چیزی مثل دیباگر در VS.NET. زمانیکه مثلا در کدهای کار با Entity framework روی سطری break point قرار می‌دید و خروجی

یک کوئری را بررسی می‌کنید، این دیباگر قابلیت دریافت مقادیر بررسی شده و حتی نشده را هم از بانک اطلاعاتی دارا است (حتی اگر این مقادیر، در کوئری اولیه درخواست نشده باشند؛ نوعی lazy loading در اینجا صورت می‌گیرد)

نویسنده: ابوالفضل روشن ضمیر
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۱:۵

سلام
من اولین بار که اجرا می‌کنم تصویر را نشون نمی‌ده به این صورت نمایش میده :

بعد از لود صفحه یک بار که صفحه را Refresh کنم همه چیز درست میشه ؟
در قسمت Console هم خطایی وجود ندارد ....
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۱:۲۹

حداقل دو علت می‌تونه داشته باشه:
الف) تصاویر رو نمی‌تونه پیدا کنه، یا صفحه کش شده بیش از حد. قسمت «اجرای کدهای jQuery Ajax فوق، چه تغییری را در صفحه سبب می‌شوند؟» را بررسی کنید که چه آدرسی توسط کدهای جی‌کوئری در حال پردازش است.
همچنین کش شدن نتایج قبلی رو هم می‌شود غیرفعال کرد:

```
$.ajax({
  cache: false /* گاهی از اوقات خصوصا برای آی ای نیاز است */
});
```

ب) چند وقت قبل در یکی از بحث‌های سایت دیدم که مورد زیر رعایت نشده بود و کدهای جی‌کوئری کار نمی‌کردند:

```
<script type="text/javascript">
        $(function () {
            // کدهای جی‌کوئری در اینجا
        });
</script>
```

اجرای کدهای جی‌کوئری نیازی به DOM حاضر و آماده دارند که توسط متد document ready آن مانند کدهای فوق باید تدارک دیده شود. نیازی به این کد نخواهد بود اگر اسکریپت‌ها در آخر صفحه و پیش از بسته شدن تگ body اضافه بشن.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۰۷/۰۷ ۱۰:۵۸

با تشکر از مطلب بسیار کاربردیتون ، در هنگام استفاده از URL Routing همانطور که قبلا راهنماییم کرده بودید برای آدرس فایل‌های جاوا اسکریپت از

```
<%=ResolveUrl("~/App_Themes/MainTheme/jquery.js")%>
```

استفاده کردم و مشکل حل شد ولی برای یو آر ال این تصاویر آیکون که در Jquery تعریف شدند میتونید کمکم کنید

```
<script type="text/javascript">
    $(document).ready(function () {

        $('#exampleMenu').sweetMenu({
            top: 200,
            padding: 8,
            iconSize: 48,
            easing: 'easeOutBounce',
            duration: 500,
            icons: [
'images/home.png',
'images/comments.png',
'images/red_heart.png',
'images/computer.png',
'images/male_user.png',
'images/yellow_mail.png'
]
        });
    });
</script>
```

اگر آدرس آیکون‌ها را به صورت

'http://site.ir/images/home.png'

تعریف کنم مشکل حل میشه ولی فکر کنم راه حل درستی نباشه . بسیار ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۷ ۱۱:۲۸

امکان قرار دادن کدهای سمت سرور داخل اسکریپت‌ها هم هست؛ مثلا:

```
data: "{'username': '" + $('#<%= TextBox1.ClientID %>').val() + "'}",
```

به این شرط که این اسکریپت داخل صفحه runat=server دار باشد یا داخل head ایی با این مشخصات.

عنوان: **پلاگین DataTables کتابخانه jQuery - قسمت دوم**
نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۲/۰۶ ۰:۲۵
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, DataTables

در قسمت قبلی شما را با DataTables آشنا کردم. به طور خلاصه نحوه اعمال کردن DataTables به یک جدول ساده html را گفتیم که با این کار به صورت پیش فرض، امکاناتی مثل فیلتر کردن داده ها، صفحه بندی و مرتب سازی آنها و نیز اعمال شدن استایل‌های css به همین جدول html خام اضافه می‌شود. نکته مهم در مثال قبلی این بود که داده‌های درون این جدول با کدنویسی خام html فراهم شدند، اما این را در نظر داشته باشید که اکثریت مواقع باید داده‌ها از یک بانک اطلاعاتی دریافت شوند و سپس درون جدول قرار بگیرند.

در این قسمت سعی خواهیم کرد تا منبع داده جدول را یک آرایه جاوا اسکریپتی و سپس کالکشنی از آبجکتهای جاوا اسکریپتی (json) در نظر بگیریم و نیز برخی ویژگی‌های پیش فرض پلاگین را غیر فعال نمائیم.

فرض کنید می‌خواهید لیستی از اطلاعات دانشجویان شامل نام (FirstName)، نام خانوادگی (LastName)، و سن (Age) را نمایش دهید. اطلاعات قرار است در جدول زیر قرار بگیرند:

```
<table id="std-grid">
      <thead>
           <th>نام</th>
           <th>نام خانوادگی</th>
           <th>سن</th>
     </thead>
     <tbody>

     </tbody>
</table>
```

مشاهده می‌کنید که این جدول فقط شامل قسمت header است و در بدنه آن هیچ سطری قرار نگرفته است. در این مثال اطلاعات از یک آرایه جاوا اسکریپتی باید خوانده شوند و تبدیل به html شده و در نهایت درون قسمت <tbody></tbody> آن تزریق شوند. خوشبختانه DataTables برای این کار امکانات آماده ای را در اختیار قرار می‌دهد. این کار بدین صورت قابل انجام است:

```
<script>
        $(document).ready(function() {
            $('#std-grid').dataTable({
                "aaData": [
                        ["24" ,"پارسائی" ,"پژمان"],
                        ["25" ,"الیاسی" ,"سعید"],
                        ["20" ,"گلزار" ,"محمد رضا"],
                        ["19" ,"ایرانی" ,"آرش"],
                        ["22" ,"فرمانی" ,"مرتضی"],
                        ["23" ,"حمیدیان" ,"سعید"],
                        ["23" ,"پارسانیا" ,"امین"],
                        ["24" ,"فقیهی" ,"محمد امین"],
                        ["25" ,"خرمی" ,"محمد"],
                        ["20" ,"امیریان" ,"سینا"],
                        ["19" ,"ایرانی" ,"آرش"],
                        ["22" ,"فرزانه" ,"وحید"],
                        ["23" ,"فرمانی" ,"امیر علی"],
                        ["23" ,"حسینی" ,"امین"],
            });
        });

    </script>
```

شرح کد:
aaData : یک آرایه دو بعدی (که به آن ماتریس یا آرایه ای از آرایه‌ها هم گفته می‌شود) است که مقادیر سلول هایی را نشان

می‌دهد که در جدول قرار خواهند گرفت. تعداد ستون‌ها در این آرایه دو بعدی باید با تعداد ستون‌های جدول html متناظر یکسان باشند.

در مثال بالا از یک ماتریس به عنوان منبع داده استفاده شد. منبع داده می‌تواند به فرمت json نیز باشد. البته در این صورت باید ستون‌های جدول html را هم به پلاگین معرفی کنید، بدین صورت:

```
$(document).ready(function() {
            $('#std-grid').dataTable({
                "aaData": [
                    {"FirstName" : "پژمان", "LastName" : "پارسائی", "Age" : "24"},
                    { "FirstName": "سعید", "LastName": "الیاسی", "Age": "25" },
                    { "FirstName": "محمد رضا", "LastName": "گلزار", "Age": "24" },
                    { "FirstName": "آرش", "LastName": "ایرانی", "Age": "24" },
                    { "FirstName": "مرتضی", "LastName": "فرمانی", "Age": "24" },
                    { "FirstName": "سعید", "LastName": "حمیدیان", "Age": "24" },
                    { "FirstName": "امین", "LastName": "پارسانیا", "Age": "24" },
                    { "FirstName": "محمد امین", "LastName": "فقیهی", "Age": "24" },
                    { "FirstName": "محمد", "LastName": "خرمی", "Age": "24" },
                    { "FirstName": "سینا", "LastName": "امیریان", "Age": "24" },
                    { "FirstName": "آرش", "LastName": "ایرانی", "Age": "24" },
                    { "FirstName": "وحید", "LastName": "فرزانه", "Age": "24" },
                    { "FirstName": "امیر علی", "LastName": "فرمانی", "Age": "24" },
                    { "FirstName": "امین", "LastName": "حسینی", "Age": "24" },
                ],
                "aoColumns": [
                      { "mDataProp": "FirstName" },
                      { "mDataProp": "LastName" },
                      { "mDataProp": "Age" }
                ]
            });
        });
```

aaData : همان طور که گفته شد در این قسمت دیتاهای درون جدول آورده می‌شوند و در این مثال آنها به فرمت json نوشته شده اند.

aoColumns : در این قسمت باید اسم ستون‌های جدول ذکر شوند.

غیرفعال کردن بعضی از ویژگی‌های پیش فرض DataTables

همان طور که گفته شد پلاگین DataTables به صورت پیش فرض ویژگی‌های مرتب سازی (sorting)، صفحه بندی (paging)، فیلتر کردن داده‌ها (filtering)، و غیره را به جدول مورد نظرش اعمال می‌کند. و بدین صورت قابل تغییر است:

```
$('#std-grid').dataTable({
          "bPaginate": false,
          "bLengthChange": false,
          "bFilter": false,
          "bSort": false,
          "bInfo": true,
          "bAutoWidth": false
});
```

bPaginate : بیان می‌کند آیا صفحه بندی سطرهای جدول فعال باشد یا نه.

bLengthChange : در صورتی که قابلیت صفحه بندی فعال باشد ، بیان می‌کند که کاربر بتواند اندازه صفحه را تغییر دهد یا نه.

bFilter : بیان می‌کند آیا قابلیت فیلتر کردن داده‌ها فعال باشد یا نه.

bSort : بیان می‌کند قابلیت مرتب سازی داده‌های جدول فعال باشد یا نه.

bInfo : بیان می‌کند که قسمت info زیر گرید نشان داده شود یا نه (در این قسمت اطلاعاتی راجع به تعداد کل رکوردهای بایند شده به جدول و نیز رکوردهای درون صفحه جاری نشان داده می‌شود)

bAutoWidth : در صورتی که این گزینه فعال باشد اندازه عرض هر ستون به صوتر خودکار توسط DataTables مقدار دهی خواهد شد.

مقدارهای قابل قبول برای هر کدام از این خصوصیات : true یا false

کدهای مربوط به این مثال را می‌توانید از لینک زیر دریافت کنید:
DataTables-DoteNetTips-Tutorial-02.zip

نظرات خوانندگان

نویسنده: farzad
تاریخ: ۱۳۹۲/۰۴/۲۵ ۲۰:۵

با سلام
آیا این پلاگین قابلیت حذف و یا ویرایش داده‌ها رو هم می‌ده؟

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۴/۲۵ ۲۱:۲۷

سلام
بله، نمونه پیاده سازی شده در mvc رو می‌تونید توی لینک زیر مشاهده کنید:

(ASP.NET MVC Editable Table (jQuery DataTables and ASP.NET MVC integration - Part II

عنوان: **پر کردن مقادیر Enum در کلاینت**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۲:۱۵
آدرس: www.dotnettips.info
برچسب‌ها: JSON, jQuery, jQuery-Tips, Enum, dropdown

در برنامه‌های وب امروز نیازی به فراخوانی ثوابت که در طول حیات برنامه انگشت شمار تغیر میکنند نیست و با توجه به استفاده از فرامین و متدهای سمت کلاینت احتیاج هست تا این ثوابت بار اول لود صفحه به کلاینت پاس داده شوند.

میتوان در این گونه موارد از قابلیت‌های گوناگونی استفاده کرد که در اینجا ما با استفاده از یک فیلد مخفی و json مقدار را به کلاینت پاس میدهیم و در این مثال در سمت کلاینت نیز دراپ دان را با این مقادیر پر میکنیم:

```
public enum PersistType
    {
        Persistable = 1,
        NotPersist = 2,
        AlwaysPersist = 3
    }
```

لیست را باید قبل از پر کردن در فیلد مخفی به json بصورت serialize شده تبدیل کرد، برای این منظور از JavaScriptSerializer موجود در اسمبلی‌های دات نت در متد زیر استفاده شده:

```
public static string ConvertEnumToJavascript(Type t)
        {
            if (!t.IsEnum) throw new Exception("Type must be an enumeration");

            var values = System.Enum.GetValues(t);
            var dict = new Dictionary<int, string>();

            foreach (object obj in values)
            {
                string name = System.Enum.GetName(t, obj);
                dict.Add(Convert.ToInt32(System.Enum.Format(t, obj, "D")), name);
            }

            return new JavaScriptSerializer().Serialize(dict);
        }
```

با توجه به اینکه در سمت کلاینت مقدار json ذخیره شده در فیلد مخفی را میتوان به صورت آبجکت برخورد کرد پس یک متد در سمت کلاینت این آبجکت را در loop قراد داده و درمتغییری در فایل جاوا اسکریپت نگهداری میکنیم:

```
var Enum_PersistType = null;

function SetEnumTypes() {
    Enum_PersistType = JSON.parse($('#hfJsonEnum_PersistType').val());
}
```

و در هر قسمت که نیاز به مقدار enum بود با توجه به ایندکس مقدار را برای نمایش ازاین متغییر بیرون میکشیم:

```
function GetPersistTypeTitle_Concept(enumId) {
    return Enum_PersistType[enumId];
}
```

برای مثال در dropdown در سمت کلاینت این نوع استفاده شده و در حالتی از صفحه فقط برای نمایش عنوان آن احتیاج به دریافت آن از سمت سرور باشد میتوان از این روش کمک گرفت

و یا در سمت javascript میتوان با استفاده از jQuery مقادیر متغییر را در dropdown پر کرد.

```
function FillDropdown() {
    $("#ddlPersistType").html("");
    $.each(Enum_PersistType, function (key, value) {
        $("#ddlPersistType").append($("<option></option>").val(key).html(value));
    });
}
```

FillDropdownListOnClient.zip

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۱:۵۴

با تشکر از شما.

فایل Newtonsoft.Json.dll در پروژه شما هست. JavaScriptSerializer توکار دات نت ازش استفاده نمی‌کنه. فقط از اسمبلی System.Web.Extensions.dll هست که استفاده می‌کنه.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۱:۵۷

ممنون از شما، برای ادامه این سری لازم میشده که در آینده اضافه میشه.
شما میتونید در صورت دلخواه کد قسمت serialize رو با این کتابخانه بنویسید:

```
//return new JavaScriptSerializer().Serialize(dict);
            return Newtonsoft.Json.JsonConvert.SerializeObject(dict);
```

نویسنده: سام ناصری
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۳:۴۴

به نظر من موضوع رو خیلی پیچیده کردی. برای تولید json لازم نیست که از کتابخانه خاصی استفاده کنی با همون استرینگ منیوپولیشن ساده هم میشه این کار را کرد:

```
public static class EnumHelper
{
    public static Dictionary<string,EnumValueType> ToDictionary<EnumType,EnumValueType>()
    {
        return
Enum.GetValues(typeof(EnumType)).Cast<EnumValueType>().ToDictionary(i=>Enum.GetName(typeof(EnumType),i)
,i=>i);
    }
    public static string ToJson<EnumType>()
    {
        return "{" + string.Join(",",
ToDictionary<EnumType,int>().Select(i=>string.Format(@"{{""{0}"":{1}}}",i.Key,i.Value)).ToArray())+ "}"
;
    }
}
```

کلاس ساده بالا به سادگی json مورد نیاز را تولید میکند.
در ضمن برای اینجکت کردنش به صفحه هم لازم نیست که از فیلد مخفی استفاده کنی. به جاش json را مستقیم در محل مورد نظر رندر کن:
در Asp.Net MVC Razor:

```
var x = @EnumHelper.ToJson<MyEnum()
```

در Asp.Net Web Forms :

```
var x = <%=EnumHelper.ToJson<MyEnum>()%>
```

همچنین همانطور که در مثالهای فوق نشان داده ام حتی لازم نیست از JSON.Parse استفاده کنی.
البته من اینها را بر اساس ذهنیاتم خیلی سریع نوشتم و کدهای فوق را تست نکرده ام که ببینم درست کار میکنند یا نه. اما منظورم این بود که بپرسم چرا از فیلد مخفی استفاده کردی و چرا از JSON.Parse استفاده کردی و اینکه چرا از JavaScriptSerializer استفاده کردی؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۴:۲۶

در حالت کلی بهتره که از JavaScriptSerializer استفاده بشه چون می‌تونه یک سری escape حروف خاص رو لحاظ کنه.

نویسنده: سام ناصری
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۵:۶

موضوع این مقاله درباره Enum است و نه ارسال داده‌های کلی به کلاینت. پس آیا فکر میکنید در اینجا چیزی برای اسکیپ شدن وجود داره؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۶:۳۲

در مورد پیچیدگی صحبت کردید. راه شما به مراتب پیچیده‌تر است از روش مطرح شده و خوانایی کمتری داره. به علاوه هدف از ارائه مقالات بهتره ارائه راه حل‌هایی باشه تا حد امکان عمومی تا این که یک سری هک خاص مطرح بشه فقط مختص به یک روش خاص که فقط در یک مساله مشخص قابل استفاده باشه. بعد هم اگر کسی این هک رو جای دیگری استفاده کرد، چون نمی‌دونه یک سری از کاراکترها باید escape بشن، در ضمن کار گیر میفته. دید دادن برای حل مساله اینجا شاید بیشتر مطرح باشه تا حل مساله با یک هک ساده که فقط همینجا قابل استفاده است. همچنین زمانیکه یک سری متد تست شده داخل فریم ورک هست چرا باید رفت سراغ هک؟

ضمنا در ASP.NET MVC نیاز دارید که یک Html.Raw رو هم اضافه کنید و گرنه اطلاعات درج شده در صفحه encode می‌شن و در متغیر جاوا اسکریپتی قابل استفاده نخواهند بود.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۲/۰۲/۱۲ ۲۳:۸

در موردی مطلبی که آقای ناصری فرمودند باید بگم زمانیکه برنامه به سمت چند زبانه میره اهمبت پیدا میکنه که میتونید برای مثال مقدار یا لیستی از مقادیر متنی برای زبان خاصی رو با resource خودش به فیلد مخفی پاس بدید و در نمایش پیغام‌های مختلف سمت کلاینت اسنفاده کنید. مثل متن پیغام هایی که خاص ارتباط ajax میباشد که به زبان‌های مختلف ارائه کرد.

نویسنده: سام ناصری
تاریخ: ۱۳۹۲/۰۲/۱۳ ۴:۳۳

من کلاً نمیفهمم. در ضمن من سه تا سوال مطرح کردم(پاراگراف آخر کامنتم) که من باز هم نمیفهمم این جواب کدومشونه.

عنوان: **آشنایی با پلاگین TickTack برای Mask ورودی کاربر**
نویسنده: محسن موسوی
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۷:۵
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, Plugin, TimePicker

همانطور که می‌دانیم پلاگین‌های جی‌کوئری، نقش مهمی را در محیط وب ایفا می‌کنند. در اینجا با یکی از این پلاگین‌ها و چگونگی استفاده از آن آشنا میشویم.

برای آشنایی با نوشتن Plugin در jQuery ، می‌توان مباحث پیشین این سایت را دنبال کرد.(JQuery Plugins #1 و JQuery Plugins #2)

jQueryTickTack Plugin :

این Plugin برای ایجاد یک TextBox برای ورود زمان توسط کاربر استفاده می‌شود. با توجه به اینکه قبلاً چند Plugin برای این کار نوشته شده است ولی هر کدام از آنها معایب و مزایای خاص خود را داشتند، برای نمونه می‌توانید به این سایت مراجعه کنید.

ویژگی‌های این Plugin عبارتند از:

1- تنظیم زمان پیش فرض

2- کنترل حداقل و حداکثر زمان وارد شده

3- تغییر ساعت و دقیقه بوسیله کلیدهای جهتی بالا و پایین

4- تغییر انتخاب ساعت و دقیقه بوسیله کلیدهای جهتی چپ و راست

5- تغییر ساعت و دقیقه بوسیله فشردن اعداد روی صفحه کلید

چگونگی استفاده از این Plugin

ابتدا کتابخانه jQuery و این پلاگین را به صفحه خود اضافه نمایید و سپس کدهای زیر را برای استفاده از این Plugin اضافه نمایید:

```
jQuery(document).ready(function () {
            $("#TextBox1").TickTack();
            $("#TextBox2").TickTack({
                 initialTime: '8:44',
                minHour: 8,
                minMinute: 0,
                maxHour: 22,
                maxMinute: 40
            });
        });
```

در ادامه به بررسی تنظیمات انجام شده در این پلاگین می‌پردازیم:
initialTime : زمان اولیه جهت نمایش به کاربر (حتما بایستی ساعت و دقیقه بوسیله ' : ' از یکدیگر جداشوند)

minHour : حداقل ساعت ورودی

minMinute : حداقل دقیقه ورودی

maxHour : حداکثر ساعت ورودی

maxMinute : حداکثر دقیقه ورودی

پس از انجام این تنظیمات و اجرا کردن برنامه،TextBox شما به صورت زیر نمایش داده می‌شود:

please input time : 13:44

پس از انتخاب TextBox ، قسمت ساعت به صورت پیش فرض انتخاب می‌شود و کاربر باید ساعت مد نظر را وارد کند؛ در اینجا، عدد اول ساعت، مد نظر است.

please input time : 13:44

برای نمونه در اینجا عدد 2 توسط کاربر وارد می‌شود؛ پس از ورود عدد و با توجه به تنظیمات انجام شده، ساعت به صورت اتوماتیک به حداکثر مقداری که می‌تواند بپذیرد تغییر می‌کند (در این مثال چون کاربر عدد 2 را وارد کرده و در تنظیمات انجام شده حداکثر ساعت دریافتی 22 و حداکثر دقیقه 40 تعریف شده است، ساعت به صورت پیشفرض به 22:40 تغییر می‌یابد)

please input time : 22:40

و پس از وارد کردن عدد دوم ساعت توسط کاربر مکان نما به قسمت دقیقه منتقل می‌شود که در این جا عدد اول دقیقه مد نظر است

please input time : 21:40

وارد کردن عدد 3 برای دقیقه

please input time : 21:30

وارد کردن عدد دوم دقیقه

please input time : 21:35

پس از وارد کردن کامل دقیقه مکان نما دوباره به قسمت ساعت باز می‌گردد.

در ادامه دوستان علاقمند لطفا جهت بهبود کیفیت کار، باگ و یا مشکلات کدنویسی را اطلاع دهند.

با تشکر

نظرات خوانندگان

نویسنده: بهروز
تاریخ: ۱۳۹۲/۰۶/۲۵ ۱۶:۱۸

با سلام خدمت جناب آقای محسن موسوی

یه مشکلی داشتم تو این پلاگین که وقتی از تو کد به تکست باکس مقداری میدیم اون مقدارو نمیگیره و فقط مقدار پیشفرض خودشو نشون میده... میخواستم در صورت امکان راهنمایی بفرمایید.

با تشکر فراوان

و من الله توفیق...

نویسنده: محسن موسوی
تاریخ: ۱۳۹۲/۰۶/۲۵ ۱۷:۵۴

سلام
از طریق سرور پلاگین را صدا بزنید یا اینکه از طریق سرور یک مقدار به متغییر جاوااسکریپتی بدهید.

نویسنده: سعید حر
تاریخ: ۱۳۹۲/۰۷/۰۶ ۱۱:۳۷

سلام
در آدرس https://ticktack.codeplex.com/ هیچ فایلی برای دانلود وجود ندارد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۶ ۱۱:۴۵

به برگه سورس پروژه مراجعه کنید.

نویسنده: محسن موسوی
تاریخ: ۱۳۹۲/۰۷/۰۶ ۱۲:۲۹

با تشکر از آقای نصیری
هنوز بازخورد خاصی از پروژه نگرفتم.در هفته‌ی آتی احتمالا release پروژه را ارائه میدهم.

عنوان: **افزونه farsiInput جهت ورودی فقط فارسی در صفحات وب**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۰ ۲۱:۰
آدرس: www.dotnettips.info
گروه‌ها: jQuery, Persian

گاهی از اوقات نیاز است کاربر در یک جعبه متنی، فقط متن فارسی وارد کند؛ حتی اگر صفحه کلید او فارسی نباشد و یا بنابر درخواست او، جهت بالا رفتن سرعت ورود اطلاعات یک چنین قابلیتی نیاز می‌شود. چندین سال قبل farsitype.js اینکار را انجام می‌داد. این اسکریپت با مرورگرهای جدید سازگار نیست و برای نمونه initKeyEvent آن در نگارش‌های قدیمی فایرفاکس کار می‌کرد، در کروم هیچ وقت پشتیبانی نشد (به نام initKeyboardEvent موجوداست؛ اما برای جایگزین کردن حروف عمل نمی‌کند) و مدتی است که فایرفاکس هم به دلایل امنیتی آن‌را غیرفعال کرده است.

به همین جهت افزونه farsiInput، که کدهای آن‌را در ادامه مشاهده می‌کنید، تهیه گردید. این افزونه تا این تاریخ با IE، فایرفاکس، کروم و اپرا سازگار است و توسط آن کاربر بدون نیاز به داشتن یک صفحه کلید فارسی می‌تواند فارسی تایپ کند. برای سوئیچ به حالت انگلیسی، دکمه Scroll lock باید روشن شود و این مورد توسط پارامتر changeLanguageKey قابل تغییر است.

```
// <![CDATA[
(function ($) {
    $.fn.farsiInput = function (options) {
        var defaults = {
            changeLanguageKey: 145 /* Scroll lock */
        };
        var options = $.extend(defaults, options);

        var lang = 'fa';

        var keys = new Array(1711, 0, 0, 0, 0, 1608, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1705,
1572, 0, 1548,
                             1567, 0, 1616, 1571, 8250, 0, 1615, 0, 0, 1570, 1577, 0, 0, 0, 1569, 1573,
0, 0, 1614, 1612, 1613, 0, 0,
                             8249, 1611, 171, 0, 187, 1580, 1688, 1670, 0, 1600, 1662, 1588, 1584,
1586, 1740, 1579, 1576, 1604, 1575,
                             1607, 1578, 1606, 1605, 1574, 1583, 1582, 1581, 1590, 1602, 1587, 1601,
1593, 1585, 1589, 1591, 1594, 1592);

        var substituteChar = function (charCode, e) {
            if (navigator.appName == "Microsoft Internet Explorer") {
                window.event.keyCode = charCode;
            }
            else {
                insertAtCaret(String.fromCharCode(charCode), e);
            }
        };

        var insertAtCaret = function (str, e) {
            var obj = e.target;
            var startPos = obj.selectionStart;
            var endPos = obj.selectionEnd;
            var scrollTop = obj.scrollTop;
            obj.value = obj.value.substring(0, startPos) + str + obj.value.substring(endPos,
obj.value.length);
            obj.focus();
            obj.selectionStart = startPos + str.length;
            obj.selectionEnd = startPos + str.length;
            obj.scrollTop = scrollTop;
            e.preventDefault();
        };

        var keyDown = function (e) {
            var evt = e || window.event;
            var key = evt.keyCode ? evt.keyCode : evt.which;
            if (key == options.changeLanguageKey) {
                lang = (lang == 'en') ? 'fa' : 'en';
                return true;
            }
        };

        var fixYeKeHalfSpace = function (key, evt) {
            var originalKey = key;
            var arabicYeCharCode = 1610;
            var persianYeCharCode = 1740;
            var arabicKeCharCode = 1603;
            var persianKeCharCode = 1705;
            var halfSpace = 8204;
```

```
            switch (key) {
                case arabicYeCharCode:
                    key = persianYeCharCode;
                    break;
                case arabicKeCharCode:
                    key = persianKeCharCode;
                    break;
            }

            if (evt.shiftKey && key == 32) {
                key = halfSpace;
            }

            if (originalKey != key) {
                substituteChar(key, evt);
            }
        };

        var keyPress = function (e) {
            if (lang != 'fa')
                return;

            var evt = e || window.event;
            var key = evt.keyCode ? evt.keyCode : evt.which;
            fixYeKeHalfSpace(key, evt);
            var isNotArrowKey = (evt.charCode != 0) && (evt.which != 0);
            if (isNotArrowKey && (key > 38) && (key < 123)) {
                var pCode = (keys[key - 39]) ? (keys[key - 39]) : key;
                substituteChar(pCode, evt);
            }
        }

        return this.each(function () {
            var input = $(this);
            input.keypress(function (e) {
                keyPress(e);
            });
            input.keydown(function (e) {
                keyDown(e);
            });
        });
    };
})(jQuery);
// ]]>
```

مثالی از نحوه بکارگیری آن:

```
<html>
<head>
    <title>تکست باکس فارسی</title>
    <script type="text/javascript" src="jquery-1.9.1.min.js"></script>
    <script type="text/javascript" src="jquery.farsiInput.js"></script>
    <style type="text/css">
        input, textarea
        {
            font-family: tahoma;
            font-size: 9pt;
        }
    </style>
</head>
<body>
    <input dir="rtl" id='text1' />
    <br />
    <textarea dir="rtl" id='text2' rows="15" cols="84"></textarea>
    <script type="text/javascript">
        $(function () {
            $("#text1, #text2").farsiInput();
        });
    </script>
</body>
</html>
```

دریافت کدهای کامل افزونه farsiInput

farsi_input.zip

نظرات خوانندگان

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۲/۰۲/۲۰ ۲۱:۲۰

خیلی ممنون :) ... زیاد از حد کاربردیه :دی

نویسنده: کاوه احمدی
تاریخ: ۱۳۹۲/۰۲/۲۱ ۱۸:۴۴

البته FarsiType اسکریپت قدیمی‌ای است که به نظرم این روزها کارکرد خود را ازدست داده است. اما اشارات شما در مورد کارکرد آن در مرورگرهای متفاوت صحیح نیست. دست کم بنده در آخرین نسخه‌های مرورگرها می‌توانم به درستی از آن استفاده کنم. احتمالا شما از آخرین نسخه آن بهره نگرفته‌اید :)

http://farsitype.ir

نویسنده: الهه
تاریخ: ۱۳۹۲/۰۲/۲۱ ۱۹:۵۷

چقدر خوب بود ، ممنونم خیلی دنبالش بودم

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۲/۲۱ ۲۱:۲۷

بدلیل سازگاری مناسب با کتابخانه JQuery افزونه FarsiInput را بهتر از FarsiType ارزیابی می‌کنم.
همچنین FarsiType قابلیت هایی دارد که پیشنهاد می‌کنم به FarsiInput اضافه شود. برای مثال تغییر Direction که کار آسانی است و همچنین تغییر زبان با ctrl + Space.
غیرفعال شدن در صورتی که صفحه کلید فارسی است، پیشنهاد نمی‌شود چرا که نیاز مبرمی به جاینشینی حروف عربی مورد نیاز است.
در کل FarsiInput سبک‌تر است و آینده بهتری را برای آن می‌توان متصور شد. همانطور که یکی دیگر از دوستان اشاره کردند این افزونه بیش از حد کاربردی است. مخصوصاً در برنامه‌های کاربردی تحت وب!
با تشکر از وحید نصیری و همچنین کاوه احمدی برای همه تلاششان.

نویسنده: یزدان
تاریخ: ۱۳۹۲/۰۲/۲۴ ۱۱:۳۴

بسیار خوب .
فـقـط اگه برای اعداد هم فارسی میشد کامل بود . اگر مقدوره قابلیت فارسی سازی اعداد رو هم اضافه کنید .

نویسنده: علیرضا
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۶:۲۴

برای فارسی کردن اعداد متغیر keys را به این صورت عوض کنید

```
var keys = new Array(1711, 0, 0, 0, 0, 1608, 0, 0, 0, 1776, 1777, 1778, 1779, 1780, 1781, 1782, 1783,
1784, 1785, 0, 1705, 1572, 0, 1548,
                                1567, 0, 1616, 1571, 8250, 0, 1615, 0, 0, 1570, 1577, 0, 0, 0, 1569, 1573,
0, 0, 1614, 1612, 1613, 0, 0,
                                8249, 1611, 171, 0, 187, 1580, 1688, 1670, 0, 1600, 1662, 1588, 1584,
1586, 1740, 1579, 1576, 1604, 1575,
                                1607, 1578, 1606, 1605, 1574, 1583, 1582, 1581, 1590, 1602, 1587, 1601,
1593, 1585, 1589, 1591, 1594, 1592);
```

همچنین برای تغییر جهت صفحه متد each را تغییر دهید

```
return this.each(function () {
            var input = $(this);
            input.css("direction","rtl");
            input.keypress(function (e) {
                keyPress(e);
            });
            input.keydown(function (e) {
                keyDown(e);
            });
```

نویسنده: علیرضا
تاریخ: ۱۳۹۲/۱۰/۰۳ ۱۳:۳

چگونه اعداد داخل editor را فارسی کنیم؟ با jquery داخل ادیتور را select کردم ولی کار نکرد!

editor: ckeditor

نویسنده: شفیقی
تاریخ: ۱۳۹۳/۰۳/۰۳ ۱۴:۵۹

باسلام
متاسفانه هم فایل شما و هم فایل FarsiType این مشکل رو داره که اگه در صفحه از Ajax و UpdatePanel استفاده بشه ، فقط در اولین مرتبه که صفحه لود میشه کار میکنه . همین که یکبار صفحه PostBack بشه ، دیگه کار نمیکنه و انگلیسی تایپ میشه .
لطفا درصورت امکان یه راهکار بدید تا قابل استفاده باشه .
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۳ ۱۷:۵

« استفاده‌ی همزمان از آپدیت پنل ASP.Net و پلاگین‌های جی‌کوئری »

نویسنده: ناهید
تاریخ: ۱۳۹۳/۰۷/۲۶ ۱۲:۳۶

سلام
اگر بخوایم فقط حروف فارسی باشه و اعداد رو نزنه چیکار باید بکنیم.
در ضمن مرسی مفید و عالی بود

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۲۶ ۱۵:۲۱

در متدهای keydown و keypress فوق، اگر متد e.preventDefault() فراخوانی شود، دکمه‌ی فشرده شده نمایش داده نخواهد شد. در همینجا بازه‌ی key دریافتی را بررسی کنید. مثلا 97 مساوی a است تا 122 که z است. اگر خارج از آن بود متد e.preventDefault() را فراخوانی کنید.

نویسنده: نسترن بصیرت
تاریخ: ۱۳۹۳/۰۹/۱۸ ۱۲:۳۷

با تشکر؛ این کد فقط توی صفحات html خوب کار میکنه و توی صفحات asp کار نمی‌کنه. برای اینکه بتونم توی صفحه asp هم

ازش استفاده کنم باید چیکار کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۸ ۱۲:۴۹

هیچ تفاوتی نمی‌کند. فقط اگر از ASP.NET Web forms استفاده می‌کنید(ASP در دهه‌ی نود میلادی منقرض شد)، نیاز است با مفاهیمی مانند ClientID آشنا باشید:

- آشنایی با انواع Control IDها در ASP.Net

مانند:

```
$('#<%= TextBox1.ClientID %>')
```

- همچنین در نگارش‌های اخیر ASP.NET Web forms می‌شود تولید این Id را کنترل کرد :

```
<asp:TextBox runat="server" ID="txtName" ClientIDMode="Static" />
```

عنوان: پلاگین DataTables کتابخانه jQuery - قسمت سوم
نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۲/۲۰ ۲۱:۲۵
آدرس: www.dotnettips.info
برچسب‌ها: jQuery, DataTables, jQuery-Plugins

در این قسمت اطلاعات را به صورت ajax از یک فایل متنی می‌خوانیم و آنها را در جدول قرار می‌دهیم. سپس به سفارشی کردن بعضی از قسمت‌های DataTables خواهیم پرداخت.

دریافت اطلاعات به صورت ajax از یک فایل متنی

فرض کنید که اطلاعات در یک فایل txt به صورت اشیاء جاوا اسکریپتی ذخیره شده اند، و این فایل بر روی سرور قرار دارد. می‌خواهیم از این فایل به عنوان منبع داده استفاده کرده و اطلاعات درون آن را به صورت ajax دریافت کرده و در یک جدول html تزریق کنیم. خوشبختانه با استفاده از امکاناتی که این پلاگین تهیه کرده است این کار به سادگی امکان پذیر است.

همان طور که در اینجا بیان شده است ، فرض کنید که جدولی داشته باشیم و بخواهیم اطلاعات راجع به مرورگرهای مختلف را در آن نمایش دهیم. قصد داریم این جدول شامل قسمتهای header و footer و نیز body باشد، بدین صورت:

```
<table id="browsers-grid">
    <thead>
       <tr>
          <th width="20%">موتور رندرگیری</th>
          <th width="25%">مرورگر</th>
          <th width="25%">(پلتفرم (ها</th>
          <th width="15%">نسخه موتور</th>
          <th width="15%">نمره css</th>
       </tr>
    </thead>

    <tbody>

    </tbody>

    <tfoot>
       <tr>
          <th>موتور رندرگیری</th>
          <th>مرورگر</th>
           <th>(پلتفرم (ها</th>
          <th>نسخه موتور</th>
          <th>نمره css</th>
       </tr>
    </tfoot>
</table>
```

برای هر ستون از این جدول عرضی در نظر گرفته شده است. اگر این کار انجام نشود به صورت خودکار به تمام ستونها عرض داده می‌شود.

داده هایی که باید در بدنه جدول قرار بگیرند، در یک فایل متنی روی سرور قرار دارند. محتویات این فایل چیزی شبیه زیر است:

```
{
   "aaData": [
      {"engine":"Trident", "browser":"Internet Explorer 4.0", "platform":"Win95+", "version":"4",
"grade":"X"},
      {"engine":"Trident", "browser":"Internet Explorer 5.0", "platform":"Win95+", "version":"5",
"grade":"C"},
      {"engine":"Trident", "browser":"Internet Explorer 5.5", "platform":"Win95+", "version":"5.5",
"grade":"A"}
   ]
}
```

همان طور که مشاهده می‌کنید فرمت ذخیره داده‌ها در این فایل به صورت json یا اشیاء جاوا اسکریپتی است. این اشیاء باید به خصوصیت aaData نسبت داده شوند که در قسمت قبل راجع به آن توضیح دادیم. تعداد این اشیاء 57 تا بود که برای سادگی بیشتر 3 تا از آنها را اینجا ذکر کردیم.

اسکریپتی که داده‌ها را از فایل متنی خوانده و آنها را در جدول قرار می‌دهد هم بدین صورت خواهد بود:

```
$(document).ready(function () {
        $('#browsers-grid').dataTable({
            "sAjaxSource": "datasource/objects.txt",
            "bProcessing": true,
            "aoColumns": [
                { "mDataProp": "engine" },
                { "mDataProp": "browser" },
                { "mDataProp": "platform" },
                { "mDataProp": "version" },
                { "mDataProp": "grade" }
            ]
    });
});
```

شرح کد:

sAjaxSource : رشته

نوع داده ای که قبول می‌کند رشته ای و بیان کننده آدرسی است که داده‌ها باید از آنجا دریافت شوند. در اینجا داده‌ها در فایل متنی objects.txt در پوشه datasource قرار دارند.

bProcessing : بولین

نوع داده‌های قابل قبول این خصوصیت true یا false هست و بیان کننده این است که یک پیغام loading تا زمانی که داده‌ها دریافت شوند و در جدول قرار بگیرند نمایش داده شوند یا خیر.

تنظیم کردن گزینه‌های اضافی دیگر

sAjaxDataProp : رشته

همان طور که گفتیم در فایل متنی که حاوی اشیاء json بود ، این اشیاء را به متغیری به اسم aaData منتسب کردیم. این نام را می‌توان تغییر داد مثلا فرض کنید در فایل متنی داده‌ها به متغیری به اسم data منتسب شده اند:

```
{
   "data": [
      {"engine":"Trident", "browser":"Internet Explorer 4.0", "platform":"Win95+", "version":"4",
"grade":"X"},
      {"engine":"Trident", "browser":"Internet Explorer 5.0", "platform":"Win95+", "version":"5",
"grade":"C"},
      {"engine":"Trident", "browser":"Internet Explorer 5.5", "platform":"Win95+", "version":"5.5",
"grade":"A"}
   ]
}
```

در این صورت باید خصوصیت sAjaxDataProp را به همان نامی که در فایل متنی مشخص کرده اید مقداردهی کنید، در غیر این صورت داده‌های جدول هیچ گاه بارگذاری نخواهند شد. بدین صورت:

```
"sAjaxDataProp": "data"
```

یا اگر داده‌ها را بدین صورت در فایل متنی ذخیره کرده اید:

```
{ "data": { "inner": [...] } }
```

آنگاه خصوصیت sAjaxDataProp بدین صورت مقداردهی خواهد شد:

```
"sAjaxDataProp": "data.inner"
```

sPaginationType : رشته
نحوه صفحه بندی و حرکت بین صفحات مختلف را بیان می‌کند. اگر با two_buttonمقدار دهی شود (مقدار پیش فرض) حرکت بین صفحات مختلف به وسیله دکمه‌های Next و Previous امکان پذیر خواهد بود. اگر با full_numbersمقدار دهی شود حرکت بین صفحات با دکمه‌های Next و Previous ، و همچنین دکمه‌های First و Last و نیز شماره صفحه فعلی و دو صفحه بعدی و دو صفحه قبلی قابل انجام است.

شکل الف) صفحه بندی به صورت full_numbers

bLengthChange : بولین
بیان می‌کند کاربر بتواند اندازه صفحه را تغییر دهید یا نه. به صورت پیش فرض این گزینه true است. اگر آن به false مقدار دهی شود لیست بازشونده مربوط به اندازه صفحه مخفی خواهد شد.

aLengthMenu : آرایه یک بعدی یا دو بعدی
به صورت پیش فرض در لیست باز شونده مربوط به تعداد رکوردهای قابل نمایش در هر صفحه اعداد 10 ، 25 ، 50 ، و 100 قرار دارند.

شکل ب) لیست بازشونده شامل اندازه‌های صفحه

در صورتی که بخواهیم این گزینه‌ها را تغییر دهیم باید خصوصیت aLengthMenu را مقدار دهی کنیم. اگر مقداری که به این خصوصیت می‌دهیم یک آرایه یک بعدی باشد، مثلا

```
"aLengthMenu": [25, 50, 100, -1],
```

نتیجه یک لیست باز شوند است که دارای چهار عنصر است که value و text آنها یکی است. (نکته: چهارمین عنصر از لیست بالا دارای مقدار 1- خواهد بود که با انتخاب این گزینه تمام رکوردها نمایش می‌یابند). اما اگر می‌خواهیم که text و value این عناصر با هم فرق کند از یک آرایه دو بعدی استفاده خواهیم کرد، مثلا:

```
"aLengthMenu": [[25, 50, 100, -1], ["همه", "صد", "پنجاه", "بیست و پنج"]],
```

iDisplayLength : عدد صحیح
تعداد رکوردهای قابل نمایش در هر صفحه هنگامی که داده‌ها در جدول ریخته می‌شوند را معین می‌کند. می‌توانید این را مقداری بدهید که در خصوصیت aLengthMenu ذکر نشده است، مثلا 28 تا.

sDom : رشته
پلاگین DataTables به صورت پیش فرض لیست بازشونده اندازه صفحه و کادر متن مربوط به جستجو را در بالای جدول داده‌ها اضافه می‌کند، و نیز اطلاعات دیگر و همچنین امکانات مربوط به صفحه بندی را به قسمت پایین جدول اضافه می‌کند. شما می‌توانید موقعیت این عناصر را با استفاده از پارامتر sDom تغییر دهید.

نحو (syntax) مقداری که پارامتر sDom قبول می‌کند مقداری عجیب و غریب است، مثلا:

```
'<"top"iflp<"clear">>rt<"bottom"iflp<"clear">>'
```

این خط بیان می‌کند که در قسمت بالای جدول یک تگ div با کلاس top قرار بگیرد. در این تگ قسمت اطلاعات (یعنی Showing x to xx from xxx entries) (با حرف i) ، کادر جستجو (با حرف f) ، لیست بازشونده مربوط به اندازه صفحه (با حرف l) ، و نیز قسمت صفحه بندی (با حرف p)قرار خواهند گرفت. در انتهای تگ div با کلاس top، یک تگ div با کلاس clear قرار خواهد گرفت. بعد قسمت مربوط به پیغام loading (با حرف r) و بعد با حرف t جدول حاوی داده‌ها قرار می‌گیرد. در نهایت یک تگ div با کلاس bottom قرار می‌گیرد و با حرفهای i ، و f ، و l و p درون آن قسمتهای اطلاعات ، کادرجستجو، لیست بازشونده اندازه صفحه و نیز قسمت صفحه بندی قرار خواهد گرفت و در نهایت یک تگ div با کلاس clear قرار خواهد گرفت.

حرفهایی که در sDom معنی خاصی می‌دهند :
l سر حرف Length Changing برای لیست بازشونده مربوط به اندازه صفحه
f سر حرف Filtering input برای قسمت کادر جستجو
t سرحرف table برای جدول حاوی داده ها
i سر حرف information برای قسمت Showing x to xx from xxx entries
p سر حرف pagination برای قسمت صفحه بندی
r حرف دوم pRocessing برای قسمت پیغام قبل از بار کردن داده‌های جدول (قسمت loading)
H و F که مربوط به themeهای jQuery UI می‌شوند که بعدا درباره آنها توضیح داده می‌شود.

همچنین بین علامت‌های کوچکتر (<) و بزرگتر (>) یعنی اگر چیزی بیاید در یک تگ div قرار خواهد گرفت. اگر بخواهیم div ی بسازیم و به آن کلاس بدهیم از نحو زیر استفاده خواهیم کرد:

```
'<"class" and '>'
```

و اگر بخواهیم یک تگ div با یک id مشخص بسازیم از نحو زیر استفاده خواهیم کرد:

```
'<"#id" and '>'
```

در نهایت جدولی مثل جدول زیر تولید خواهد شد:

Showing 1 to 25 of 57 entries
Search:

Show بیست و پنج entries
First Previous 1 2 3 Next Last

موتور رندرگیری	مرورگر	پلتفرم (ها)	نسخه موتور	نمره css
Gecko	Firefox 1.0	+Win 98+ / OSX.2	1.7	A
Gecko	Firefox 1.5	+Win 98+ / OSX.2	1.8	A
Gecko	Firefox 2.0	+Win 98+ / OSX.2	1.8	A
Gecko	Firefox 3.0	+Win 2k+ / OSX.3	1.9	A
Gecko	Camino 1.0	+OSX.2	1.8	A
Gecko	Camino 1.5	+OSX.3	1.8	A
Gecko	Netscape 7.2	Win 95+ / Mac OS 8.6-9.2	1.7	A
Gecko	Netscape Browser 8	+Win 98SE	1.7	A
Gecko	Netscape Navigator 9	+Win 98+ / OSX.2	1.8	A
Gecko	Mozilla 1.0	+Win 95+ / OSX.1	1	A
Gecko	Mozilla 1.1	+Win 95+ / OSX.1	1.1	A
Gecko	Mozilla 1.2	+Win 95+ / OSX.1	1.2	A
Gecko	Mozilla 1.3	+Win 95+ / OSX.1	1.3	A
Gecko	Mozilla 1.4	+Win 95+ / OSX.1	1.4	A
Gecko	Mozilla 1.5	+Win 95+ / OSX.1	1.5	A
Gecko	Mozilla 1.6	+Win 95+ / OSX.1	1.6	A
Gecko	Mozilla 1.7	+Win 98+ / OSX.1	1.7	A
Gecko	Mozilla 1.8	+Win 98+ / OSX.1	1.8	A
Gecko	Seamonkey 1.1	+Win 98+ / OSX.2	1.8	A
Gecko	Epiphany 2.20	Gnome	1.8	A
KHTML	Konqureror 3.1	KDE 3.1	3.1	C
KHTML	Konqureror 3.3	KDE 3.3	3.3	A
KHTML	Konqureror 3.5	KDE 3.5	3.5	A
Misc	NetFront 3.1	Embedded devices	-	C
Misc	NetFront 3.4	Embedded devices	-	A
موتور رندرگیری	مرورگر	پلتفرم (ها)	نسخه موتور	نمره css

Showing 1 to 25 of 57 entries
Search:

Show بیست و پنج entries
First Previous 1 2 3 Next Last

شکل ج) جدول نهایی تولید شده توسط DataTables

کدهای نهایی این مثال را از DataTables-DoteNetTips-Tutorial-03.zip دریافت کنید.

نظرات خوانندگان

نویسنده: sorosh
تاریخ: ۱۳۹۲/۰۴/۰۷ ۷:۴۷

با سلام و عرض ادب

زمانیکه من با Jquery , Ajax سطرهای دیتای جدول مورد نظر برای DataTable شدن را از سمت سرور ایجاد می‌کنم متاسفانه بار اول دیتاها رو نشون میده ولی Search نمیکنه و صفحه بندی هم نمیکنه و ... در ضمن کدهای مربوطه رو هم می‌گذارم . لطفا راهنمایی کنید که اگه خواستیم دیتاها را از سمت سرور بیاریم و کار بده باید چه کار کرد؟ مرسی

```
$(document).ready(function () {
                dataparam2 = "cmd=FillScope";
                $.ajax({
                    url: "Default2.aspx",
                    type: "POST",
                    data: dataparam2,
                    async: true,
                    success: function (msg) {
                        if (msg != '') {
                            var data = eval("(" + msg + ")");
                            $("#tbodytblMain").html('');
                            for (var i = 0; i < data.length; i++) {
                                $("#tbodytblMain").append(
                                         "<tr class='odd gradeX'>"
                                         + "<td style='width:200px'>" + data[i].T + "</td>"
                                         + "<td style='width:150px'>" + data[i].P + "</td>"
                                          + "<td>" + data[i].S + "</td>"
                                         + "<td>" + data[i].TP + "</td>"
                                         + "<td>" + data[i].Sp + "</td>" + "</tr>");

                            }
                        }
                    },
                    error: function (msg) {

                    }
                });


                $('#tblMain').dataTable();
            });
```

کد سمت سرور:

```
if (Request["cmd"] == "FillScope")
        {
            string Val = "برخوار";
            JavaScriptSerializer js = new JavaScriptSerializer();
            string serText = "";
            MUIDataClassesDataContext db = new MUIDataClassesDataContext();
            var LST = (from x in db.tblProjectInfos
                       where x.tblScope.xScopeName.Contains(Val)
                       orderby x.tblScope.xScopeName
                       select new
                       {

                           P = x.xPlace,
                           S = x.tblScope.xScopeName,
                           TP = x.tblProjectType.xProjectTypeName,
                           Sp = x.tblStatus.xStatusName
                       });
            serText = js.Serialize(LST);
            Response.Write(serText);
            Response.End();

        }
```

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۴/۰۸ ۳:۱۱

سلام

رندر کردن جدول حاوی داده‌ها باید به data tables سپرده بشه. بدین صورت که داده‌های دریافتی از سرور به فرمت مناسبی تبدیل بشن و بعد به خصوصیت aaData نسبت داده بشن، البته به تبع اون و حتما باید خصوصیت aoColumns هم مقدار دهی بشه.

```
$(document).ready(function () {
      $.ajax({
          url: "Default.aspx/GetBrowsers",
          contentType: "application/json; charset=utf-8",
          dataType: "json",
          type: "POST",
          success: function (response) {
          if (response != "") {
                    var data = eval("(" + response.d + ")");
                    $('#browsers-grid').dataTable({
                            "aaData": data,
                            "bProcessing" : true,
                            "aoColumns": [
                                { "mData": "Engine" },
                                { "mData": "Name" },
                                { "mData": "Platform" },
                                { "mData": "Version", "sClass": "center" },
                                { "mData": "Grade", "sClass": "center" }
                            ]
                    });
                }
          },
      });
});
```

کدهای سمت سرور:

مثلا فرض کنید ذر سمت سرور بخواهید لیستی از مرورگرها رو برگشت بدین. کلاس زیر رو در نظر بگیرید:

```
public class Browser
{
    public int Id { get; set; }
    public string Engine { get; set; }
    public string Name { get; set; }
    public string Platform { get; set; }
    public float Version { get; set; }
    public string Grade { get; set; }
}
```

برای برگشت دادن لیستی از مرورگرها به طرف کلاینت، متدی مثل زیر خواهید داشت:

```
[WebMethod]
public static string GetBrowsers()
{
    List<Browser> browsers = new List<Browser>()
        {
            new Browser
                {
                    Id = 1,
                    Engine = "Trident",
                    Name = "Internet Explorer 4.0",
                    Platform = "Win95+",
                    Version = 4,
                    Grade = "X"
                },
            new Browser
                {
                    Id = 2,
                    Engine = "Trident",
                    Name = "Internet Explorer 5.0",
```

```
                    Platform = "Win95+",
                    Version = 5,
                    Grade = "C"
                },
        };
    return browsers.ToJson();
}
```

در متد بالا، لیستی از مرورگرها با استفاده از یک متد الحاقی تبدیل به فرمت json میشه و به طرف کاربر فرستاده میشه.

عنوان: **اعتبارسنجی سایتهای چند زبانه در ASP.NET MVC - قسمت اول**
نویسنده: محسن اسماعیل پور
تاریخ: ۱۳۹۲/۰۳/۰۸ ۰:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net MVC, jQuery, Validation, Localization, Globalization, MVC

اگر در حال تهیه یک سایت چند زبانه هستید و همچنین سری مقالات Globalization در ASP.NET MVC رو دنبال کرده باشید میدانید که با تغییر Culture فایلهای Resource مورد نظر بارگذاری و نوشته‌های سایت تغییر میابند ولی با تغییر Culture رفتار اعتبارسنجی در سمت سرور نیز تغییر و اعتبارسنجی بر اساس Culture فعلی سایت انجام میگیرد. بررسی این موضوع را با یک مثال شروع میکنیم.

یک پروژه وب بسازید سپس به پوشه Models یک کلاس با نام ValueModel اضافه کنید. تعریف کلاس به شکل زیر هست:

```
public class ValueModel
{
    [Required]
    [Display(Name = "Decimal Value")]
    public decimal DecimalValue { get; set; }

    [Required]
    [Display(Name = "Double Value")]
    public double DoubleValue { get; set; }

    [Required]
    [Display(Name = "Integer Value")]
    public int IntegerValue { get; set; }

    [Required]
    [Display(Name = "Date Value")]
    public DateTime DateValue { get; set; }
}
```

به سراغ کلاس HomeController بروید و کدهای زیر را اضافه کنید:

```
[HttpPost]
public ActionResult Index(ValueModel valueModel)
{
    if (ModelState.IsValid)
    {
        return Redirect("Index");
    }

    return View(valueModel);
}
```

Culture را به fa-IR تغییر میدهیم، برای اینکار در فایل web.config در بخش system.web کد زیر اضافه نمایید:

```
<globalization culture="fa-IR" uiCulture="fa-IR" />
```

و در نهایت به سراغ فایل Index.cshtml بروید کدهای زیر رو اضافه کنید:

```
@using (Html.BeginForm())
{
    <ol>
        <li>
            @Html.LabelFor(m => m.DecimalValue)
            @Html.TextBoxFor(m => m.DecimalValue)
            @Html.ValidationMessageFor(m => m.DecimalValue)
        </li>
```

```
        <li>
            @Html.LabelFor(m => m.DoubleValue)
            @Html.TextBoxFor(m => m.DoubleValue)
            @Html.ValidationMessageFor(m => m.DoubleValue)
        </li>
        <li>
            @Html.LabelFor(m => m.IntegerValue)
            @Html.TextBoxFor(m => m.IntegerValue)
            @Html.ValidationMessageFor(m => m.IntegerValue)
        </li>
        <li>
            @Html.LabelFor(m => m.DateValue)
            @Html.TextBoxFor(m => m.DateValue)
            @Html.ValidationMessageFor(m => m.DateValue)
        </li>
        <li>
            <input type="submit" value="Submit"/>
        </li>
    </ol>
}
```

پرژه را اجرا نمایید و در ۲ تکست باکس اول ۲ عدد اعشاری را و در ۲ تکست باکس آخر یک عدد صحیح و یک تاریخ وارد نمایید و سپس دکمه Submit را بزنید. پس از بازگشت صفحه از سمت سرور در در ۲ تکست باکس اول با این پیامها روبرو میشوید که مقادیر وارد شده نامعتبر میباشند.

اگر پروژه رو در حالت دیباگ اجرا کنیم و نگاهی به داخل ModelState بیاندازیم، میبینیم که کاراکتر جدا کننده قسمت اعشاری برای fa-IR '/' میباشد که در اینجا برای اعداد مورد نظر کاراکتر '.' وارد شده است.

برای فایق شدن بر این مشکل یا باید سمت سرور اقدام کرد یا در سمت کلاینت. در بخش اول راه حل سمت کلاینت را بررسی مینماییم.

در سمت کلاینت برای اینکه کاربر را مجبور به وارد کردن کاراکترهای مربوط به Culture فعلی سایت نماییم باید مقادیر وارد شده را اعتبارسنجی و در صورت معتبر نبودن مقادیر پیام مناسب نشان داده شود. برای اینکار از کتابخانه jQuery Globalize استفاده میکنیم. برای اضافه کردن jQuery Globalize از طریق کنسول nuget فرمان زیر اجرا نمایید:

```
PM> Install-Package jquery-globalize
```

پس از نصب کتابخانه اگر به پوشه Scripts نگاهی بیاندازید میبینید که پوشای با نام jquery.globalize اضافه شده است. درداخل پوشه زیر پوشی دیگری با نام cultures وجود دارد که در آن Culture های مختلف وجود دارد و بسته به نیاز میتوان از آنها استفاده کرد. دوباره به سراغ فایل Index.cshtm بروید و فایلهای جاوا اسکریپتی زیر را به صفحه اضافه کنید:

```
<script src="~/Scripts/jquery.validate.js"> </script>
<script src="~/Scripts/jquery.validate.unobtrusive.js"> </script>
<script src="~/Scripts/jquery.globalize/globalize.js"> </script>
<script src="~/Scripts/jquery.globalize/cultures/globalize.culture.fa-IR.js"> </script>
```

در فایل globalize.culture.fa-IR.js کاراکتر جدا کننده اعشاری '.' در نظر گرفته شده است که مجبور به تغییر آن هستیم. برای اینکار فایل را باز کرده و numberFormat را پیدا کنید و آن را به شکل زیر تغییر دهید:

```
numberFormat: {
    pattern: ["n-"],
    ".": "/",
    currency: {
        pattern: ["$n-", "$ n"],
        ".": "/",
        symbol: "ریال"
    }
},
```

و در نهایت کدهای زیر را به فایل Index.cshtml اضافه کنید و برنامه را دوباره اجرا نمایید :

```
Globalize.culture('fa-IR');
$.validator.methods.number = function(value, element) {
    if (value.indexOf('.') > 0) {
        return false;
    }
    var splitedValue = value.split('/');
    if (splitedValue.length === 1) {
        return !isNaN(Globalize.parseInt(value));
    } else if (splitedValue.length === 2 && $.trim(splitedValue[1]).length === 0) {
        return false;
    }
    return !isNaN(Globalize.parseFloat(value));
};
};
```

در خط اول Culture را ست مینمایم و در ادامه نحوه اعتبارسنجی را در unobtrusive validation تغییر میدهیم . از آنجایی که برای اعتبارسنجی عدد وارد شده از تابع parseFloat استفاده میشود، کاراکتر جدا کننده قسمت اعشاری قابل قبول برای این تابع '.' است پس در داخل تابع دوباره '/' به '.' تبدیل میشود و سپس اعتبارسنجی انجام میشود از اینرو اگر کاربر '.' را نیز وارد نماید قابل قبول است به همین دلیل با این خط کد (if (value.indexOf('.') > 0 وجود نقطه را بررسی میکنیم تا در صورت وجود '.' پیغام خطا نشان داده شود.در خط بعدی بررسی مینماییم که اگر عدد وارد شده اعشاری نباشد از تابع parseInt استفاده نماییم. در خط بعدی این حالت را بررسی مینماییم که اگر کاربر عددی همچون ۱۲/ وارد کرد پیغام خطا صادر شود.

برای اعتبارسنجی تاریخ شمسی متاسفانه توابع کمکی برای تبدیل تاریخ در فایل globalize.culture.fa-IR.js وجود ندارد ولی اگر نگاهی به فایلهای Culture عربی بیاندازید همه دارای توابع کمکی برای تبدیل تاریخ هجری به میلادی هستند به همین دلیل امکان اعتبارسنجی تاریخ شمسی با استفاده از jQuery Globalize میسر نمیباشد. من خودم تعدادی توابع کمکی را به globalize.culture.fa-IR.js اضافه کردام که از تقویم فارسی آقای علی فرهادی برداشت شده است و با آنها کار اعتبارسنجی را انجام میدهیم. لازم به ذکر است این روش ۱۰۰% تست نشده است و شاید راه کاملا اصولی نباشد ولی به هر حال در اینجا توضیح میدهم. در فایل globalize.culture.fa-IR.js قسمت Gregorian_Localized را پیدا کنید و آن را با کدهای زیر جایگزین کنید:

```
Gregorian_Localized: {
    firstDay: 6,
    days: {
        names: ["شنبه", "جمعه", "پنجشنبه", "چهارشنبه", "سه شنبه", "دوشنبه", "یکشنبه"],
        namesAbbr: ["شنبه", "جمعه", "پنجشنبه", "چهارشنبه", "سه شنبه", "دوشنبه", "یکشنبه"],
        namesShort: ["ش", "ج", "پ", "چ", "س", "د", "ی"]
    },
    months: {
        names: ["ژانویه", "فوریه", "مارس", "آوریل", "می", "ژوئن", "ژوئیه", "اوت", "سپتامبر", "اکتبر", "نوامبر", "دسامبر", "", ""],
        namesAbbr: ["ژانویه", "فوریه", "مارس", "آوریل", "می", "ژوئن", "ژوئیه", "اوت", "سپتامبر", "اکتبر", "نوامبر", "دسامبر", "", ""]
    },
    AM: ["ق.ظ", "ق.ظ", "ق.ظ"],
    PM: ["ب.ظ", "ب.ظ", "ب.ظ"],
    patterns: {
        d: "yyyy/MM/dd",
        D: "yyyy/MM/dd",
        t: "hh:mm tt",
        T: "hh:mm:ss tt",
        f: "yyyy/MM/dd hh:mm tt",
        F: "yyyy/MM/dd hh:mm:ss tt",
        M: "dd MMMM"
    },
    JalaliDate: {
        g_days_in_month: [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31],
        j_days_in_month: [31, 31, 31, 31, 31, 31, 30, 30, 30, 30, 30, 29]
    },
    gregorianToJalali: function (gY, gM, gD) {
        gY = parseInt(gY);
        gM = parseInt(gM);
        gD = parseInt(gD);
        var gy = gY - 1600;
        var gm = gM - 1;
        var gd = gD - 1;

        var gDayNo = 365 * gy + parseInt((gy + 3) / 4) - parseInt((gy + 99) / 100) + parseInt((gy + 399) / 400);

        for (var i = 0; i < gm; ++i)
            gDayNo += Globalize.culture().calendars.Gregorian_Localized.JalaliDate.g_days_in_month[i];
        if (gm > 1 && ((gy % 4 == 0 && gy % 100 != 0) || (gy % 400 == 0)))
            /* leap and after Feb */
            ++gDayNo;
        gDayNo += gd;

        var jDayNo = gDayNo - 79;

        var jNp = parseInt(jDayNo / 12053);
        jDayNo %= 12053;

        var jy = 979 + 33 * jNp + 4 * parseInt(jDayNo / 1461);

        jDayNo %= 1461;

        if (jDayNo >= 366) {
            jy += parseInt((jDayNo - 1) / 365);
            jDayNo = (jDayNo - 1) % 365;
        }

        for (var i = 0; i < 11 && jDayNo >= Globalize.culture().calendars.Gregorian_Localized.JalaliDate.j_days_in_month[i]; ++i) {
            jDayNo -= Globalize.culture().calendars.Gregorian_Localized.JalaliDate.j_days_in_month[i];
        }
        var jm = i + 1;
        var jd = jDayNo + 1;

        return [jy, jm, jd];
    },
    jalaliToGregorian: function (jY, jM, jD) {
```

```
        jY = parseInt(jY);
        jM = parseInt(jM);
        jD = parseInt(jD);
        var jy = jY - 979;
        var jm = jM - 1;
        var jd = jD - 1;

        var jDayNo = 365 * jy + parseInt(jy / 33) * 8 + parseInt((jy % 33 + 3) / 4);
        for (var i = 0; i < jm; ++i) jDayNo +=
Globalize.culture().calendars.Gregorian_Localized.JalaliDate.j_days_in_month[i];

        jDayNo += jd;

        var gDayNo = jDayNo + 79;

        var gy = 1600 + 400 * parseInt(gDayNo / 146097); /* 146097 = 365*400 + 400/4 - 400/100 +
400/400 */
        gDayNo = gDayNo % 146097;

        var leap = true;
        if (gDayNo >= 36525) /* 36525 = 365*100 + 100/4 */ {
            gDayNo--;
            gy += 100 * parseInt(gDayNo / 36524); /* 36524 = 365*100 + 100/4 - 100/100 */
            gDayNo = gDayNo % 36524;

            if (gDayNo >= 365)
                gDayNo++;
            else
                leap = false;
        }

        gy += 4 * parseInt(gDayNo / 1461); /* 1461 = 365*4 + 4/4 */
        gDayNo %= 1461;

        if (gDayNo >= 366) {
            leap = false;

            gDayNo--;
            gy += parseInt(gDayNo / 365);
            gDayNo = gDayNo % 365;
        }

        for (var i = 0; gDayNo >=
Globalize.culture().calendars.Gregorian_Localized.JalaliDate.g_days_in_month[i] + (i == 1 && leap) ;
i++)
            gDayNo -= Globalize.culture().calendars.Gregorian_Localized.JalaliDate.g_days_in_month[i] +
(i == 1 && leap);
        var gm = i + 1;
        var gd = gDayNo + 1;

        return [gy, gm, gd];
    },
    checkDate: function (jY, jM, jD) {
        return !(jY < 0 || jY > 32767 || jM < 1 || jM > 12 || jD < 1 || jD >
            (Globalize.culture().calendars.Gregorian_Localized.JalaliDate.j_days_in_month[jM - 1] + (jM
== 12 && !((jY - 979) % 33 % 4))));
    },
    convert: function (value, format) {
        var day, month, year;

        var formatParts = format.split('/');
        var dateParts = value.split('/');
        if (formatParts.length !== 3 || dateParts.length !== 3) {
            return false;
        }

        for (var j = 0; j < formatParts.length; j++) {
            var currentFormat = formatParts[j];
            var currentDate = dateParts[j];
            switch (currentFormat) {
                case 'dd':
                    if (currentDate.length === 2 || currentDate.length === 1) {
                        day = currentDate;
                    } else {
                        year = currentDate;
                    }
                    break;
                case 'MM':
                    month = currentDate;
                    break;
                case 'yyyy':
```

```
                    if (currentDate.length === 4) {
                        year = currentDate;
                    } else {
                        day = currentDate;
                    }
                    break;
                default:
                    return false;
            }
        }

        year = parseInt(year);
        month = parseInt(month);
        day = parseInt(day);
        var isValidDate = Globalize.culture().calendars.Gregorian_Localized.checkDate(year, month,
day);
        if (!isValidDate) {
            return false;
        }

        var grDate = Globalize.culture().calendars.Gregorian_Localized.jalaliToGregorian(year, month,
day);
        var shDate = Globalize.culture().calendars.Gregorian_Localized.gregorianToJalali(grDate[0],
grDate[1], grDate[2]);

        if (year === shDate[0] && month === shDate[1] && day === shDate[2]) {
            return true;
        }

        return false;
    }
},
```

روال کار در تابع convert به اینصورت است که ابتدا تاریخ وارد شده را بررسی مینماید تا معتبر بودن آن معلوم شود به عنوان مثال اگر تاریخی مثل 31/12/1392 وارد شده باشد و در ادامه برای بررسی بیشتر تاریخ یک بار به میلادی و تاریخ میلادی دوباره به شمسی تبدیل میشود و با تاریخ وارد شده مقایسه میشود و در صورت برابری تاریخ معتبر اعلام میشود. در فایل Index.cshtml کدهای زیر اضافی نمایید:

```
$.validator.methods.date = function (value, element) {
    return Globalize.culture().calendars.Gregorian_Localized.convert(value, 'yyyy/MM/dd');
};
```

برای اعتبارسنجی تاریخ میتوانید از ۲ فرمت استفاده کنید:

۱ - yyyy/MM/dd

۲ - dd/MM/yyyy

البته از توابع اعتبارسنجی تاریخ میتوانید به صورت جدا استفاده نمایید و لزومی ندارد آنها را همراه با jQuery Globalize بکار ببرید. در آخر خروجی کار به این شکل است:

Decimal Value

1/3

Double Value

1.3 The field Double Value must be a number.

Integer Value

11/ The field Integer Value must be a number.

Date Value

1392/12/30 The field Date Value must be a date.

Submit

در کل استفاده از jQuery Globalize برای اعتبارسنجی در سایتهای چند زبانه به نسبت خوب میباشد و برای هر زبان میتوانید از culture مورد نظر استفاده نمایید. در قسمت دوم این مطلب به بررسی بخش سمت سرور میپردازیم.

عنوان: **Image Annotations**
نویسنده: سجـــــاد ف
تاریخ: ۱۳۹۲/۰۳/۱۷ ۰:۱۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, jQuery, jQuery Ajax, Data Annotation, jQuery-Plugins

Add Note

می‌خواهیم با تغییر jQuery Image Annotation این پلاگین و برای asp.net استفاده کنیم

ایجاد دیتابیس

ابتدا یک دیتابیس به نام Coordinates ایجاد کنید و سپس جدول زیر رو ایجاد کنید

```
USE [Coordinates]
GO
CREATE TABLE [dbo].[Coords2](
[top] [int] NULL,
[left] [int] NULL,
[width] [int] NULL,
[height] [int] NULL,
[text] [nvarchar](50) NULL,
[id] [uniqueidentifier] NULL,
[editable] [bit] NULL
) ON [PRIMARY]
GO
```

ایجاد کلاس Coords برای خواندن و ذخیره اطلاعات

```
public class Coords
{
    public string top;
    public string left;
    public string width;
    public string height;
    public string text;
    public string id;
    public string editable;

    public Coords(string top, string left, string width, string height, string text, string id, string
editable)
    {
        this.top = top;
        this.left = left;
        this.width = width;
        this.height = height;
        this.text = text;
        this.id = id;
        this.editable = editable;


    }
}
```

فرم اصلی برنامه شامل 3 وب سرویس به شرح زیر می‌باشد

1-GetDynamicContext

این متد در زمان لود اطلاعات از دیتابیس استفاده می‌شود (وقتی که postback صورت می‌گیرد)

```
[WebMethod]
    public static List<Coords> GetDynamicContext(string entryId, string entryName)
    {
        List<Coords> CoordsList = new List<Coords>();

        string connect = "Connection String";
        using (SqlConnection conn = new SqlConnection(connect))
        {
            string query = "SELECT [top], [left], width, height, text, id, editable FROM  Coords2";
            using (SqlCommand cmd = new SqlCommand(query, conn))
            {
                conn.Open();
                using (SqlDataReader reader=cmd.ExecuteReader())
                {
                    while (reader.Read())
                    {
                        CoordsList.Add(new Coords(reader["top"].ToString(), reader["left"].ToString(),
                                                  reader["width"].ToString(),
reader["height"].ToString(),
                                                  reader["text"].ToString(), reader["id"].ToString(),
                                                  reader["editable"].ToString()));
                    }
                }

               conn.Close();
            }
        }

         return CoordsList;


    }
```

2,3 -SaveCoordsو DeleteCoords

این دو متد هم واسه ذخیره و حذف می‌باشند که نکته خاصی ندارند و خودتون بهینه اش کنید(در فایل ضمیمه موجودند)

تغییر فایل jquery.annotate.js جهت فراخوانی وب سرویس ها

فقط لازمه که سه قسمت زیر رو در فایل اصلی تغییر بدید

```
$.fn.annotateImage.ajaxLoad = function (image) {
        ///<summary>
        ///Loads the annotations from the "getUrl" property passed in on the
        ///     options object.
        ///</summary>

        $.ajax({
            type: "POST",
            contentType: "application/json; charset=utf-8",
            url: "Default.aspx/GetDynamicContext",
            data: "{'entryId': '" + 1 + "','entryName': '" + 2 + "'}",
            dataType: "json",
            success: function (msg) {
                image.notes = msg.d;
                $.fn.annotateImage.load(image);
            }
        });



 };
```

```
$.fn.SaveCoords = function (note) {
        $.ajax({
            type: "POST",
            contentType: "application/json; charset=utf-8",
            url: "Default.aspx/SaveCoords",
            data: "{'top': '" + note.top + "','left': '" + note.left + "','width': '" + note.width +
"','height': '" + note.height + "','text': '" + note.text + "','id': '" + note.id + "','editable': '" +
note.editable + "'}",
            dataType: "json",
            success: function (msg) {
                note.id = msg.d;
             }
        });



 };



 $.fn.annotateView.prototype.edit = function () {

        ///<summary>

        ///Edits the annotation.

        ///</summary>

        if (this.image.mode == 'view') {

            this.image.mode = 'edit';

            var annotation = this;

            // Create/prepare the editable note elements

            var editable = new $.fn.annotateEdit(this.image, this.note);

            $.fn.annotateImage.createSaveButton(editable, this.image, annotation);

            // Add the delete button

            var del = $('<a>حذف</a>');

            del.click(function () {

                var form = $('#image-annotate-edit-form form');

                $.fn.annotateImage.appendPosition(form, editable)

                if (annotation.image.useAjax) {
```

```
                $.ajax({
                    type: "POST",
                    contentType: "application/json; charset=utf-8",
                    url: "Default.aspx/DeleteCoords",
                    //                        url: annotation.image.deleteUrl,
                    //                        data: form.serialize(),
                    data: "{'id': '" + editable.note.id + "'}",
                    dataType: "json",
                    success: function (msg) {
                        //                image.notes = msg.d;
                        //                $.fn.annotateImage.load(image);
                    },
                    error: function (e) { alert("An error occured deleting that note.") }
                });
            }
            annotation.image.mode = 'view';
            editable.destroy();
            annotation.destroy();
        });
        editable.form.append(del);
        $.fn.annotateImage.createCancelButton(editable, this.image);
    }
};
```

این پروژه شامل یه سری فایل css هم هست که می‌تونید کل پروژه رو از اینجا دانلود کنید

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۷ ۰:۲۲

ضمن تشکر، فقط نکته استفاده از JSON.stringify در حین کار با jQuery Ajax رو بهتره اعمال کنید تا در دراز مدت و حالت‌های مختلف ورودی به مشکل برنخورید. به صورت خلاصه اطلاعات ارسالی رو جمع نزنید و تبدیل به رشته نکنید. یک شیء کامل درست کنید و اجازه بدید JSON.stringify اون رو تبدیل کنه.

نویسنده: یزدان
تاریخ: ۱۳۹۲/۰۳/۱۸ ۱۰:۳۵

اگر متدهای وب سرویس رو درون صفحاتی قرار بدیم که نیاز به لاگین کردن و احراز هویت جهت دسترسی به آنها باشد ، آیا میتوان متدهای وب سرویس رو خارج از اون صفحات فراخوانی نمود ؟
چه تضمینی وجود دارد که در خارج از صفحات با سطح دسترسی (حداقل لاگین) وب سرویسها فراخوانی نشوند ؟
به نظر شما MS AJAX کارش به پایان رسیده ؟
تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۸ ۱۰:۵۴

- اگر دقت کرده باشید در کدهای فوق این متدها استاتیک تعریف شدن، یعنی مراحل چرخه طول عمر یک صفحه به آن‌ها اعمال نشده و اصلا جزئی از مباحث اعتبارسنجی صفحه جاری لحاظ نخواهند شد.
- در وب فرم‌ها استفاده از وب متدها یک روش برای کار با jQuery Ajax است. روش دوم استفاده از Generic handlerها و فایل‌های ashx است. در این موارد به علت استاتیک نبودن handlerهای تولیدی، می‌شود همه نوع اعتبارسنجی رو اعمال کرد اعم از روش Forms Authentication مثلا توسط context.Request.IsAuthenticated یا حتی روش منسوخ شده استفاده از سشن‌ها برای اعتبارسنجی با پیاده سازی IRequiresSessionState.
- در مطلب فوق اصلا از MS Ajax استفاده نشده. اون هم جایگاه خودش رو در کاربردهای خاص خودش دارد.

عنوان: **آموزش (jQuery) جی کوئری #6**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۴/۰۲ ۲۱:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, jQuery, Web Design, Learning jQuery

در ادامه مطلب قبلی آموزش (jQuery) جی کوئری #5 به ادامه بحث می‌پردازیم.
در پست‌های قبلی مروری بر jQuery داشته و در چند پست انواع روش‌های انتخاب عناصر صفحه وب را توسط jQuery بررسی کردیم. در پست‌های آینده با مباحث پیشرفته‌تری همچون انجام عملیاتی روی المانهای انتخاب شده، خواهیم پرداخت؛ امید است مفید واقع شود.

۲ -۲ - ایجاد عناصر HTML جدید

گاهی اوقات نیاز می‌شود که یک یا چند عنصر جدید به صفحه‌ی در حال اجرا اضافه شوند. این حالت می‌تواند به سادگی قرار گرفتن یک متن در جایی از صفحه و یا به پیچیدگی ایجاد و نمایش یک جدول حاوی اطلاعات دریافت شده از بانک اطلاعاتی باشد.
ایجاد عناصر به صورت پویا در یک صفحه در حال اجرا کار ساده ای برای jQuery می‌باشد، زیرا همانطور که در پست آموزش (jQuery) جی کوئری #1 مشاهده کردیم $() با دریافت دستور ساخت یک عنصر HTML آن را در هر زمان ایجاد می‌کند، دستور زیر :

```
$("<div>Hello</div>")
```

یک عنصر div ایجاد می‌کند و آماده افزودن آن به صفحه در هر زمان می‌باشد.تمامی توابع و متدهایی را که تاکنون بررسی کردیم قابل اعمال بروی اینگونه اشیا نیز می‌باشند. شاید در ابتدا ایجاد عناصر به این شکل خیلی مفید به نظر نرسد، اما زمانی که بخواهیم کارهای حرفه ای‌تری انجام دهیم؛ برای مثال کار با AJAX، خواهیم دید که تا چه اندازه ایجاد عناصر به این روش می‌تواند مفید باشد.
دقت کنید که یک راه کوتاهتر نیز برای ایجاد یک عنصر خالی وجود دارد که به شکل زیر است:

```
$("<div>")
//  همه اینها معادل هستند

$("<div></div>")

$("<div/>")
```

اما برای ایجاد عناصری که خود می‌توانند حاوی عناصر دیگر باشند استفاده از راههای کوتاه توصیه نمی‌شود مانند نوشتن تگ <script>. اما راههای زیادی برای انجام اینکار وجود دارد.
برای اینکه مزه اینکار را بچشید بد نیست نگاهی به مثال زیر بیندازید (نگران قسمت‌های نامفهوم نباشید به مرور با آنها آشنا خواهیم شد):

```
$("<div class='foo'>I have foo!</div><div>I don't</div>")
     .filter(".foo").click(function() {
          alert("I'm foo!");
      }).end().appendTo("#someParentDiv");
```

در این مثال ابتدا ما یک المان div ایجاد کردیم که دارای کلاس foo می‌باشد، و خود شامل یک div دیگر است. در ادامه div که دارای کلاس foo بوده را انتخاب کرده و رویداد کلیک را به آن بایند کردیم. و در انتها این div را با محتویاتش به المانی با Id=someParentDiv در سلسله مراتب DOM اضافه می‌کند.
برای اجرا این کد می‌توانید کد آن را دانلود کرده و فایل chapter2/new.divs.htmlرا اجرا کنید خروجی مانند تصویر زیر خواهد بود:

جهت تکمیل مطلب فعلی یک مثال کاملتر از این سایت جهت بررسی انتخاب کردم:

```
$( "<div/>", {
```

```
    "class": "test",
     text: "Click me!",
     click: function() {
           $( this ).toggleClass( "test" );
     }
   }).appendTo( "body" );
```

در این مثال کمی پیشرفته‌تر یک div ایجاد شده کلاس test را برای آن قرار داده و عنوان ان را برابر text قرار میدهد و یک رویداد کلیک برای آن تعریف می‌کند و در نهایت آن را به body سایت اضافه می‌کند.

با توجه به اینکه مطالب بعدی طولانی بوده و تقریبا مبحث جدایی است؛ در پست بعدی به بررسی توابع و متدهای **مدیریت مجموعه انتخاب شده** خواهیم پرداخت.

عنوان: آموزش (jQuery) جی کوئری #7
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۴/۰۴ ۲۱:۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Web Design, Learning jQuery

پس از انواع روش‌های انتخاب عناصر در jQuery اکنون زمان آشنایی با متدها و توابعی جهت پردازش مجموعه انتخاب شده رسیده است.

۲-۳- مدیریت مجموعه انتخاب شده

هز زمان که مجموعه ای از عناصر انتخاب انتخاب می‌شوند، خواه این عناصر از طریق انتخاب کننده‌ها انتخاب شده باشند و یا تابع $() در صدد ایجاد آن باشد، مجموعه ای در اختیار داریم که آماده دستکاری و اعمال تغییر با استفاده از متدهای jQuery می‌باشد. این متدها را در پست‌های آتی بررسی خواهیم کرد. اما اکنون به این نکته می‌پردازیم که اگر بخواهیم از همین مجموعه انتخاب شده زیر مجموعه ای ایجاد کنیم و یا حتی آن را گسترش دهیم، چه باید کرد؟ به طور کلی در این پست پیرامون این مورد بحث خواهد شد که چگونه می‌توانیم مجموعه انتخاب شده را به آن صورت که می‌خواهیم بهیود دهیم.
برای درک مطالبی که قصد توضیح آنها را در این قسمت داریم، یک صفحه کارگاهی دیگر نیز در فایل قابل دانلود این کتاب موجود می‌باشد که با نام chapter2/lab.wrapped.set.html قابل دسترسی می‌باشد. نکته مهم در مورد این صفحه کارگاهی آن است که می‌بایست عبارات و دستورهای کامل را با ساختار صحیح وارد کنیم در غیر این اینصورت این صفحه کاربردی نخواهد داشت.

۲-۳-۱-تعیین اندازه یک مجموعه عناصر

قبلا اشاره کردیم که مجموعه عناصر jQuery شباهت هایی با آرایه دارد. یکی از این شباهت‌ها داشتن ویژگی length می‌باشد که مانند آراه در جاوااسکریپت ، تعداد عناصر موجود در مجموعه را شامل می‌شود.
افزون بر این ویژگی، jQueryیک متد را نیز معرفی کرده است که دقیقا شبیه به length عمل می‌کند. این متد size() می‌باشد که استفاده از آن را در مثال زیر مشاهده می‌کنید.

```
$('#someDiv')
      .html('There are '+$('a').size()+' link(s) on this page.');
```

این مثال تمام لینک‌های موجود در صفحه را شناسایی می‌کند و سپس بااستفاده از متد size() تعداد آنها را بر می‌گرداند. در واقع یک رشته ایجاد می‌شود و در یک عنصر با شناسه someDiv قرار داده می‌شود. متد html در پست‌های آتی بررسی می‌شود. فرم کلی متد size() را در زیر مشاهده می‌کنید.

size()

تعداد عناصر موجود در مجموعه را محاسبه می‌کند

پارامترها

بدون پارامتر

خروجی

تعداد عناصر مجموعه

اکنون که تعداد عناصر مجموعه را می‌دانیم چگونه می‌توانیم به هریک از آنها دسترسی مستقیم داشته باشیم؟

۲-۳-۲-بکارگیری عنصرهای مجموعه

به طور معمول پس از انتخاب یک مجموعه با استفاده از متدهای jQuery، عملی را بروی آن عناصر انتخاب شده انجام می‌دهیم، مانند مخفی کردن آنها با متد hide() ، اما گاهی اوقات می‌خواهیم بروی یک یا چند مورد خاص از عناصر انتخاب شده عملی را اعمال کنیم. jQuery چند روش مختلف را به منظور اینکار ارایه می‌دهد.

از آنجا که مجموعه عناصر انتخاب شده در jQuery مانند آرایه در جاوااسکریپت می‌باشد، بنابراین به سادگی می‌توانیم از اندیس برای دستیابی به عناصر مختلف مجموعه استفاده کنیم. برای مثال به منظور دسترسی به اولین عکس از مجموعه عکس‌های انتخاب

شده که دارای صفت alt می‌باشند از دستور زیر استفاده می‌کنیم:

```
$('img[alt]')[0]
```

اما اگر ترجیح می‌دهید به جای اندیس از یک متد استفاده کنید، jQuery متد get() را در نظر گرفته است:

get(index)
برای واکشی یک یا تمام عناصر موجود در مجموعه استفاده می‌شود. اگر برای این متد پارامتری ارسال نشود، تمام عناصر را در قالب یک آرایه جاوااسکریپت بر می‌گرداند، اما در صورت ارسال یک پارامتر، تنها آن عنصر را بر می‌گرداند.
پارامتر
شماره اندیس یک عنصر که می‌بایست یک مقدار عددی باشد.
خروجی
یک یا آرایه ای از عناصر
دستور زیر مانند دستور قبلی عمل می‌کند:

```
$('img[alt]').get(0)
```

متد get() می‌تواند برای بدست آوردن یک ارایه از عناصر پیچیده نیز استفاده شود. مثلا:

```
var allLabeledButtons = $('label+button').get();
```

خروجی دستور بالا لیست تمام buttonهای موجود در صفحه است که بعد از عنصر label قرار گرفته اند، در نهایت این آرایه در متغیری به نام allLabeledButtons قرار خواهد گرفت.

در متد get() دیدیم که با دریافت شماره اندیس یک عنصر، آن عنصر را برای ما برمی گرداند، عکس این عمل نیز امکان پذیر می‌باشد. فرض کنید می‌خواهیم از میان تمام عناصر عکس، شماره اندیس عکسی با شناسه findMe را بدست آوریم. برای این منظور می‌توانیم از کد زیر بهره ببریم:

```
var n = $('img').index($('img#findMe')[0]);
```

فرم کلی متد index() به صورت زیر است:

index(element)
عنصر ارسالی را در مجموعه عناصر پیدا می‌کند، سپس شماره اندیس ان را بر می‌گرداند. اگر چنین عنصری در مجموعه یافت نشد خروجی 1- خواهد بود.
پارامتر
پارامتر این متد می‌تواند یک عنصر و یا یک انتخاب کننده باشد که خروجی انتخاب کننده نیز در نهایت یک عنصر خواهد بود.
خروجی
شماره اندیس عنصر در مجموعه

۲-۳-۳-برش و کوچک کردن مجموعه ها
ممکن است شرایطی پیش آید که پس از بدست آوردن یک مجموعه عناصر انتخاب شده نیاز باشد که عنصری به آن مجموعه اضافه و یا حتی عنصری را از آن حذف کنیم تا در نهایت مجموعه ای باب میل ما بدست آید. برای انجام چنین تغییرهایی در یک مجموعه jQuery کلکسیون بزرگی از متدها را برای ما به همراه دارد. اولین موردی که به آن می‌پردازیم، افزودن یک عنصر به مجموعه می‌باشد.

اضافه کردن عناصر بیشتر به یک مجموعه عنصر انتخاب شده

همواره ممکن است شرایطی پیش آید که پس از ایجاد یک مجموعه عناصر انتخاب شده، بخواهیم عنصری را به آن اضافه کنیم. یکی از دلایلی که باعث می‌شود این امر در jQuery بیشتر مورد نیاز باشد توانایی استفاده از متدهای زنجیره ای در jQuery است. ابتدا یک مثال ساده را بررسی می‌کنیم. فرض کنید می‌خواهیم تمام عناصر عکس که دارای یکی از دو خصوصیت alt و یا title می‌باشند را انتخاب کنیم، با استفاده از انتخاب کننده‌های قدرتمند jQuery دستوری مانند زیر خواهیم نوشت:

```
$('img[alt],img[title]')
```

اما برای آنکه با متد add() که به منظور افزودن عنصر به مجموعه عناصر می‌باشد آشنا شوید این مثال را به صورت زیر می‌نویسیم:

```
$('img[alt]').add('img[title]')
```

استفاده از متد add() به این شکل موجب می‌شود تا بتوانیم مجموعه‌های مختلف را به یکدیگر متصل کنیم و یک مجموعه کلی‌تر از عناصر انتخاب شده ایجاد کنیم. متد add() در این حالت مانند متد end() عمل می‌کند که در قسمت ۲-۳-۶ شرح داده خواهد شد.

ساختار کلی متد add() به صورت زیر است:

add(expression)

ابتدا یک کپی از مجموعه انتخاب شده ایجاد می‌کند، سپس با افزودن محتویات پارامتر expression به آن نمونه، یک مجموعه جدید تشکیل می‌دهد. پارامتر expression می‌تواند حاوی یک انتخاب کننده، قطعه کد HTML، یک عنصر و یا آرایه ای از عناصر باشد.

پارامتر

در این پارامتر مواردی (مانند رشته، آرایه، المان) که می‌خواهیم به مجموعه عناصر انتخاب شده اضافه شوند قرار می‌گیرد. که می‌تواند انتخاب کننده، قطعه کد HTML، یک عنصر و یا ارایه ای از عناصر باشد.

خروجی

یک کپی از مجموعه اصلی به علاوه موارد اضافه شده.

اصلاح عناصر یک مجموعه عنصر انتخاب شده

در قسمت قبل دیدیم که چگونه با استفاده از متد add() و با بکار گیری آن در توابع زنجیره ای، توانستیم عناصری جدید به مجموعه انتخاب شده اضافه کنیم. عکس این عمل را نیز می‌توان با ستفاده از متد not() در توابع زنجیره ای انجام داد. این متد عملکرد شبیه به فیلتر :not دارد، اما با این تفاوت که بکار گیری آن مانند متد add() می‌باشد و می‌توان در هر جایی از زنجیره از آن استفاده کرد تا عناصر مورد نظر را از مجموعه انتخاب شده حذف کنیم.

فرض کنید می‌خواهیم تمامی عناصر عکسی را که دارای خصوصیت title می‌باشند به استثنای آن موردی که واژه puppy در مقدار مربوط به این صفت استفاده کرده اند را انتخاب کنیم. این کار به سادگی و با استفاده از دستوری مانند(]"img[title]:not([title*="puppy می‌توان انجام داد. اما برای آن که مثالی از چگونگی کار متد not() ببینید، این کار را به شکل زیر انجام می‌دهیم:

```
$('img[title]').not('[title*=puppy]')
```

این دستور تمام عکس‌های دارای خصوصیت title را به استثنا titleهایی که مقدار puppy در آنها وجود دارد را انتخاب می‌کند.

شکل کلی متد not() مانند زیر است:

not(expression)

ابتدا یک کپی از مجموعه انتخاب شده ایجاد می‌کند، سپس از آن کپی عناصری را که expression مشخص می‌کند را حذف می‌نماید.

پارامتر

این پارامتر تعیین کننده عناصر در نظر گرفته شده برای حذف می‌باشد. این پارامتر می‌تواند یک عنصر، ارایه ای از عناصر، انتخاب کننده و یا یک تابع باشد.

اگر این پارامتر تابع باشد، تک تک عناصر مجموعه به آن ارسال می‌شوند و هر یک که خروجی تابع را برابر با مقدار true کند، حذف می‌شود.

خروجی

یک کپی از مجموعه اصلی بدون موارد حذف شده.

این شیوه برای ایجاد مجموعه هایی که انتخاب کننده‌ها قادر به ساخت آن‌ها نمی‌باشند، کاربرد بسیار مناسبی دارد، زیرا از تکنیک‌های برنامه نویسی استفاده می‌کند و دست ما را برای اعمال انتخاب‌های گوناگون باز می‌کند.

اگر در شرایطی خاص با حالتی روبرو شدید که احساس کردید عکس این انتخاب برای شما کارایی دارد، باز می‌توانید از یکی دیگر از متدهای jQuery استفاده کنید، متد filter() عملکردی مشابه با متد not() دارد با این تفاوت که عناصری از مجموعه حذف می‌شوند که خروجی تابع را false کنند.

فرض کنید می‌خواهیم تمام عناصر td که دارای یک عنصر عددی می‌باشند را انتخاب کنیم. با وجود قدرت فوق العاده انتخاب کننده‌های jQuery به ما ارایه می‌دهند، انجام چنین کاری با استفاده از انتخاب کننده‌ها غیر ممکن است. در این حالت از متد filter() را به شکل زیر استفاده می‌کنیم:

```
$('td').filter(function(){return this.innerHTML.match(/^\d+$/)})
```

دستور فوق یک مجموعه از تمام عناصر td انتخاب می‌کند، سپس تک تک عناصر مجموعه انتخاب شده را به تابعی که پارامتر متد filter() می‌باشد، ارسال می‌کند. این تابع با استفاده از عبارت منظم مقدار عنصر کنونی را می‌سنجد. اگر این مقدار یک یا زنجیره ای از ارقام بود، خروجی تابع true خواهد بود، و ان عنصر از مجموعه حذف نمی‌شود، اما اگر این مقدار عددی نبود، خروجی تابع false بوده و عنصر از مجموعه کنار گذاشته می‌شود.

شکل کلی متد filter() به شکل زیر است.

filter(expression)

ابتدا یک کپی از مجموعه انتخاب شده ایجاد می‌کند، سپس از آن کپی عناصری را که expression مشخص می‌کند را حذف می‌نماید.

پارامتر

این پارامتر تعیین کننده عناصر در نظر گرفته شده برای حذف می‌باشد. این پارامتر می‌تواند یک عنصر، ارایه ای از عناصر، انتخاب کننده و یا یک تابع باشد.

اگر این پارامتر تابع باشد، تک تک عناصر مجموعه به آن ارسال می‌شوند و هر یک که خروجی تابع را برابر با مقدار false کند، حذف می‌شود.

خروجی

یک کپی از مجموعه اصلی بدون عناصر حذف شده.

ایجاد یک زیر مجموعه از مجموعه عناصر انتخاب شده

گاهی اوقات داشتن یک زیر مجموعه از عناصر یک مجموعه، چیزی است که دنبال آن هستیم. برای این منظور jQuery متد slice() را ارایه می‌کند که عناصر را بر اساس جایگاه آن‌ها به زیر مجموعه هایی کوچکتر تقسیم می‌کند. نتیجه استفاده از این متد یک مجموعه جدید برگرفته از تعدادی عناصر پشت سر هم،از یک مجموعه انتخاب شده خواهد بود:

شکل کلی متد slice() مانند زیر است:

slice(begin, end)

ایجاد و برگرداندن یک مجموعه جدید از بخشی از عناصر پشت سر هم در یک مجموعه اصلی.

پارامتر

begin: پارامتر begin که یک پارامتر عددی می‌باشد و مقدار اولیه آن از صفر آغاز می‌شود، نشان دهنده اولین عنصری است که می‌خواهیم در مجموعه جدید حضور داشته باشد.

end: پارامتر دوم که آن هم یک پارامتر عددی می‌باشد و از صفر آغاز می‌شود، در این متد اختیاری است. این پارامتر اولین عنصری است که نمی‌خواهیم از آن به بعد در مجموعه جدید حضور داشته باشد را مشخص می‌کند. اگر مقداری برای این پارامتر ننویسیم، به صورت پیش فرض تا انتهای مجموعه انتخاب می‌شود.

خروجی

یک مجموعه عنصر جدید.

اگر بخواهیم از یک مجموعه کلی، تنها یک عنصر را در قالب یک مجموعه انتخاب کنیم می‌توانیم از متد slice() استفاده کنیم و مکان آن عنصر در مجموعه را به آن ارسال کنیم. دستور زیر مثالی از این حالت می‌باشد:

```
$('*').slice(2,3);
```

این مثال ابتدا تمام عناصر موجود در صفحه را انتخاب می‌کند، سپس سومین عنصر از آن مجموعه را در یک مجموعه جدید باز می‌گرداند. دقت کنید که دستور فوق با دستور $('*').get(2) کاملا متفاوت است، چرا که خروجی این دستور تنها یک عنصر است، در حالی که خروجی دستور فوق یک مجموعه است.

از همین رو دستور زیر باعث ایجاد یک مجموعه که شامل چهار عنصر اولیه صفحه می‌باشد، می‌شود.

```
$('*').slice(0,4);
```

برای ایجاد یک مجموعه از عناصر انتهایی موجود در صفحه نیز می‌توان از دستوری مانند زیر استفاده کرد:

```
$('*').slice(4);
```

این دستور تمام عناصر موجود در صفحه را انتخاب می‌کند، سپس مجموعه ای جدید می‌سازد که تمام عناصر به استثنای چهار عنصر اول را در خود جای می‌دهد.

۲-۳-٤-ایجاد مجموعه بر اساس روابط

jQuery به ما این توانایی را داده است تا مجموعه هایی را انتخاب کنیم، که اساس انتخاب عناصر، رابطه سلسله مراتبی آنها با عناصر HTML صفحه باشد. اکثر این متدها یک پارامتر اختیاری از نوع انتخاب کننده دریافت می‌کنند که می‌تواند برای انتخاب عناصر مجموعه استفاده شود. در صورتی که چنین پارامتری ارسال نگردد، تمام عناصر واجد شرایط متد در مجموعه انتخاب می‌شوند.

جدول ۲-٤-متدهای موجود برای ایجاد مجموعه‌های جدید بر اساس روابط

متد	توضیح
children ()	مجموعه ای را برمی گرداند که شامل تمام فرزندان بدون تکرار از عناصر مجموعه می‌باشد.
contents ()	مجموعه ای شامل محتویات تمام عناصر برمی گرداند. (از این متد معمولا برای عناصر iframe استفاده می‌شود)
next ()	مجموعه ای شامل فرزندان پدرش که بعد از خود این عنصر می‌باشند را برمی گرداند. این مجموعه عنصر تکراری ندارد.
nextAll ()	مجموعه ای شامل تمام فرزندان پدرش که بعد از خود این عنصر می‌باشند را بر می‌گرداند.
parent ()	مجموعه ای شامل نزدیک‌ترین پدر اولین عنصر مجموعه را بر می‌گرداند.
parents ()	مجموعه ای شامل تمام پدران مستقیم عناصر مجموعه را بر می‌گرداند. این مجموعه عنصر تکراری ندارد.
prev ()	مجموعه ای شامل فرزندان پدرش که قبل از خود این عنصر می‌باشند را برمی گرداند. این مجموعه عنصر تکراری ندارد.

توضیح	متد
مجموعه ای شامل تمام فرزندان پدرش که قبل از خود این عنصر می‌باشند را بر می‌گرداند.	prevAll ()
مجموعه ای بدون عنصر تکراری را بر می‌گرداند که شامل تمام فرزندان پدر خود عنصر خواهد بود.	siblings ()

تمامی جدول بالا غیر از متد contents() پارامتری از نوع رشته که انتخاب کننده برای متد می‌باشند، استفاده می‌کند.

۲-۳-۵-استفاده از مجموعه‌های انتخاب شده برای انتخاب عناصر

با وجود اینکه تاکنون با شمار زیادی از توانایی‌های انتخاب و انتخاب کننده‌ها در jQuery آشنا شده اید، هنوز چند مورد دیگر نیز برای افزایش قدرت انتخاب باقی مانده است.

متد find() بروی یک مجموعه عناصر انتخاب شده به کار گرفته می‌شود و یک پارامتر ورودی نیز دارد. این پارامتر که یک انتخاب کننده است تنها بروی فرزندان این مجموعه اعمال می‌شود. برای مثال فرض کنید یک مجموعه از عناصر انتخاب و در متغیر wrapperSet قرار گرفته است. با دستور زیر می‌توانیم تمام عناصر (تگ) cite را که درون یک تگ p قرار گرفته اند را انتخاب کنیم، به شرطی که آن‌ها فرزندان عناصر مجموعه wrapperSet باشند:

```
wrappedSet.find('p cite')
```

البته می‌توانیم این تکه کد را به صورت زیر هم بنویسیم:

```
$('p cite', wrapperSet)
```

مانند سایر متدهای معرفی شده قدرت اصلی این متد نیز هنگام استفاده در متدهای زنجیره ای مشخص می‌شود.

شکل کلی متد find() مانند زیر است:

find(selector)

یک مجموعه عنصر جدید ایجاد می‌کند که شامل فرزندان عناصر مجموعه قبل می‌شود.

پارامتر

یک انتخاب کننده است که در قالب یک رشته به این متد ارسال می‌شود.

خروجی

یک مجموعه عنصر جدید

جهت پیدا کردن عناصری که داخل یک wrapperSet می‌توانیم از متد دیگری به نام contains() نیز استفاده کنیم. این متد مجموعه ای را بر می‌گرداند که شامل تمام عناصری است که در انتخاب کننده پارامتر ورودی است. مثلا

```
$('p').contains('Lorem ipsum')
```

این دستور تمامی عناصر p را که شامل *Lorem ipsum* است را بر می‌گرداند. قالب کلی متد مانند زیر است:

contains(text)

مجموعه ای از عناصر که شامل متن ورودی می‌باشند را بر می‌گرداند.

پارامتر

رشته ورودی که می‌خواهیم در عنصر فراخوان متد جستجو شود.

خروجی

مجموعه ای از عناصر از نوع فراخوان متد را بر می‌گرداند که شامل متن ورودی باشد.

آخرین متدی که به بررسی آن می‌پردازیم متد is() می‌باشد. با استفاده از این متد می‌توانیم اطمینان حاصل کنیم که دست کم یک

عنصر از مجموعه عناصر، شرایط مشخص شده توسط ما را دارا باشد. یک انتخاب کننده به این متد ارسال می‌شود، اگر عنصری از مجموعه عناصر انتخاب شد، خروجی متد true می‌شود و در غیر این صورت مقدار false بر گردانده خواهد شد. برای مثال:

```
var hasImage = $('*').is('img');
```

در صورت وجود دست کم یک عنصر عکس در کل عناصر صفحه، دستور بالا مقدار متغیر hasImage را برابر true قرار می‌دهد.
قالب کلی متد is() مانند زیر است:

is(selector)
بررسی می‌کند که آیا عنصری در مجموعه وجود دارد که انتخاب کننده ارسالی آن را انتخاب کند؟
پارامتر
یک انتخاب کننده است که در قالب یک رشته به این متد ارسال می‌شود.
خروجی
مقدار true در صورت وجود دست کم یک عنصر و false در صورت عدم وجود توسط تابع برگردانده می‌شود.

۲-۳-۶-مدیریت زنجیره‌های jQuery
تاکنون در مورد استفاده از متدها و توابع زنجیره ای زیاد بحث کرده ایم و انجام چندین عمل در یک دستور را به عنوان یک قابلیت بزرگ معرفی کرده ایم و البته از آن هم استفاده کردیم و در ادامه نیز استفاده خواهیم کرد. به کار گیری متدها به صورت زنجیره ای نه تنها موجب نوشتن کدهای قدرتمند و قوی به صورت مختصر و خلاصه می‌شود، بلکه از لحاظ کارایی نیز نکته مثبتی محسوب می‌شود، زیرا برای اعمال هر متد نیازی به محاسبه و انتخاب مجدد مجموعه نخواهد بود.
بنابراین متدهای مختلفی که در زنجیره استفاده می‌کنیم، برخی از آنها ممکن است مجموعه‌های جدیدی تولید کنند. برای مثال استفاده از متد clone() موجب می‌شود تا مجموعه ای جدید از کپی عناصر در مجموعه اول ایجاد شود. زمانی که یکی از متدهای زنجیره یک مجموعه جدید را تولید می‌کند، دیگر راهی برای استفاده از مجموعه پیشین در زنجیره نخواهیم داشت و این نکته زنجیره ما را به خطر می‌اندازد. عبارت زیر را در نظر بگیرید:

```
$('img').clone().appendTo('#somewhere');
```

این مثال دو مجموعه ایجاد می‌کندو نخست مجموعه ای شامل تمام عناصر عکس صفحه ایجاد می‌شود و مجموعه دوم کپی مجموعه اول است که به انتهای عنصری با شناسه somewhere اضافه می‌شود. حال اگر بخواهیم پس از اعمال کپی بروی مجموعه اصلی عملی مانند افزودن یک کلاس را بروی آن انجام دهیم چه باید بکنیم؟ همچنین نمی‌توانیم مجموعه اصلی را به انتهای زنجیره انتقال دهیم، چون بروی قسمتی دیگر اثر خواهد گذاشت.
برای مرتفع کردن چنین نیازی، jQuery متد end() را معرفی کرده است. زمانی از این متد استفاده می‌شود، یک نسخه پشتیبان از مجموعه کنونی ایجاد می‌شود . همان مجموعه برگردانده می‌شود. بنابراین اگر متدی پس از آن ظاهر شودف اثرش بروی مجموعه اولیه خواهد بود. مثال زیر را در نظر بگیرید:

```
$('img').clone().appendTo('#somewhere').end().addClass('beenCloned');
```

این مثال دو مجموعه ایجاد می‌کندو نخست مجموعه ای شامل تمام عناصر عکس صفحه ایجاد می‌شود و مجموعه دوم کپی مجموعه اول است که به انتهای عنصری با شناسه somewhere اضافه می‌شود. اما با استفاده از متد end() همان مجموعه اولیه در ادامه زنجیره قرار خواهد گرفت و سپس متد addClass() بروی تمامی عناصر عکس اعمال می‌شود، نه تنها عکس‌های موجود در مجموعه اول، اگر از متد end() استفاده نشود متد addClass() بروی عناصر مجموعه دوم اعمال خواهد شد.
قالب کلی متد end() به شکل زیر است:

end()
در متدهای زنجیره ای استفاده می‌شود و از مجموعه کنونی یک پشتیبان می‌گیرد تا همان مجموعه در زنجیره جریان داشته باشد.
پارامتر

ندارد

خروجی

مجموعه عنصر قبلی

شاید در نظر گرفتن مجموعه‌ها در متدهای زنجیره ای به شکل یک پشته به درک بهتر از متد end() کمک کند. هر زمان که یک مجموعه جدید در زنجیره ایجاد می‌شود، آن مجموعه به بالای پشته افزوده می‌شود، اما با فراخوانی متد end() ، بالاترین مجموعه از این پشته برداشته می‌شود و مجدادا مجموعه پیشین در زنجیره قرار می‌گیرد.

متد دیگری که توانایی ایجاد تغییر در این پشته خیالی را دارد، متد andSelf() می‌باشد. این متد دو مجموعه بالای پشته را با یکدیگر ادغام می‌کند و آن‌ها را به یک مجموعه تبدیل می‌کند.

شکل کلی متد andSelf() به صورت زیر است:

andSelf()

دو مجموعه پیشین در یک زنجیره را با یکدیگر ادغام می‌کند.

پارامتر

ندارد

خروجی

مجموعه عنصری ادغام شده

در مباحث بعدی **کار با صفت‌ها و ویژگی‌های عناصر** بحث خواهد شد.

موفق و موید باشید

نظرات خوانندگان

نویسنده: منصور جعفری
تاریخ: ۱۳۹۳/۰۱/۰۳ ۱۶:۲۲

سلام
مثلا در مورد طراحی یک سایت که اطلاعاتی بصورت تکراری پشت سر هم تکرار میشن (مثلا کامنت‌های که برای یک موضوع ارسال میشن) چطور باید باید اطلاعات مثلا مربوط به یک فیلد رو دستکاری انجام بدیم
برای مثال

```
@foreach(var item in Model)
{
   <td class="text-right itemfarsi">@item.Farsi</td>
}
```

چطور میشه مثلا همین تیبل دیتا رو برای هر کامنت باتوجه به متن اون تغییر داد
من با استفاده از کدهای زیر دستور خودم رو انجام میدم اما در مورد تمام مطالب فقط اطلاعات مربوط به قسمت اول رو برمیگردونه.

```
$(document).ready(function () {
        var content = $(".itemfarsi").text();
        if (content.length >= 50) {
            var mycont = content.substring(0, 50);
            $(".itemfarsi").html(mycont);
        } else {
            $(".itemfarsi").html(content);
        }
    });
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۰۳ ۱۷:۲

از متد each می‌شود استفاده کرد.

عنوان: آموزش QUnit #1
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۲/۰۴/۱۷ ۱۹:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, jQuery, QUnit, Unit testing

مقدمه: تست و آزمایش کد برنامه‌ها و وب سایت‌هایمان، بهترین راه کاهش خطا و مشکلات آنها بعد از انتشار است. از جمله روش‌های موجود، تست واحد است که ویژوال استادیو نیز از آن برای پروژه‌های دات نت پشتیبانی می‌کند. با افزایش روز افزون کتابخانه‌های جاوا اسکریپتی و جی کوئری، نیاز به تست کدهای جاواسکریپتی نیز بیشتر به نظر می‌رسد و بهتر است تست واحد و آزمایش شوند. اما برخلاف کدهای C# و ASP.NET تست کدهای جاوا اسکریپت، مخصوصا زمانی که به دستکاری عناصر DOM می‌پردازیم و یا رویدادهای درون صفحه وب را با استفاده از جی کوئری می‌نویسیم، حتی اگر در فایل جداگانه‌ای نوشته شود، این بدان معنی نیست که آماده تست واحد است و ممکن است امکان نوشتن تست وجود نداشته باشد.
بنابراین چه چیزی یک تست واحد است؟ در بهترین حالت توابعی که مقداری را برمی گردانند، بهترین حالت برای تست واحد است. اما در بیشتر موارد شما نیاز دارید تا تاثیر کد را بر روی عناصر صفحه نیز مشاهده نمایید.

ساخت تست واحد

برای تست پذیری بهتر، توابع جاوا اسکریپت و هر کد دیگری، آن را می‌بایست طوری بنویسید که مقادیر تاثیر گذار در اجرای تابع به عنوان ورودی تابع در نظر گرفته شده باشند و همیشه نتیجه به عنوان خروجی تابع برگردانده شود؛ قطعه کد زیر را در نظر بگیرید:

```
function prettyDate(time){
    var date = new Date(time || ""),
      diff = (((new Date()).getTime() - date.getTime()) / 1000),
      day_diff = Math.floor(diff / 86400);

    if ( isNaN(day_diff) || day_diff < 0 || day_diff >= 31 )
      return;

    return day_diff == 0 && (
        diff < 60 && "just now" ||
        diff < 120 && "1 minute ago" ||
        diff < 3600 && Math.floor( diff / 60 ) +
          " minutes ago" ||
        diff < 7200 && "1 hour ago" ||
        diff < 86400 && Math.floor( diff / 3600 ) +
          " hours ago") ||
      day_diff == 1 && "Yesterday" ||
      day_diff < 7 && day_diff + " days ago" ||
      day_diff < 31 && Math.ceil( day_diff / 7 ) +
        " weeks ago";
  }
```

تابع pertyDate اختلاف زمان حال را نسبت به زمان ورودی، بصورت یک رشته برمی گرداند. اما در اینجا مقدار زمان حال، در خط سوم، در خود تابع ایجاد شده است و در صورتی که بخواهیم برای چندین مقدار آن را تست کنیم زمان حال متفاوتی در نظر گرفته می‌شود و حداکثر، زمان 31 روز قبل را نمایش داده و در بقیه تاریخ ها undefined را بر می‌گرداند. برای تست واحد، چند تغییر می‌دهیم.

بهینه سازی، مرحله اول:

پارامتری به عنوان مقدار زمان جاری برای تابع در نظر می‌گیریم و تابع را جدا کرده و در یک فایل جداگانه قرار می‌دهیم. فایل prettydate.js بصورت زیر خواهد شد.

```
function prettyDate(now, time){
  var date = new Date(time || ""),
    diff = (((new Date(now)).getTime() - date.getTime()) / 1000),
    day_diff = Math.floor(diff / 86400);

  if ( isNaN(day_diff) || day_diff < 0 || day_diff >= 31 )
    return;

  return day_diff == 0 && (
      diff < 60 && "just now" ||
      diff < 120 && "1 minute ago" ||
      diff < 3600 && Math.floor( diff / 60 ) +
```

```
        " minutes ago" ||
      diff < 7200 && "1 hour ago" ||
      diff < 86400 && Math.floor( diff / 3600 ) +
        " hours ago") ||
    day_diff == 1 && "Yesterday" ||
    day_diff < 7 && day_diff + " days ago" ||
    day_diff < 31 && Math.ceil( day_diff / 7 ) +
      " weeks ago";
}
```

حال یک تابع برای تست داریم، چند تست واحد واقعی می‌نویسیم

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Refactored date examples</title>
  <script src="prettydate.js"></script>
  <script>
  function test(then, expected) {
    results.total++;
    var result = prettyDate("2013/01/28 22:25:00", then);
    if (result !== expected) {
      results.bad++;
      console.log("Expected " + expected +
        ", but was " + result);
    }
  }
  var results = {
    total: 0,
    bad: 0
  };
  test("2013/01/28 22:24:30", "just now");
  test("2013/01/28 22:23:30", "1 minute ago");
  test("2013/01/28 21:23:30", "1 hour ago");
  test("2013/01/27 22:23:30", "Yesterday");
  test("2013/01/26 22:23:30", "2 days ago");
  test("2012/01/26 22:23:30", undefined);
  console.log("Of " + results.total + " tests, " +
    results.bad + " failed, " +
    (results.total - results.bad) + " passed.");
  </script>
</head>
<body>

</body>
</html>
```

در کد بالا یک تابع بدون استفاده از Qunit برای تست واحد نوشته ایم که با آن تابع prettyDate را تست می‌کند. تابع test مقدار زمان حال و رشته خروجی را گرفته و آن را با تابع اصلی تست می‌کند در آخر تعداد تست ها، تست‌های شکست خورده و تست های پاس شده گزارش داده می‌شود.

خروجی می‌تواند مانند زیر باشد:

.Of 6 tests, 0 failed, 6 passed

.Expected 2 day ago, but was 2 days ago

.f 6 tests, 1 failed, 5 passed

عنوان: آموزش QUnit #2
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۲/۰۴/۲۱ ۱۲:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, Unit testing, jQuery, QUnit

فریم ورک تست جاوا اسکریپت QUnit: انتخاب و استفاده از یک فریم ورک برای تست کدهای جاوا اسکریپت، قطعا نتیجه بهتری را به همراه خواهد داشت. من در این جا از QUnit که یکی از بهترین‌های تست واحد است، استفاده می‌کنم. برای این کار فایل‌های qunit.js و qunit.css را دانلود و مانند زیر برای تست واحد آماده کنید:

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Refactored date examples</title>

  <link rel="stylesheet" href="../qunit.css">
  <script src="../qunit.js"></script>
  <script src="prettydate.js"></script>

  <script>
  test("prettydate basics", function() {
    var now = "2013/01/28 22:25:00";
    equal(prettyDate(now, "2013/01/28 22:24:30"), "just now");
    equal(prettyDate(now, "201308/01/28 22:23:30"), "1 minute ago");
    equal(prettyDate(now, "2013/01/28 21:23:30"), "1 hour ago");
    equal(prettyDate(now, "2013/01/27 22:23:30"), "Yesterday");
    equal(prettyDate(now, "2013/01/26 22:23:30"), "2 days ago");
    equal(prettyDate(now, "2012/01/26 22:23:30"), undefined);
  });
  </script>
</head>
<body>

<div id="qunit"></div>

</body>
</html>
```

در کد بالا ابتدا فایل‌های فریم ورک و فایل prettydate.js را اضافه کردیم. برای نمایش نتیجه تست، یک تگ div با نام qunit در بین تگ body اضافه می‌کنیم.

تابع test:

این تابع برای تست توابع نوشته شده، استفاده می‌شود. ورودی‌های این تابع، یکی عنوان تست و دومی یک متود دیگر، به عنوان ورودی دریافت می‌کند که در آن بدنه تست نوشته می‌شود.

تابع equal:

اولین تابع برای سنجش تست واحد equal است و در آن، تابعی که می‌خواهیم تست کنیم با مقدار خروجی آن مقایسه می‌شود. فایل را با نام test.htm ذخیره و آن را در مرورگر خود باز نمایید. خروجی در شکل آورده شده است:

همین طور که در تصویر بزرگ می‌بینید اطلاعات مرورگر، زمان تکمیل تست و تعداد تست، تعداد تست پاس شده و تعداد تست شکست خورده، نشان داده شده است.

اگر یکی از تست‌ها با شکست روبرو شود رنگ پس زمینه قرمز و جزئیات شکست نمایش داده می‌شوند.

بهینه سازی، مرحله اول:

در حال حاضر تست ما کامل نیست زیرا امکان تست n weeks ago یا تعداد هفته پیش میسر نیست. قبل از آنکه این را به آزمون اضافه کنیم، تغییراتی در تست می‌دهیم

```
test("prettydate basics", function() {
  function date(then, expected) {
    equal(prettyDate("2013/01/28 22:25:00", then), expected);
  }
  date("2013/01/28 22:24:30", "just now");
  date("2013/01/28 22:23:30", "1 minute ago");
  date("2013/01/28 21:23:30", "1 hour ago");
  date("2013/01/27 22:23:30", "Yesterday");
  date("2013/01/26 22:23:30", "2 days ago");
  date("2012/01/26 22:23:30", undefined);
});
```

تابع prettyDate را در تابع دیگری به نام date قرار می‌دهیم. این تغییر سبب می‌شود تا امکان مقایسه زمان ورودی تست جاری با تست قبلی فراهم شود.

تست دستکاری عناصر DOM:

تا اینجا با تست توابع آشنا شدید، حالا می‌خواهیم تغییراتی در prettyDate دهیم تا امکان انتخاب عناصر DOM و به روزرسانی آن نیز وجود داشته باشد. فایل prettyDate2.js در زیر آورده شده است:

```
var prettyDate = {
  format: function(now, time){
    var date = new Date(time || ""),
      diff = (((new Date(now)).getTime() - date.getTime()) / 1000),
      day_diff = Math.floor(diff / 86400);

    if ( isNaN(day_diff) || day_diff < 0 || day_diff >= 31 )
      return;

    return day_diff === 0 && (
        diff < 60 && "just now" ||
        diff < 120 && "1 minute ago" ||
        diff < 3600 && Math.floor( diff / 60 ) +
          " minutes ago" ||
        diff < 7200 && "1 hour ago" ||
        diff < 86400 && Math.floor( diff / 3600 ) +
          " hours ago") ||
      day_diff === 1 && "Yesterday" ||
      day_diff < 7 && day_diff + " days ago" ||
      day_diff < 31 && Math.ceil( day_diff / 7 ) +
        " weeks ago";
  },

  update: function(now) {
    var links = document.getElementsByTagName("a");
    for ( var i = 0; i < links.length; i++ ) {
      if ( links[i].title ) {
        var date = prettyDate.format(now, links[i].title);
        if ( date ) {
          links[i].innerHTML = date;
        }
      }
    }
  }
};
```

prettyDate شامل دو تابع، یکی format که weeks ago به آن اضافه گردیده و تابع update که با انتخاب تگ‌ها، مقدار title را به تابع فرمت و خروجی آن را در Html هر عنصر قرار می‌دهد. حال یک تست واحد می‌نویسیم:

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Refactored date examples</title>
  <link rel="stylesheet" href="../qunit.css">
  <script src="../qunit.js"></script>
  <script src="prettydate2.js"></script>
  <script>
  test("prettydate.format", function() {
    function date(then, expected) {
      equal(prettyDate.format("2013/01/28 22:25:00", then),
        expected);
    }
    date("2013/01/28 22:24:30", "just now");
    date("2013/01/28 22:23:30", "1 minute ago");
    date("2013/01/28 21:23:30", "1 hour ago");
    date("2013/01/27 22:23:30", "Yesterday");
    date("2013/01/26 22:23:30", "2 days ago");
    date("2012/01/26 22:23:30", undefined);
  });

  function domtest(name, now, first, second) {
    test(name, function() {
      var links = document.getElementById("qunit-fixture")
        .getElementsByTagName("a");
      equal(links[0].innerHTML, "January 28th, 2013");
      equal(links[2].innerHTML, "January 27th, 2013");
      prettyDate.update(now);
      equal(links[0].innerHTML, first);
      equal(links[2].innerHTML, second);
    });
```

```
  }
  domtest("prettyDate.update", "2013-01-28T22:25:00Z",
    "2 hours ago", "Yesterday");
  domtest("prettyDate.update, one day later", "2013/01/29 22:25:00",
    "Yesterday", "2 days ago");
  </script>
</head>
<body>

<div id="qunit"></div>
<div id="qunit-fixture">

<ul>
  <li id="post57">
    <p>blah blah blah...</p>
    <small>
      Posted <span>
        <a href="/2013/01/blah/57/" title="2013-01-28T20:24:17Z"
          >January 28th, 2013</a>
      </span>
      by <span><a href=""></a></span>
    </small>
  </li>
  <li id="post57">
    <p>blah blah blah...</p>
    <small>
      Posted <span>
        <a href="/2013/01/blah/57/" title="2013-01-27T22:24:17Z"
          >January 27th, 2013</a>
      </span>
      by <span><a href=""></a></span>
    </small>
  </li>
</ul>

</div>

</body>
</html>
```

همین طور که مشاهده می‌کنید در تست واحد اول خود تابع prettyDate.format را تست نموده ایم. در تست بعدی عناصر DOM نیز دستکاری و تست شده است. تابع domtest با جستجوی تگ qunit-fixture و تگ‌های a درون آن، مقدار نهایی html آن با مقدار داده شده، مقایسه شده است.

Refactored date examples ■ noglobals ■ notrycatch

☐ Hide passed tests

Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/28.0.1500.71 Safari/537.36

Tests completed in 268 milliseconds.
12 tests of 14 passed, 2 failed.

1. prettydate.format (0, 6, 6) Rerun

2. prettyDate.update (0, 4, 4) Rerun

3. prettyDate.update, one day later (2, 2, 4) Rerun

1. okay

2. okay

3. failed
Expected: "Yesterday"
Result: "22 hours ago"
Diff: "Yesterday" "22 hours ago"
Source: at Object.<anonymous>

4. failed
Expected: "2 days ago"
Result: "Yesterday"
Diff: "2 days ago" "Yesterday"
Source: at Object.<anonymous>

در شکل بالا نتیجه تست واحد نشان داده شده است.

عنوان: آموزش QUnit #3
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۲/۰۴/۲۴ ۱۷:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, Unit testing, jQuery, QUnit

در قسمت‌های قبلی با مفهوم تست واحد و کتابخانه quint آشنا شدید و مثالی را نیز با هم بررسی کردیم. در ادامه به قابلیت‌های بیشتر این کتابخانه می‌پردازیم.

توابع اعلان نتایج:

qunit سه تابع را جهت اعلان نتایج تست واحد فراهم نموده است

تابع ok:

تابع پایه‌ای تست واحد، دو پارامتر را به عنوان ورودی دریافت می‌کند و در صورتیکه بررسی نتیجه پارامتر اول برابر true باشد، تست با موفقیت روبرو شده است. پارامتر دوم برای نمایش یک پیام است. در مثال زیر حالت‌های مختلف آن بررسی شده است. مقادیر true، non-empty string به معنی موفقیت و مقادیر null،"",NaN،0،false و undefined به معنی شکست تست می‌باشد. در واقع خروجی تابع ارسالی به اعلان ok یکی از نتایج بالا می‌تواند باشد.

```
//ok( truthy [, message ] )

test( "ok test", function() {
  ok( true, "true succeeds" );
  ok( "non-empty", "non-empty string succeeds" );

  ok( false, "false fails" );
  ok( 0, "0 fails" );
  ok( NaN, "NaN fails" );
  ok( "", "empty string fails" );
  ok( null, "null fails" );
  ok( undefined, "undefined fails" );
});
```

تابع equal:

این اعلان یک مقایسه ساده بین پارامتر اول و دوم تابع می‌باشد که شرط برابری(==) را بررسی می‌نماید. وقتی مقدار اول و دوم برابر باشند، اعلان موفقیت و در غیر این صورت، تست با شکست رویرو شده و هر دو پارامتر نمایش داده می‌شوند.

```
//equal( actual, expected [, message ] )

test( "equal test", function() {
  equal( 0, 0, "Zero; equal succeeds" );
  equal( "", 0, "Empty, Zero; equal succeeds" );
  equal( "", "", "Empty, Empty; equal succeeds" );
  equal( 0, 0, "Zero, Zero; equal succeeds" );

  equal( "three", 3, "Three, 3; equal fails" );
  equal( null, false, "null, false; equal fails" );
});
```

زمانی که می‌خواهید مؤکداً شرط === را بررسی نمایید از strictEqual() استفاده کنید.

تابع deepEqual:

تکمیل شده دو تابع قبل می‌باشد و حتی امکان مقایسه دو شی را نیز با هم دارا است. علاوه بر این، امکان مقایسه NaN، تاریخ، عبارات باقاعده، آرایه‌ها و توابع نیز وجود دارند.

```
//deepEqual( actual, expected [, message ] )

test( "deepEqual test", function() {
  var obj = { foo: "bar" };
```

```
  deepEqual( obj, { foo: "bar" }, "Two objects can be the same in value" );
});
```

در صورتیکه نمی‌خواهید محتوای دو مقدار را با هم مقایسه کنید، از equal استفاده نمایید اما عموما deepEqual انتخاب بهتری می‌باشد.

تست عملیات کاربر:

گاهی لازم است رویدادهایی که از عملیات کاربران صدا زده می‌شوند تست شوند. در این موارد با صدا زدن تابع trigger جی‌کوئری، تابع مورد نظر را تست نمایید. به مثال زیر توجه نمایید:

```
function KeyLogger( target ) {
  if ( !(this instanceof KeyLogger) ) {
    return new KeyLogger( target );
  }
  this.target = target;
  this.log = [];

  var self = this;

  this.target.off( "keydown" ).on( "keydown", function( event ) {
    self.log.push( event.keyCode );
  });
}
```

این مثال یک گزارش دهنده است و در صورتیکه کاربر، کلیدی را فشار دهد، کد آن را گزارش می‌دهد و در آرایه log ذخیره می‌نماید. حال لازم است بصورت دستی این رویداد را صدا زده و تایع را تست کنیم. تست را بصورت زیر می‌نویسیم:

```
test( "keylogger api behavior", function() {

  var event,
      $doc = $( document ),
      keys = KeyLogger( $doc );

  // trigger event
  event = $.Event( "keydown" );
  event.keyCode = 9;
  $doc.trigger( event );

  // verify expected behavior
  equal( keys.log.length, 1, "a key was logged" );
  equal( keys.log[ 0 ], 9, "correct key was logged" );

});
```

برای این کار تابع KeyLogger را با شی document جی کوئری صدا زدیم و نتیجه را در متغییر keys قرار داده‌ایم. بعد رویداد keydown را با کد 9 پرکرده تایع trigger متغییر $doc را با مقدار event صدا زده‌ایم که در واقع بصورت دستی، یک رویداد اتفاق افتاده است. در آخر هم با اعلان equal تست واحد را انجام داده‌ایم.

عنوان: پلاگین DataTables کتابخانه jQuery - قسمت چهارم
نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۴/۲۵ ۷:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net MVC, jQuery, DataTables, jQuery-Plugins

همان طور که قبلا اشاره کردیم، این پلاگین می‌تواند از یک زبان برنامه نویسی سمت سرور داده‌های مورد نیاز خودش را دریافت کند. می‌توانید داده‌ها را با استفاده از AJAX و به صورت JSON از سرور دریافت کرده و با استفاده از DataTables آنها را در جدول تزریق کنید. در این قسمت سعی خواهیم کرد تا با استفاده از jQuery DataTables یک گرید را در MVC ایجاد کنیم. البته برای حذف جزئیات داده‌ها به جای این که از یک بانک اطلاعاتی دریافت شوند، در حافظه ساخته می‌شوند. در هر صورت اساس کار یکی است.

قصد داریم تا مانند مثال قسمت قبل، مجموعه ای از اطلاعات مربوط به مرورگرهای مختلف را در یک جدول نشان دهیم، اما این بار منبع داده ما فرق می‌کند. منبع داده از طرف سرور فراهم می‌شود. هر مرورگر - همان طور که در قسمت قبل مشاهده نمودید - شامل اطلاعات زیر خواهد بود:

موتور رندرگیری (Engine)

نام مرورگر (Name)

پلتفرم (Platform)

نسخه موتور (Version)

نمره سی اس اس (Grade)

به همین دلیل در سمت سرور، کلاسی خواهیم ساخت که نمایانگر یک مرورگر باشد. بدین صورت:

```
public class Browser
{
    public int Id { get; set; }
    public string Engine { get; set; }
    public string Name { get; set; }
    public string Platform { get; set; }
    public float Version { get; set; }
    public string Grade { get; set; }
}
```

استفاده از روش server side processing برای دریافت داده‌ها از سرور

این روش، یکی از امکانات jQuery DataTables است که با استفاده از آن، کلاینت تنها یک مصرف کننده صرف خواهد بود و وظیفه پردازش اطلاعات - یعنی تعداد رکوردهایی که برگشت داده می‌شود، صفحه بندی، مرتب سازی، جستجو، و غیره - به عهده سرور خواهد بود.

برای به کار گیری این روش، اولین کار این است که ویژگی bServerSide را true کنیم، مثلا بدین صورت:

```
var $table = $('#browsers-grid');
$table.dataTable({
      "bServerSide": true,
      "sAjaxSource": "/Home/GetBrowsers"
 });
```

همچنین ویژگی sAjaxSource را به Url ی که باید داده‌ها از آن دریافت شوند مقداردهی می‌کنیم.

به صورت پیش فرض مقدار ویژگی bServerSide مقدار false است؛ که یعنی منبع داده این پلاگین از سمت سرور خوانده نشود. اگر true باشد منبع داده و خیلی اطلاعات دیگر مربوط به داده‌های درون جدول باید از سرور به مرورگر کاربر پس فرستاده شوند. با true کردن مقدار bServerSide، آنگاه DataTables اطلاعاتی را راجع به شماره صفحه جاری، اندازه هر صفحه، شروط فیلتر کردن داده ها، مرتب سازی ستون ها، و غیره را به سرور می‌فرستد. همجنین انتظار می‌رود تا سرور در پاسخ به این درخواست، داده‌های مناسبی را به فرمت JSON به مرورگر پس بفرستد. در حالتی که bServerSide مقدار true به خود بگیرد، پلاگین فقط رابطه متقابل بین کاربر و سرور را مدیریت می‌کند و هیچ پردازشی را انجام نمی‌دهد.

در این درخواست XHR یا Ajax ی پارامترهایی که به سرور ارسال می‌شوند این‌ها هستند:

iDisplayStart عدد صحیح
نقطه شروع مجموعه داده جاری

iDisplayLength عدد صحیح
تعداد رکوردهایی که جدول می‌تواند نمایش دهد. تعداد رکوردهایی که از طرف سرور برگشت داده می‌شود باید با این عدد یکسان باشند.

iColumns عدد صحیح
تعداد ستونهایی که باید نمایش داده شوند.

sSearch رشته
فیلد جستجوی عمومی

bRegex بولین
اگر true باشد معنی آن این است که می‌توان از عبارات باقاعده برای جستجوی عبارتی خاص در کل ستون‌های جدول استفاده کرد. مثلا در کادر جستجو نوشت :

```
^[1-5]$
```

که یعنی 1 و 5 همه عددهای بین 1و 5.

bSearchable_(int) بولین
نمایش می‌دهد که یک ستون در طرف کاربر قابلیت searchable آن true هست یا نه.

sSearch_(int) رشته
فیلتر مخصوص هر ستون. اگر از ویژگی multi column filtering پلاگین استفاده شود به صورت sSearch0 ، sSearch1 ، sSeach2 و ... به طرف سرور ارسال می‌شوند. شماره انتهای هر کدام از پارامترها بیانگر شماره ستون جدول است.

bRegex_(int) بولین
اگر true باشد، بیان می‌کند که می‌توان از عبارت با قاعده در ستون شماره int جهت جستجو استفاده کرد.

bSortable_(int) بولین

مشخص می‌کند که آیا یک ستون در سمت کلاینت، قابلیت مرتب شدن بر اساس آن وجود دارد یا نه. (در اینجا int اندیس ستون را مشخص می‌کند)

iSortingCols عدد صحیح
تعداد ستون هایی که باید مرتب سازی بر اساس آنها صورت پذیرد. در صورتی که از امکان multi column sorting استفاده کنید این مقدار می‌تواند بیش از یکی باشد.

iSortCol_(int) عدد صحیح
شماره ستونی که باید بر اساس آن عملیات مرتب سازی صورت پذیرد.

sSortDir_(int) رشته
نحوه مرتب سازی ؛ شامل صعودی (asc) یا نزولی (desc)

mDataProp_(int) رشته
اسم ستون‌های درون جدول را مشخص می‌کند.

sEcho رشته
اطلاعاتی که datatables از آن برای رندر کردن جدول استفاده می‌کند.

شکل زیر نشان می‌دهد که چه پارامترهایی به سرور ارسال می‌شوند.

Parameters application/x-www-form-urlencoded

bRegex	false
bRegex_0	false
bRegex_1	false
bRegex_2	false
bRegex_3	false
bRegex_4	false
bSearchable_0	true
bSearchable_1	true
bSearchable_2	true
bSearchable_3	true
bSearchable_4	true
bSortable_0	true
bSortable_1	true
bSortable_2	true
bSortable_3	true
bSortable_4	true
iColumns	5
iDisplayLength	25
iDisplayStart	0
iSortCol_0	1
iSortingCols	1
mDataProp_0	Engine
mDataProp_1	Name
mDataProp_2	Platform
mDataProp_3	Version
mDataProp_4	Grade
sColumns	
sEcho	2
sSearch	
sSearch_0	
sSearch_1	
sSearch_2	
sSearch_3	
sSearch_4	
sSortDir_0	desc

شکل ب) پارامترهای ارسالی به سرور به صورت json

بعضی از این پارامترها بسته به تعداد ستون‌ها قابل تغییر هستند. (آن پارامترهایی که آخرشان یک عدد هست که نشان دهنده شماره ستون مورد نظر می‌باشد)

در پاسخ به هر درخواست XHR که datatables به سرور می‌فرستد، انتظار دارد تا سرور نیز یک شیء json را با فرمت مخصوص که شامل پارامترهای زیر می‌شود به او پس بفرستد:

iTotalRecords عدد صحیح
تعداد کل رکوردها (قبل از عملیات جستجو) یا به عبارت دیگر تعداد کل رکوردهای درون آن جدول از دیتابیس که داده‌ها باید از آن دریافت شوند. تعداد کل رکوردهایی که در طرف سرور وجود دارند. این مقدار فقط برای نمایش به کاربر برگشت داده می‌شود و نیز از آن برای صفحه بندی هم استفاده می‌شود.

iTotalDisplayRecords عدد صحیح
تعداد کل رکوردها (بعد از عملیات جستجو) یا به عبارت دیگر تعداد کل رکوردهایی که بعد از عملیات جستجو پیدا می‌شوند نه فقط آن تعداد رکوردی که به کاربر پس فرستاده می‌شوند. تعداد کل رکوردهایی که با شرط جستجو مطابقت دارند. اگر کاربر چیزی را جستجو نکرده باشد مقدار این پارامتر با پارامتر iTotalRecords یکسان خواهد بود.

sEcho عدد صحیح
یک عدد صحیح است که در قالب رشته در تعامل بین سرور و کلاینت جا به جا می‌شود. این مقدار به ازاء هر درخواست تغییر می‌کند. همان مقداری که مرورگر به سرور می‌دهد را سرور هم باید به مرورگر تحویل بدهد. برای جلوگیری از حملات XSS باید آن را تبدیل به عدد صحیح کرد. پلاگین DataTables مقدار این پارامتر را برای هماهنگ کردن و منطبق کردن درخواست ارسال شده و جواب این درخواست استفاده می‌کند. همان مقداری که مروگر به سرور می‌دهد را باید سرور تحویل به مرورگر بدهد.

sColumns رشته
اسم ستون‌ها که با استفاده از کاما از هم جدا شده اند. استفاده از آن اختیاری است و البته منسوخ هم شده است و در نسخه‌های جدید jQuery DataTables از آن پشتیبانی نمی‌شود.

aaData آرایه
همان طور که قبلا هم گفتیم، مقادیر سلول هایی را که باید در جدول نشان داده شوند را در خود نگهداری می‌کند. یعنی در واقع داده‌های جدول در آن ریخته می‌شوند. هر وقت که DataTables داده‌های مورد نیازش را دریافت می‌کند، سلول‌های جدول html مربوطه اش را از روی آرایه aaData ایجاد می‌کند. تعداد ستون‌ها در این آرایه دو بعدی، باید با تعداد ستون‌های جدول html مربوطه به آن یکسان باشد

شکل زیر پارامترها دریافتی از سرور را نشان می‌دهند:

شکل ب) پارامترهای دریافتی از سرور به صورت json

استفاده از روش server side processing در mvc

همان طور که گفتیم، کلاینت به سرور یک سری پارامترها را ارسال می‌کند و آن پارامترها را هم شرح دادیم. برای دریافت این پارامترها طرف سرور، احتیاج به یک مدل هست. این مدل به صورت زیر پیاده سازی خواهد شد:

```
/// <summary>
/// Class that encapsulates most common parameters sent by DataTables plugin
/// </summary>
public class jQueryDataTableParamModel
{
    /// <summary>
    /// Request sequence number sent by DataTable,
    /// same value must be returned in response
    /// </summary>
    public string sEcho { get; set; }
    /// <summary>
    /// Text used for filtering
    /// </summary>
    public string sSearch { get; set; }
    /// <summary>
    /// Number of records that should be shown in table
    /// </summary>
    public int iDisplayLength { get; set; }
    /// <summary>
    /// First record that should be shown(used for paging)
    /// </summary>
    public int iDisplayStart { get; set; }
    /// <summary>
    /// Number of columns in table
    /// </summary>
    public int iColumns { get; set; }
    /// <summary>
    /// Number of columns that are used in sorting
    /// /// </summary>
    public int iSortingCols { get; set; }
    /// <summary>
    /// Comma separated list of column names
    /// </summary>
    public string sColumns { get; set; }
}
```

مدل بایندر mvc وظیفه مقداردهی به خصوصیات درون این کلاس را بر عهده دارد، بقیه پارامترهایی که به سرور ارسال می‌شوند و در این کلاس نیامده اند، از طریق شیء Request در دسترس خواهند بود.

اکشن متدی که مدل بالا را دریافت می‌کند، می‌تواند به صورت زیر پیاده سازی شود. این اکشن متد وظیفه پاسخ دادن به درخواست DataTables بر اساس پارامترهای ارسال شده در مدل DataTablesParam را دارد. خروجی این اکشن متد شامل پارارمترهای مورد نیاز پلاگین DataTables برای تشکیل جدول است که آنها را هم شرح دادیم.

```
public JsonResult GetBrowsers(jQueryDataTableParamModel param)
{
        IQueryable<Browser> allBrowsers = new Browsers().CreateInMemoryDataSource().AsQueryable();

        IEnumerable<Browser> filteredBrowsers;

        // Apply Filtering
        if (!string.IsNullOrEmpty(param.sSearch))
        {
                filteredBrowsers = new Browsers().CreateInMemoryDataSource()
                    .Where(x => x.Engine.Contains(param.sSearch)
                                || x.Grade.Contains(param.sSearch)
                                || x.Name.Contains(param.sSearch)
                                || x.Platform.Contains(param.sSearch)
                    ).ToList();
                float f;
                if (float.TryParse(param.sSearch, out f))
                {
                    filteredBrowsers = filteredBrowsers.Where(x => x.Version.Equals(f));
                }
        }
        else
        {
                filteredBrowsers = allBrowsers;
        }

        // Apply Sorting
        var sortColumnIndex = Convert.ToInt32(Request["iSortCol_0"]);
        Func<Browser, string> orderingFunction = (x => sortColumnIndex == 0 ? x.Engine :
                                                        sortColumnIndex == 1 ? x.Name :
                                                        sortColumnIndex == 2 ? x.Platform :
                                                        sortColumnIndex == 3 ? x.Version.ToString()
:
                                                        sortColumnIndex == 4 ? x.Grade :
                                                            x.Name);

        var sortDirection = Request["sSortDir_0"]; // asc or desc
        filteredBrowsers = sortDirection == "asc" ? filteredBrowsers.OrderBy(orderingFunction) :
filteredBrowsers.OrderByDescending(orderingFunction);

        // Apply Paging
        var enumerable = filteredBrowsers.ToArray();
        IEnumerable<Browser> displayedBrowsers = enumerable.Skip(param.iDisplayStart).
                Take(param.iDisplayLength).ToList();

        return Json(new
        {
                sEcho = param.sEcho,
                iTotalRecords = allBrowsers.Count(),
                iTotalDisplayRecords = enumerable.Count(),
                aaData = displayedBrowsers
        }, JsonRequestBehavior.AllowGet);
}
```

تشریح اکشن متد GetBrowsers :

این اکشن متد از مدل jQueryDataTableParamModel به عنوان پارامتر ورودی خود استفاده می‌کند. این مدل همان طور هم که گفتیم، شامل یک سری خصوصیت است که توسط پلاگین jQuery DataTables مقداردهی می‌شوند و همچنین مدل بایندر mvc وظیفه بایند کردن این مقادیر به خصوصیات درون این کلاس را بر عهده خواهد داشت. درون بدنه اکشن متد GetBrowsers داده‌ها بعد از اعمال عملیات فیلترینگ، مرتب سازی، و صفحه بندی به فرمت مناسبی درآمده و به طرف مرورگر فرستاده خواهند شد.

برای پیاده سازی کدهای طرف کلاینت نیز، درون یک View کدهای زیر قرار خواهند گرفت:

```
$(function () {
        var $table = $('#browsers-grid');
        $table.dataTable({
                "bProcessing": true,
                "bStateSave": true,
                "bServerSide": true,
                "bFilter": true,
                "sDom": 'T<"clear">lftipr',
                "aLengthMenu": [[5, 10, 25, 50, -1], [5, 10, 25, 50, "All"]],
                "bAutoWidth": false,
                "sAjaxSource": "/Home/GetBrowsers",
                "fnServerData": function (sSource, aoData, fnCallback) {
                    $.ajax({
                        "dataType": 'json',
                        "type": "POST",
                        "url": sSource,
                        "data": aoData,
                        "success": fnCallback
                    });
                },
                "aoColumns": [
                    { "mDataProp": "Engine" },
                    { "mDataProp": "Name" },
                    { "mDataProp": "Platform" },
                    { "mDataProp": "Version" },
                    { "mDataProp": "Grade" }
                ],
                "oLanguage": {
                        "sUrl": "/Content/dataTables.persian.txt"
                }
        });
});
```

تشریح کدها:

fnServerData :

این متد، در واقع نحوه تعامل سرور و کلاینت را با استفاده از درخواستهای XHR مشخص خواهد کرد.

oLanguage :

برای فعال سازی زبان فارسی، فیلدهای مورد نیاز ترجمه شده و در یک فایل متنی قرار داده شده اند. کافی است آدرس این فایل متنی به ویژگی oLanguage اختصاص داده شوند.

مثال این قسمت را از لینک زیر دریافت کنید:
DataTablesTutorial04.zip

لازم به ذکر است پوشه bin، obj، و packages جهت کاهش حجم این مثال از solution حذف شده اند. برای اجرای این مثال از اینجا کمک بگیرید.

مطالعه بیشتر

برای مطالعه بیشتر در مورد این پلاگین و نیز پیاده سازی آن در MVC می‌توانید به لینک زیر نیز مراجعه بفرمائید که بعضی از قسمتهای این مطلب هم از مقاله زیر استفاده کرده است:
jQuery DataTables and ASP.NET MVC Integration - Part I

نظرات خوانندگان

نویسنده: Sorosh
تاریخ: ۱۳۹۲/۱۱/۱۱ ۸:۴۹

با سلام و تشکر. می‌خوام یک ستون ردیف به این جدول اضافه کنم که به ازای صفحه، ردیف جلو بره یعنی از 1 نشه دوباره و یکی هم اضافه کردن یک Attr خاص و جدید به هر سطر در جدول مثل ProjectCode که اون هم داخلش Id اون سطر در دیتا بیس هستش . ممنونم اگه کمکم کنید

کد کارم رو هم می‌دهم . البته من در WebForm کار کردم.

```
var $table = $('#browsers-grid');
$table.dataTable({
"bJQueryUI": true,
"bProcessing": true,
"bSortClasses": false,
"bServerSide": true,
"bFilter": true,
"sPaginationType": "full_numbers",
"sScrollY": 400,
"sScrollX": "100%",
"sScrollXInner": "110%",
"bLengthChange": false,
"iDisplayLength": 20,

"aLengthMenu": [[5, 10, 25, 50, -1], [5, 10, 25, 50, "All"]],
"bAutoWidth": false,
"sAjaxSource": "Commands.aspx?cmd=all",
"fnServerData": function (sSource, aoData, fnCallback) {
  $.ajax({
"dataType": 'json',
"type": "POST",
"url": sSource,
"data": aoData,
"success": fnCallback
  });
},

"aoColumns":
  [

  { "mDataProp": "Code" },
  { "mDataProp": "Caption" },
  { "mDataProp": "Comment" }
  ],

"oLanguage":
  {
"sUrl": "dataTables.persian.txt"
  }
});
```

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۱۱/۱۱ ۹:۴۷

سلام، خواهش می‌کنم. می‌تونید در سمت سرور بعد از واکشی اطلاعات از دیتابیس اونو داخل یک منبع داده درون حافظه ای بریزید و هر تعداد ستون که لازم دارید به اون منبع داده جدید اضافه کنید. و با مقدارهای مناسبی هر مدخل از اون منبع داده رو پر کنید. مثلا فرضا اگه جدول دیتابیس شما دارای سه ستون Code و Caption و Comment هست کلاس جدیدی بسازید که این سه تا ستون رو داره (به عنوان پروپرتی) و پروپرتی‌های دلخواه دیگه ای هم داره. مثلا پروپرتی RowNumber . بعد لیستی از داده‌ها رو که از دیتابیس واکشی کردید داخل لیستی از ViewModel ساخته شده بریزید و خصوصیت RowNumber هر ViewModel رو مقداردهی مناسبی کنید.

نویسنده: Sorosh
تاریخ: ۱۳۹۲/۱۱/۱۱ ۱۱:۶

متشکرم از پاسخ شما . البته اگه می‌شد نمونه‌ای ایجاد کنید خیلی بهتر بود که باز هم ممنونم. اما فقط برای بحث خصوصیت هر

سطر چی؟ می‌دونید اگه بخواهیم به هر سطر یک Attr جدید بدهیم چطوری باید اونو به TR نسبت بدهیم. مثلا من می‌خواهم به ازای هر سطر <tr ProjectCode='12'><tr/> داشته باشم.

نویسنده: Sorosh
تاریخ: ۱۳۹۲/۱۱/۱۱ ۱۳:۱

با سلام مجدد؛ بالاخره پیداش کردم. برای ایجاد خصوصیت جدید به ازاء هر سطر و گرفتن مقدار از aoData بصورت زیر است .
مثلا برای اضافه کردن Attr - StudentCode

```
fnRowCallback": function (nRow, aData, iDisplayIndex)
{

  var StudentCode= aData["Code"];
  $(nRow).attr("StudentCode",StudentCode);
  return nRow;
}
```

با تشکر از شما

نویسنده: sorosh
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۳:۳۱

با سلام؛ من می‌خوام با قراردادن یک دکمه روی فرم به نام به‌روز رسانی، خاصیت اطلاعات داخل کوکی که با متد bStateSave کار می‌کنه از بین بره و جدول دوباره از نو رفرش بشه و حالت‌های جستجو و مرتب سازی قبلی از بین بره. متشکرم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۴:۲

- راه حل رسمی برای حذف کوکی ندارد . فقط مدت زمان آن قابل تنظیم است.
- به مطلب « کوکی در جاوا اسکریپت » در مورد حذف کوکی در همان سمت کلاینت مراجعه کنید. نام کوکی‌ها را هم با استفاده از افزونه Cookies manager می‌توانید مشاهده کنید.

نویسنده: رویا حیدری
تاریخ: ۱۳۹۳/۱۰/۱۶ ۱۶:۳۱

با سلام و تشکر فراوان. من میخواستم یک ستون با عنوان عملیات به جدول اضافه کنم که به صورت منو باشه و عملیات مختلف مثل ویرایش و حذف و تغییرات دیگه را از اون منو انتخاب و انجام بدم. برای اینکار باید چه کاری انجام بدم؟ و اینکه آیا برای اضافه کردن امکان ویرایش باید حتما از Editable DataTable استفاده کنم یا میشه تو همین نوع ساده هم امکان ویرایش را اضافه کرد؟ (Inline Edit نمی‌خوام انجام بدم)
ممنون میشم اگه راهنمایی کنید.

نویسنده: رویا حیدری
تاریخ: ۱۳۹۳/۱۰/۱۷ ۱۵:۰

در این صفحه تا حدودی به جوابم رسیدم.

عنوان: **بارگزاری PartialView با استفاده از jQuery در زمان اجرا**
نویسنده: پویا امینی
تاریخ: ۱۳۹۲/۰۵/۲۲ ۱۸:۲۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net MVC, jQuery, jQuery Ajax, MVC

مزیت استفاده از PartialViewها، ماژولار کردن برنامه است. برای مثال اگر صفحه جاری شما قرار است از چهار قسمت اخبار، منوی پویا، سخن روز و آمار کاربران تشکیل شود، می‌توان هر کدام را توسط یک PartialView پیاده سازی کرد و سپس صفحه اصلی را از کنار هم قرار دادن این PartialViewها تهیه نمود.(منبع).
در این پست قصد دارم به نحوه بارگزاری یک PartialView با استفاده از ASP.NET MVC بپردازم.
ابتدا یک پروژه جدید ایجاد کنید. حال می‌خواهیم زمانیکه صفحه اصلی سایت بارگزاری می‌شود، لیست تمام محصولات را نمایش دهد. برای نمایش محصولات یک PartialView جدید را ایجاد می‌کنیم؛ همانند شکل ذیل:

حال برای ساده کردن مثال، متن ثابتی را درون این PartialView می‌نویسیم:

```
product  List Partial
```

در ادامه قصد داریم که درIndex، این View را بارگزاری کنیم. برای این کار از متد ajax مربوط به کتابخانه jQuery به صورت زیر استفاده می‌کنیم:

```
$(function () {
        $.ajax({
            //مشخص کردن  اکشنی که باید فراخوانی شود
            url: '/Home/Details',
            contentType: 'application/html; charset=utf-8',
            type: 'GET',
            //نوع نتیجه بازگشتی
            dataType: 'html'

        })
.success(function (result) {
    //زمانی که کدهای سمت سرور بدون خطا اجرا شده اند
    //این قسمت فراخوانی می‌شود و نتیجه اکشن درون متغیر
    //result
    //قرار می‌گیرد
    $('#sectionContents').html(result);
```

```
})
.error(function (xhr, status) {
    alert(xhr.responseText);
});
    });
```

و پس از آن محلی را که قرار است PartialView در آن بارگزاری شود، ایجاد می‌کنیم:

```
<div id="sectionContents"></div>
```

حال فقط باید اکشن مورد نظر را در HomeController پیاده سازی کنیم:

```
public ActionResult Details()
        {
           return PartialView("partial/_ProductList");
        }
```

با استفاده از این کد مشخص کردیم که این اکشن، یک PartialView را با نام _ProductList برگشت می‌دهد. البته در یک مثال واقعی باید لیست محصولات را هم به این PartialView پاس دهیم.
نتیجه اجرا به صورت زیر است:

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۲۲ ۱۹:۵۹

مطالب تکمیلی که از همین روش استفاده می‌کنند:

به روز رسانی غیرهمزمان قسمتی از صفحه به کمک jQuery در ASP.NET MVC

پیاده سازی دکمه «بیشتر» یا «اسکرول نامحدود» به کمک jQuery در ASP.NET MVC

آشنایی با تکنیک‌های Ajax در ASP.NET MVC

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۲۳ ۰:۲۸

یک سؤال: «بارگزاری» درست هست یا « بارگذاری »؟ البته مطابق مطلبی که در ویکی‌پدیا نوشته باید از «بارگیری» استفاده کرد!

نویسنده: imo0
تاریخ: ۱۳۹۲/۰۶/۰۷ ۱۰:۶

یک سوال. من یک پارشال ویو دارم که خودش نیاز داره به یه سری کدهای جاوا اسکریپت و استایل .
من پارشال ویو هارو با اجکس لود میکنم . چطوری اون فایل‌های استایل و غیرشم لود کنم ؟. @RenderSection جواب نمیده.
چون اصلا صفحه ریفرش نمیشه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۰۷ ۱۱:۸

- راه اول: زمانیکه View اصلی در برگیرنده آن نمایش داده می‌شود، کلیه فایل‌های متناظر را هم الحاق کنید تا به صورت خودکار در جزئی از صفحه آن، که بعدا به روز خواهد شد، نیز اعمال شود.
- راه دوم: اصلا از RenderSection در یک partial view که قرار است Ajax ایی بارگذاری شود، استفاده نکنید. معمولی این‌ها را الصاق یا تعریف کنید. مثل تعاریف یک HTML ساده. یک نفر هم اینجا براش HtmlHelper نوشته ولی نکته اصلی یکی است؛ الصاق و تعریف معمولی فایل‌های مورد نیاز.
- همچنین خود jQuery امکان بارگذاری اسکریپت‌ها را به صورت پویا دارد . زمانیکه complete عملیات Ajax ایی رخداد، متد getScript عنوان شده را فراخوانی کنید. برای CSS هم به صورت زیر عمل کنید:

```
$("<style></style>").appendTo("head").html(data);
```

نویسنده: خوشقدم
تاریخ: ۱۳۹۲/۰۶/۱۳ ۸:۵۴

سلام

مشکلی که من دارم این است که در صفحه دوتا PartialView دارم که هر کدام با استفاده jquery همان طور که شما توضیح داده اید لود می‌شوند ، اما برای رفرش کردن اونها مشکل دارم و نمی‌تونم به هر کدام شماره Page و سایر اطلاعات WebGrid چطوری ارسال کنم؟

ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۱۳ ۹:۰

نحوه صحیح ارسال پارامترها توسط jQuery Ajax
ارسال کوئری استرینگ‌ها به همراه عملیات Ajax در jQuery

ضمنا بحث وب گرید ، خارج از موضوع مطلب جاری است. از پارامتر ajaxUpdateContainerId آن افزونه خاص باید برای مدیریت مسایل paging و sorting استفاده کنید.

نویسنده: مجتبی شایان فر
تاریخ: ۱۳۹۲/۰۶/۲۸ ۱۳:۴

باسلام و خسته نباشید.
من مثال فوق را خط به خط اجرا کردم ولی partial view نمایش داده نمیشه. فکر کنم مکان قطعه کد Ajax را اشتباه جایگذاری کردم اگه ممکنه راهنمائی می‌کنید که قطعه:

```
$( function () {
$.ajax({
//مشخص کردن  اکشنی که باید فراخوانی شود
url: '/Home/Details' ,
contentType: 'application/html; charset=utf-8' ,
type: 'GET' ,
//نوع نتیجه بازگشتی
dataType: 'html'

})
.success( function (result) {
  //زمانی که کدهای سمت سرور بدون خطا اجرا شده اند
  //این قسمت فراخوانی می‌شود و نتیجه اکشن درون متغیر
  //result
  //قرار می‌گیرد
  $( '#sectionContents' ).html(result);
})
.error( function (xhr, status) {
  alert(xhr.responseText);
});
  });
```

دقیقا کجای Index باید قرارداده بشه؟ سورس پروژه را هم ارسال می‌کردید خیلی خوب می‌شد .
ممنون.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۲۸ ۲۰:۴

نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery

نویسنده: منیژه محمدی
تاریخ: ۱۳۹۲/۰۸/۱۶ ۱۰:۴۴

سلام
من این مثال را قدم به قدم انجام دادم ولی خطای 404 میده و localhost.../home/Details را پیدا نمی‌کند.
با اینکه من پارشیال ویو را در این تابع تعریف کردم ....

نویسنده: وحید نصیری

تاریخ: ۱۳۹۲/۰۸/۱۶ ۱۰:۵۴

- برنامه را با فایرباگ دیباگ کنید تا مسیرهای درخواستی و خطاها را بهتر مشاهده کنید.
- نباید آدرس‌ها را مستقیما و به صورت رشته‌ای تعریف کنید. اطلاعات بیشتر
- در مثال عنوان شده «partial/_ProductList» این مورد به معنای وجود یک پوشه partial در اینجا است (تصویر اول). البته غیر الزامی است ولی برای باز تولید این مثال باید به آن دقت داشته باشید.

نویسنده: منیژه محمدی
تاریخ: ۱۳۹۲/۰۸/۱۷ ۱۱:۳۶

با سلام
من همه موارد را خیلی چک کردم از اینکه مسیر یا نام را درست داده باشم
علت این بود که من یک [httppost] بالای ان ActionResult که میگفت پیدا نمیکنم اضافه کرده بودم برای همین ان را پیدا نمی‌کرد
حالا من میخام یک سری اطلاعات به این اکشن پست کنم اینطوری مشکل پیدا میکنم چطور میتونم ان را حل کنم ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۱۷ ۱۱:۴۸

type در $.ajax مثال اصلی بحث جاری، مساوی Get است؛ تبدیلش کنید به Post. کامنت اول این مطلب سه مثال مشابه دیگر را عنوان کرده. اکثر این‌ها از حالت Post استفاده کرده‌اند.

نویسنده: آروین
تاریخ: ۱۳۹۲/۰۸/۲۱ ۱۱:۱۹

با عرض سلام و خسته نباشید
این روش فرقش با jquery.load('url') چیه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۱ ۱۴:۲۴

load در پشت صحنه از همین متد ajax استفاده می‌کند. تمام متدهای کمکی از این دست فقط محصور کننده تابع ajax اصلی جی‌کوئری هستند.

نویسنده: رشنو
تاریخ: ۱۳۹۲/۱۲/۱۵ ۱۰:۳۶

با سلام و تشکر
لیست محصولات رو چطور باید پاس داد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۵ ۱۰:۵۸

به روش‌های متداول « بررسی نحوه انتقال اطلاعات از یک کنترلر به Viewهای مرتبط با آن »

عنوان: **طراحی ValidationAttribute دلخواه و هماهنگ سازی آن با ASP.NET MVC**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۷/۲۱ ۱۵:۳۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, MVC, ASP.Net MVC, jQuery

در سری پست‌های آقای مهندس یوسف نژاد با عنوان Globalization در ASP.NET MVC روشی برای پیاده سازی کار با Resourceها در ASP.NET با استفاده از دیتابیس شرح داده شده است. یکی از کمبودهایی که در این روش وجود داشت عدم استفاده از این نوع Resourceها از طریق Attributeها در ASP.NET MVC بود. برای استفاده از این روش در یک پروژه به این مشکل برخورد کردم و پس از تحقیق و بررسی چند پست سرانجام در این پست پاسخ خود را پیدا کرده و با ترکیب این روش با روش آقای یوسف نژاد موفق به پیاده سازی Attribute دلخواه شدم.
در این پست و با استفاده از سری پست‌های آقای مهندس یوسف نژاد در این زمینه، یک Attribute جهت هماهنگ سازی با سیستم اعتبار سنجی ASP.NET MVC در سمت سرور و سمت کلاینت (با استفاده از jQuery Validation) بررسی خواهد شد.

قبل از شروع مطالعه سری پست‌های MVC و Entity Framework الزامی است.

برای انجام این کار ابتدا مدل زیر را در برنامه خود ایجاد می‌کنیم.

```
using System;

public class SampleModel

{

    public DateTime StartDate { get; set; }

    public string Data { get; set; }

    public int Id { get; set; }

}
```

با استفاده از این مدل در برنامه در زمان ثبت داده‌ها هیچ گونه خطایی صادر نمی‌شود. برای اینکه بتوان از سیستم خطای پیش فرض ASP.NET MVC کمک گرفت می‌توان مدل را به صورت زیر تغییر داد.

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

public class SampleModel
{
    [Required(ErrorMessage = "Start date is required")]
    public DateTime StartDate { get; set; }

    [Required(ErrorMessage = "Data is required")]
    public string Data { get; set; }

    public int Id { get; set; }
}
```

حال ویو این مدل را طراحی می‌کنیم.

```
@model SampleModel
@{
    ViewBag.Title = "Index";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<section>
    <header>
        <h3>SampleModel</h3>
```

```
    </header>
    @Html.ValidationSummary(true, null, new { @class = "alert alert-error alert-block" })

    @using (Html.BeginForm("SaveData", "Sample", FormMethod.Post))
    {
        <p>
            @Html.LabelFor(x => x.StartDate)
            @Html.TextBoxFor(x => x.StartDate)
            @Html.ValidationMessageFor(x => x.StartDate)
        </p>
        <p>
            @Html.LabelFor(x => x.Data)
            @Html.TextBoxFor(x => x.Data)
            @Html.ValidationMessageFor(x => x.Data)
        </p>
        <input type="submit" value="Save"/>
    }
</section>
```

و بخش کنترلر آن را به صورت زیر پیاده سازی می‌کنیم.

```
public class SampleController : Controller
    {
        //
        // GET: /Sample/

        public ActionResult Index()
        {
            return View();
        }

        public ActionResult SaveData(SampleModel item)
        {
            if (ModelState.IsValid)
            {
                //save data
            }
            else
            {
                ModelState.AddModelError("","لطفا خطاهای زیر را برطرف نمایید");
                RedirectToAction("Index", item);
            }
            return View("Index");
        }
    }
```

حال با اجرای این کد و زدن دکمه Save صفحه مانند شکل پایین خطاها را نمایش خواهد داد.

تا اینجای کار روال عادی همیشگی است. اما برای طراحی Attribute دلخواه جهت اعتبار سنجی (مثلا برای مجبور کردن کاربر به وارد کردن یک فیلد با پیام دلخواه و زبان دلخواه) باید یک کلاس جدید تعریف کرده و از کلاس RequiredAttribute ارث ببرد. در پارامتر ورودی این کلاس جهت کار با Resourceهای ثابت در نظر گرفته شده است اما برای اینکه فیلد دلخواه را از دیتابیس بخواند این روش جوابگو نیست. برای انجام آن باید کلاس RequiredAttribute بازنویسی شود.

کلاس طراحی شده باید به صورت زیر باشد:

```
public class VegaRequiredAttribute : RequiredAttribute, IClientValidatable
    {
#region Fields (2)

        private readonly string _resourceId;
        private String _resourceString = String.Empty;

#endregion Fields

#region Constructors (1)

        public VegaRequiredAttribute(string resourceId)
        {
            _resourceId = resourceId;
            ErrorMessage = _resourceId;
            AllowEmptyStrings = true;
        }

#endregion Constructors

#region Properties (1)

        public new String ErrorMessage
        {
            get { return _resourceString; }
            set { _resourceString = GetMessageFromResource(value); }
        }

#endregion Properties

#region Methods (2)
```

```
// Public Methods (1)

        public IEnumerable<ModelClientValidationRule> GetClientValidationRules(ModelMetadata metadata,
ControllerContext context)
        {

            yield return new ModelClientValidationRule
            {
                ErrorMessage = GetMessageFromResource(_resourceId),
                ValidationType = "required"
            };
        }
// Private Methods (1)

        private string GetMessageFromResource(string resourceId)
        {
            var errorMessage = HttpContext.GetGlobalResourceObject(_resourceId, "Yes") as string;
            return errorMessage ?? ErrorMessage;
        }

#endregion Methods
    }
```

در این کلاس دو نکته وجود دارد.

1- ابتدا دستور

```
HttpContext.GetGlobalResourceObject(_resourceId, "Yes") as string;
```

که عنوان کلید Resource را از سازنده کلاس گرفته (کد اقای یوسف نژاد) رشته معادل آن را از دیتابیس بازیابی میکند.

2- ارث بری از اینترفیس IClientValidatable، در صورتی که از این اینترفیس ارث بری نداشته باشیم. Validator طراحی شده در طرف کلاینت کار نمی‌کند. بلکه کاربر با کلیک بروی دکمه مورد نظر داده‌ها را به سمت سرور ارسال می‌کند. در صورت وجود خطا در پست بک خطا نمایش داده خواهد شد. اما با ارث بری از این اینترفیس و پیاده سازی متد GetClientValidationRules می‌توان تعریف کرد که در طرف کلاینت با استفاده از Unobtrusive jQuery پیام خطای مورد نظر به کنترل ورودی مورد نظر (مانند تکست باکس) اعمال می‌شود. مثلا در این مثال خصوصیت data-val-required به input هایی که قبلا در مدل ما Reqired تعریف شده اند اعمال می‌شود.

حال در مدل تعریف شده می‌توان به جای Required می‌توان از VegaRequiredAttribute مانند زیر استفاده کرد. (همراه با نام کلید مورد نظر در دیتابیس)

```
public class SampleModel
{
    [VegaRequired("RequiredMessage")]
    public DateTime StartDate { get; set; }

    [VegaRequired("RequiredMessage")]
    public string Data { get; set; }

    public int Id { get; set; }
}
```

ورودی Validator مورد نظر نام کلیدی است به زبان دلخواه که عنوان آن RequiredMessage تعریف شده است و مقدار آن در دیتابیس مقداری مانند " **تکمیل این فیلد الزامی است** " است. با این کار در زمان اجرا با استفاده از این ولیدتور ابتدا کلید مورد نظر با توجه به زبان فعلی از دیتابیس بازیابی شده و در متادیتابی مدل ما قرار می‌گیرد. به جای استفاده از Resourceها می‌توان پیام‌های خطای دلخواه را در دیتابیس ذخیره کرد و در مواقع ضروری جهت جلوگیری از تکرار از آنها استفاده نمود.

با اجرای برنامه اینبار خروجی به شکل زیر خواهد بود.

جهت فعال ساری اعتبار سنجی سمت کلاینت ابتدا باید اسکریپت‌های زیر به صفحه اضافه شود.

```
<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.validate.unobtrusive.min.js")"
type="text/javascript"></script>
```

سپس در فایل web.config تنظیمات زیر باید اضافه شود

```
<appSettings>
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true"/>
</appSettings>
```

سپس برای اعمال Validator طراحی شده باید توسط کدهای جاوا اسکریپت زیر داده‌های مورد نیاز سمت کلاینت رجیستر شوند.

```
<script type="text/javascript">
        jQuery.validator.addMethod('required', function (value, element, params) {
            if (value == null | value == "") {
                return false;
            } else {
                return true;
            }

        }, '');

        jQuery.validator.unobtrusive.adapters.add('required', {}, function (options) {
            options.rules['required'] = true;
            options.messages['required'] = options.message;
        });
    </script>
```

البته برای مثال ما قسمت بالا به صورت پیش فرض رجیستر شده است اما در صورتی که بخواهید یک ولیدتور دلخواه و غیر استاندارد بنویسید روال کار به همین شکل است.

موفق و موید باشید.

منابع ^ و ^ و ^ و ^

نظرات خوانندگان

نویسنده: امیر خلیلی
تاریخ: ۱۳۹۲/۰۸/۰۴ ۸:۵۹

یک مشکل اساسی وجود داره
هنگامی که با استفاده از جاواسکریپت اعتبارسنجی انجام میشه کاربر یا اون شخصی که قراره کدهای مخرب را وارد کنه میتونه استفاده از فایل‌های JS را در مرورگش غیر فعال کنه

و اینگونه اعتبار سنجی انجام نمیشه !

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۰۴ ۹:۹

اعتبارسنجی در ASP.NET MVC دو مرحله‌ای است. مرحله اول آن فقط سمت کاربر است. مرحله دوم به تفسیر IClientValidatable در سمت سرور ختم می‌شود.

عنوان: ذخیره TreeView ساخته شده توسط KendoUI در Asp.net MVC
نویسنده: حسین حسینی 128
تاریخ: ۱۳۹۲/۰۸/۱۳ ۱۲:۵
آدرس: www.dotnettips.info
گروه‌ها: MVC, jQuery, Kendo UI

همانطور که از نمونه مثال‌های خود Kendo UI مشاهده میشود ، نحوه استفاده از TreeView آن به صورت زیر است :

```
<div>
@(Html.Kendo().TreeView()
        .Name("treeview")
        .TemplateId("treeview-template")
        .HtmlAttributes(new { @class = "demo-section" })
        .DragAndDrop(true)
        .BindTo(Model.Where(e=>e.ParentFolderID==null).OrderBy(e=>e.Order), mappings =>
        {
            mappings.For<DAL.Folder>(binding => binding
                    .ItemDataBound((item, folder) =>
                    {
                        item.Text = folder.FolderName;
                        item.SpriteCssClasses = "folder";
                        item.Expanded=true;
                        item.Id = folder.FolderID.ToString();
                    })
                    .Children(folder => folder.Folder1));
            mappings.For<DAL.Folder>(binding => binding
                    .ItemDataBound((item, folder) =>
                    {
                        item.Text = folder.FolderName;
                        item.SpriteCssClasses = " folder";
                        item.Expanded = true;
                        item.Id = folder.FolderID.ToString();
                    }));
        })
)
</div>
```

```
<style type="text/css" scoped>
    .demo-section {
        width: 200px;
    }

    #treeview  .k-sprite ,#treeview2 .k-sprite {
        background-image: url("@Url.Content("/Content/kendo/images/coloricons-sprite.png")");
    }
    .rootfolder { background-position: 0 0; }
    .folder { background-position: 0 -16px; }
    .pdf { background-position: 0 -32px; }
    .html { background-position: 0 -48px; }
    .image { background-position: 0 -64px; }
    .delete-link,.edit-link {
        width: 12px;
        height: 12px;
        overflow: hidden;
        display: inline-block;
        vertical-align: top;
        margin: 2px 0 0 3px;
        -webkit-border-radius: 5px;
        -mox-border-radius: 5px;
        border-radius: 5px;
    }
    .delete-link{
        background: transparent url("@Url.Content("/Content/kendo/images/close.png")") no-repeat 50%
50%;
    }
    .edit-link{
        background: transparent url("@Url.Content("/Content/kendo/images/edit.png")") no-repeat 50%
50%;
    }
</style>
```

استفاده از این TreeView ساده است ولی اگر احتیاج داشته باشیم که پس از drag&drop کردن گره‌ها آن را ذخیره کنیم چگونه باید عمل کنیم؟ این ریالسمت در خود Kendo تعبیه نشده است ، پس به صورت زیر عمل میکنیم :

پس از ساختن TreeView و اصلاح آن به شکل دلخواه لینک زیر را در ادامه اش می‌آوریم :

```
<a href="#" id="serialize">ذخیره</a>
```

سپس در تگ اسکریپت‌های خود این کد جاوا اسکریپت را برای serialize کردن تمام گره‌های TreeView مینویسیم :

```
$('#serialize').click(function () {
            serialized = serialize();
            window.location.href = "Folder/SaveMenu?serial=" + serialized + "!";
        });

        function serialize() {
            var tree = $("#treeview").data("kendoTreeView");
            var json = treeToJson(tree.dataSource.view());
            return JSON.stringify(json);
        }

        function treeToJson(nodes) {

            return $.map(nodes, function (n, i) {

                var result = { id: n.id};
                //var result = { text: n.text, id: n.id, expanded: n.expanded, checked: n.checked };
                if (n.hasChildren)
                    result.items = treeToJson(n.children.view());

                return result;
            });
        }
```

حال به تشریح کدها میپردازیم :

تابع serialize درخط اول تمام عناصر داخلی treeview را گرفته ، به صورت یک datasource قابل انقیاد درآورده وسپس datasource آن را به یک نمایش قابل تجزیه تبدیل میکند و به متد treeToJSON میفرستد.

```
var tree = $("#treeview").data("kendoTreeView");
var json = treeToJson(tree.dataSource.view());
```

تابع treeToJSON درخت را به عنوان یکسری گره گرفته و تمام عناصر آن را به فرمت json میبرد . (قسمتی که به صورت توضیحی درآمده میتواند برای بدست آوردن تمام اطلاعات گره از جمله متن و انتخاب شدن checkbox آن و غیره مورد استفاده قرار بگیرد) در ادامه این تابع اگر گره درحال استفاده فرزندی داشته باشد به صورت بازگشتی همین تابع برای آن فراخوانده میشود.

```
var result = { id: n.id};
 //var result = { text: n.text, id: n.id, expanded: n.expanded, checked: n.checked };
 if (n.hasChildren)
 result.items = treeToJson(n.children.view());
```

سپس اطلاعات برگشتی که در فرمت json هستند در خط آخر serialaize به رشته تبدیل میشوند و به رویداد کلیک که از آن فراخوانده شده بود بازمیگردند.

```
return JSON.stringify(json);
```

خط آخر رویداد نیز یک Action در Controller مورد نظر را هدف قرار میدهد و رشته بدست آمده را به آن ارسال میکند و صفحه redirect میشود.

```
window.location.href = "Folder/SaveMenu?serial=" + serialized + "!";
```

رشته ای که با عنوان serialize به کنترلر ارسال میشود مانند زیر است :

```
"[{\"id\":\"2\"},{\"id\":\"5\",\"items\":[{\"id\":\"3\"},{\"id\":\"6\"},{\"id\":\"7\"}]}]!"
```

این رشته مربوط به درختی به شکل زیر است :

همانطور که میبینید گره دوم که "پوشه چهارم45" نام دارد شامل سه فرزند است که در رشته داده شده با عنوان item شناخته شده است. حال باید این رشته با برنامه نویسی سی شارپ جداسازی کرد :

```
string serialized;
Dictionary<int, int> numbers = new Dictionary<int, int>();
```

```
public ActionResult SaveMenu(string serial)
        {
            var newfolders = new List<Folder>();
            serialized = serial;
            calculte_serialized(0);
            return RedirectToAction("Index");
        }
```

```
void calculte_serialized(int parent)
        {
            while (serialized.Length > 0)
            {
                var id_index=serialized.IndexOf("id");
                if (id_index == -1)
                {
                    return;
                }
                serialized = serialized.Substring(id_index + 5);
                var quote_index = serialized.IndexOf("\"");
                var id=serialized.Substring(0, quote_index);
                numbers.Add(int.Parse(id), parent);
                serialized = serialized.Substring(quote_index);
                var condition = serialized.Substring(0,3);
                switch (condition)
                {
                    case "\"},":
                        break;
                    case "\",\"":
                        calculte_serialized(int.Parse(id));
                        break;
                    case "\"}]":
                        return;
                        break;
                    default:
                        break;
                }
            }
        }
```

calculte_serialized با گرفتن 0 کار خود را شروع میکند یعنی از گره هایی که پدر ندارند و تمام idها را همراه با پدرشان در یک دیکشنری میریزد و هرکجا که به فرزندی برخورد به صورت بازگشتی فراخوانی میشود. پس از اجرای کامل آن ما درخت را در یک دیکشنری به صورت عنصرهای مجزا در اختیار داریم که میتوانیم در پایگاه داده ذخیره کنیم.

نظرات خوانندگان

نویسنده: امیر بختیاری
تاریخ: ۱۳۹۲/۱۱/۲۵ ۸:۳۱

با سلام و تشکر از شما
اگر tree دارای چک باکس باشد و بخواهیم نود هایی که چک خورده است را ذخیره کنیم چگونه عمل کنیم؟
قابلیت drag هم نداشت مهم نیست . یک tree ثابت از دیتا بیس پر می‌شود و بعد گزینه هایی که تیک دارد را در جدولی دیگر ذخیره می‌کنیم

عنوان: **مشکل اعتبار سنجی jQuery validator در Bootstrap tabs**
نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۰۹/۲۴ ۹:۱۰
آدرس: www.dotnettips.info
گروه‌ها: MVC, jQuery, Validation, Twitter Bootstrap

با استفاده از چهارچوب بوت استرپ می‌توان رابطهای کاربر استانداردی ساخت که قبلا دوستان در این سایت مطالبی را در این باب نوشته اند.
در مطلب صفحات مودال در بوت استرپ 3 در مورد ترکیب قالب بوت استرپ با سیستم اعتبارسنجی MVC و jQuery validation و نمایش فرم‌های مودال بوت استرپ صحبت شده و بسیار کامل هست.

مشکل:
هنگام کار با بوت استرپ اگر از tabهای آن در فرم برنامه استفاده کنید، هنگام validate کردن فرم متوجه می‌شوید که **فقط** کنترل‌های تب جاری اعتبارسنجی میشوند و بدون توجه به سایر کنترل هایی که در تب‌های دیگر هستند و احیانا در وضعیت عدم اعتبار هستند، فرم اعتبارسنجی و کامل اعلام شده و اطلاعات ارسال می‌شود.
دلیل:
دلیل این اتفاق این هست که jQuery validation به طور پیش فرض کنترل‌هایی که در صفحه مخفی هستند را اعتبارسنجی نمی‌کند و نادیده می‌گیرد.

راه حل:

با تغییر رفتار پیش فرض سیستم اعتبارسنجی می‌شود این مساله را حل کرد. با اضافه کردن **ignore: ""** اعلام می‌شود که کنترل‌های مخفی هم اعتبارسنجی شوند.
در نهایت کد کامل ترکیب قالب بوت استرپ با سیستم اعتبارسنجی جی کوئری به صورت زیر می‌شود. (فقط همان **ignore: ""** اضافه شده است).

```
function enableBootstrapStyleValidation() {
    $.validator.setDefaults({
        ignore: "",
        highlight: function (element, errorClass, validClass) {
            if (element.type === 'radio') {
                this.findByName(element.name).addClass(errorClass).removeClass(validClass);
            } else {
                $(element).addClass(errorClass).removeClass(validClass);
                $(element).closest('.form-group').removeClass('has-success').addClass('has-error');
            }
            $(element).trigger('highlited');
        },
        unhighlight: function (element, errorClass, validClass) {
            if (element.type === 'radio') {
                this.findByName(element.name).removeClass(errorClass).addClass(validClass);
            } else {
                $(element).removeClass(errorClass).addClass(validClass);
                $(element).closest('.form-group').removeClass('has-error').addClass('has-success');
            }
            $(element).trigger('unhighlited');
        }
    });
}
```

البته می‌توان این تغییر رفتار پیش فرض را فقط به فرم خاصی هم اعمال کرد. به این صورت

```
$("#form").validate({
...
rules: {...},
messages: {...},
ignore: "",
...
});
```

نظرات خوانندگان

نویسنده: م راد
تاریخ: ۱۳۹۲/۱۰/۰۷ ۱۳:۳۸

با تشکر از ارسال مطلب مفیدتون،
اما موضوعی که پیش میاد اینکه چطوری به ازای هر تب ابتدا اعتبارسنجی فیلدهای اون تب انجام شه و بعد تب بعدی نمایش داده شه؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۰۷ ۱۹:۳۳

خوب این همون حالت پیش فرضی هست که توضیح دادن: «هنگام validate کردن فرم متوجه می‌شوید که **فقط** کنترل‌های تب جاری اعتبارسنجی میشوند و بدون توجه به سایر کنترل هایی که در تب‌های دیگر هستند». این tab جاری منظور همون به ازای هر tab جداگانه هست.

برای نمایش خودکار تب بعدی هم باید کدنویسی کنید. فقط کافی هست که کلاس active رو به div تب مورد نظر اضافه کنید:

```
$("#home").removeClass("active");  // this deactivates the home tab
$("#profile").addClass("active");  // this activates the profile tab
```

عنوان: **استفاده از pjax بجای ajax در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۳ ۱۲:۴۵
آدرس: www.dotnettips.info
گروه‌ها: MVC, jQuery, jQuery Ajax, SEO

عموما از ajax برای ارائه سایت‌هایی سریع، با حداقل ریفرش و حداقل مصرف پهنای باند سرور، استفاده می‌شود. اما این روش، مشکلات خاص خود را نیز دارا است. عموما محتوای پویای بارگذاری شده، سبب تغییر آدرس صفحه‌ی جاری در مرورگر نمی‌شود. برای مثال اگر قرار است چندین برگه در صفحه به صورت ajax ایی بارگذاری شوند، تغییر سریع محتوا را مشاهده می‌کنید، اما خبری از تغییر آدرس جاری صفحه در مرورگر نیست. همچنین روش‌های ajax ایی عموما SEO friendly نیستند. زیرا اکثر موتورهای جستجو فاقد پردازشگرهای جاوا اسکریپت می‌باشند و محتوای پویای ajax ایی را مشاهده نمی‌کنند. برای آدرس دهی این مشکلات مهم، افزونه‌ای به نام pjax طراحی شده‌است که کار آن دریافت محتوای HTML ایی از سرور و قرار دادن آن در یک جایگاه خاص مانند یک div است. در پشت صحنه‌ی آن از jQuery ajax استفاده شده، به همراه push state

pjax = pushState + AJAX

Push state API همان HTML5 History API است؛ به این معنا که هرچند محتوای صفحه‌ی جاری به صورت پویا بارگذاری می‌شود، اما آدرس مرورگر نیز به صورت خودکار تنظیم خواهد شد؛ به همراه عنوان صفحه. به علاوه تاریخچه‌ی مرور صفحات نیز در مرورگر به روز رسانی شده و امکان حرکت بین صفحات توسط دکمه‌های back و forward همانند قبل وجود خواهد داشت.
همچنین اگر مرورگر جاری سایت، امکان استفاده از جاوا اسکریپت را نداشته باشد، به صورت خودکار به حالت بارگذاری کامل صفحه سوئیچ خواهد کرد.
سایت‌های بسیاری خودشان را با این الگو وفق داده‌اند. برای نمونه Twitter و Github از مفهوم pjax استفاده‌ی وسیعی دارند.
برای نمونه، layout یا master page یک سایت را درنظر بگیرید. به ازای مرور هر صفحه، یکبار باید تمام قسمت‌های تکراری layout از سرور بارگذاری شوند. توسط pjax به سرور اعلام می‌کنیم، ما تنها نیاز به body صفحات را داریم و نه کل صفحه را.
همچنین اگر مرورگر از جاوا اسکریپت استفاده نمی‌کند، لطفا کل صفحه را همانند گذشته بازگشت بده. به علاوه مسایل سمت کلاینت مانند تغییر آدرس مرورگر و تغییر عنوان صفحه نیز به صورت خودکار مدیریت شوند. این تکنیک را دقیقا در حین مرور مخزن‌های کد Github می‌توانید مشاهده کنید. فقط قسمتی که لیست فایل‌ها را ارائه می‌دهد، از سرور دریافت می‌گردد و نه کل صفحه.

بکارگیری pjax در ASP.NET MVC

مطابق توضیحاتی که ارائه شد، برای پیاده سازی سازی pjax نیاز به دو فایل layout داریم. یکی برای حالت ajax ایی و دیگری برای حالت بارگذاری کامل صفحه. حالت ajax ایی آن تنها از رندرکردن body پشتیبانی می‌کند؛ و نه ارائه تمام قسمت‌های صفحه مانند هدر، فوتر، منوها و غیره. بنابراین خواهیم داشت:

الف) تعریف فایل‌های layout سازگار با pjax

ابتدا یک فایل جدید را به نام _PjaxLayout.cshtml به پوشه‌ی Shared اضافه کنید؛ با این محتوا:

```
<title>@ViewBag.Title</title>
@RenderBody()
```

سپس layout اصلی سایت را به نحو ذیل تغییر دهید

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>
    <link href="~/Content/Site.css" rel="stylesheet" />
    <script src="~/Scripts/jquery-1.8.2.min.js"></script>
    <script src="~/Scripts/jquery.pjax.js"></script>
```

```
    <script type="text/javascript">
        $(function () {
            $(document).pjax('a[withpjax]', '#pjaxContainer', { timeout: 5000 });
        });
    </script>
</head>
    <body>
        <div>Main layout ...</div>
        <div id="pjaxContainer">
            @RenderBody()
        </div>
    </body>
</html>
```

در فایل PjaxLayout خبری از هدر و فوتر نیست و فقط یک عنوان و نمایش body را به همراه دارد.
فایل layout اصلی سایت همانند قبل است. فقط RenderBody آن داخل یک div با id مساوی pjaxContainer قرار گرفته و از آن در فراخوانی افزونه‌ی pjax استفاده شده‌است. همانطور که ملاحظه می‌کنید، مطابق تنظیمات ابتدای هدر layout، فقط لینک‌هایی که دارای ویژگی withpjax باشند، توسط pjax پردازش خواهند شد.

ب) تغییر فایل ViewStart برنامه

در فایل ViewStart، کار مقدار دهی layout پیش فرض صورت گرفته‌است. اکنون نیاز است این فایل را جهت معرفی layout دوم تعریف شده مخصوص pjax، اندکی ویرایش کنیم:

```
@{
    if (Request.Headers["X-PJAX"] != null)
    {
        Layout = "~/Views/Shared/_PjaxLayout.cshtml";
    }
    else
    {
        Layout = "~/Views/Shared/_Layout.cshtml";
    }
}
```

افزونه‌ی pjax، هدری را به نام X-PJAX به سرور ارسال می‌کند. بر این اساس می‌توان تصمیم گرفت که آیا از layout اصلی (در صورتیکه مرورگر از جاوا اسکریپت پشتیبانی نمی‌کند و این هدر را ارسال نکرده‌است) یا از layout سبک‌تر pjax استفاده شود.

ج) آزمایش برنامه

```
using System.Web.Mvc;

namespace PajxMvcApp.Controllers
{
    public class HomeController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult About()
        {
            return View();
        }
    }
}
```

یک کنترلر ساده را به نحو فوق با دو اکشن متد و دو View متناظر با آن ایجاد کنید.
سپس View متد Index را به نحو ذیل تغییر دهید:

```
@{
ViewBag.Title = "Index";
}
```

```
<h2>Index</h2>

@Html.ActionLink(linkText: "About", actionName:"About", routeValues: null,
                         controllerName:"Home", htmlAttributes: new { withpjax = "with-pjax"})
```

در این View یک لینک معمولی به اکشن متد About اضافه شده‌است. فقط در ویژگی‌های html آن، یک ویژگی جدید به نام withpjax را نیز اضافه کرده‌ایم تا در صورت امکان و پشتیبانی مرورگر، از pjax استفاده شود.
اکنون اگر برنامه را اجرا کنید، چنین خروجی را در برگه‌ی network آن مشاهده خواهید کرد:

همانطور که ملاحظه می‌کنید، با کلیک بر روی لینک About، یک درخواست pjax ایی به سرور ارسال شده‌است؛ به همراه هدرهای ویژه آن. هنوز قسمت‌های اصلی layout سایت مشخص هستند (و مجددا از سرور درخواست نشده‌اند). آدرس صفحه عوض شده‌است. به علاوه قسمت body آن تنها تغییر کرده‌است.

این مثال را از اینجا نیز می‌توانید دریافت کنید

PajxMvcApp.zip

برای مطالعه بیشتر

A Faster Web With PJAX

Favour PJAX over dynamically loaded partial views

What is PJAX and why

Pjax.Mvc

Using pjax with ASP.Net MVC3

Getting started with PJAX with ASP.NET MVC

ASP.NET MVC with PAjax or PushState/ReplaceState and Ajax

نظرات خوانندگان

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۳/۰۳ ۱۶:۲۳

با سلام؛ آیا pjax توسط کلیه مرورگرها پشتیبانی میشه؟ مخصوصا IE ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۳ ۱۷:۱

- فقط در مرورگرهایی پشتیبانی می‌شود که push state را پیاده سازی کرده باشند: لیست کامل آن‌ها
- اگر مرورگری history.pushState API را پشتیبانی نکند، بارگذاری صفحات آن معمولی خواهند بود (شبیه به حالت بارگذاری کامل برای موتورهای جستجو؛ بدون از کار افتادن برنامه).

نویسنده: MD
تاریخ: ۱۳۹۳/۰۳/۰۴ ۲۳:۵۶

با سلام
آیا در Asp Web Form هم قابل استفاده است؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۵ ۰:۲۶

در مثالی که توضیح داده شد، محتوای قسمتی از صفحه به صورت پویا با محتوای صفحه‌ای دیگر جایگزین می‌شود.
صفحه‌ی اول با master page کامل سایت رندر می‌شود. با کلیک بر روی لینک مشاهده‌ی صفحه‌ی بعدی، فقط محتوای آن صفحه (بدون master page اصلی سایت؛ شکل دوم) بجای div محتوای صفحه‌ی اول تزریق می‌شود.
اگر صفحه‌ی دوم به صورت معمولی درخواست شود، با master page کامل سایت رندر خواهد شد.
اما ... در وب فرم‌ها هر چند امکان انتخاب master page به صورت پویا وجود دارد ، اما به علت اینکه هر صفحه View State خاص خودش را خواهد داشت (بر اساس کنترل‌هایی که دارد)، تزریق محتوای آن داخل یک صفحه‌ی دیگر سبب تخریب View state جاری و از پیش موجود می‌شود. در نتیجه امکان ارسال اطلاعات به سرور را با پیام view state is corrupted از دست خواهید داد.

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۳/۰۵ ۱۳:۵

با سلام؛ تمام مراحل رو انجام دادم اما جواب نمیده. مشکل از کد زیر که نیست؟

```
@Html.ActionLink(item.Name, item.ActionName, item.ControllerName,
 new { Id = item.Id, area = item.AreaName }, new { @class = "drop", withpjax = "with-pjax" })
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۵ ۱۳:۱۳

- مثال امتحان شده، در انتهای بحث پیوست شده.
- برای دیباگ و خطایابی برنامه‌های جاوااسکریپتی، به این مطلب مراجعه کنید: « نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery »

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۳/۰۵ ۱۴:۴۳

با سلام؛ زمانی که یک ویو رو از طریق pjax نمایش میدم اگه داخل اون ویو یک لینک وجود داشته باشد که از طریق متد post یک اکشن رو فراخوانی می‌کند لینک کار نمی‌کند. مثلا در پروژه IRIS توی بعضی صفحات که دکمه امتیاز دهی وجود داره اگه این صفحات با pjax لود بشن دیگه دکمه‌ها کار نمی‌کنن. به نظر شما مشکل کجاست؟ آیا اسکریپت‌ها لود نمیشن؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۵ ۱۵:۱

- المان‌هایی که به صورت پویا به صفحه اضافه می‌شوند، تحت کنترل مجدد jQuery نخواهند بود، مگر اینکه از متدهای live (منسوخ شده) و یا on استفاده شود. مطابق مستندات این کتابخانه (انتهای صفحه)، متد on به صورت پیش فرض، برای کارکرد مجدد pjax، اعمال می‌شود. بنابراین فقط باید بررسی کنید که آیا نگارش jQuery مورد استفاده، از متد on پشتیبانی می‌کند یا خیر (از آخرین نگارش آن استفاده کنید).
- همچنین تمام کدهای سایر قسمت‌های برنامه هم مانند دکمه امتیازدهی که اشاره شد، باید تغییر کرده و از متد on استفاده کنند. مثلا اگر click دارند، باید بشوند on click.

```
//The new api
$(document).pjax('a[withpjax]', '#pjax-container')

//Which is roughly the same as
$(document).on('click', 'a[withpjax]', function(event) {
  $.pjax.click(event, '#pjaxContainer')
})
```

نویسنده: مهدی
تاریخ: ۱۳۹۳/۰۳/۱۶ ۲۱:۱۲

همه چیز خوبه؛ اما وقتی توی گرید میاییم میزنیم Edit و میریم صفحه Edit و گزینه Save رو میزنیم، برمیگردیم به صفحه Index در حالی که لینک تغییر نکرده و لینک مثلاً اینطوری مونده

www.test.com/edit/1

و وقتی میزنیم گرید به صفحه 2 بره ارور میده که اصلاً همچین صفحه ای وجود نداره

حالا راهی نیست که از Controller که کد زدیم برای Edit و در آخر گفتیم بره به صفحه Index این Pjax و اینا رو آخر لینک ننویسه؟

localhost:18162/Tbl1
Application name
Tbl1
Home
About
Contact
List of Tbl1
صفحه Index
Create
DeleteSelected
ExportView
Print
Drag a column header and drop it here to group by that column
Edit/Detail/Delete
Name
Family
Age
Salary
Coment
Edit
Details
Delete
a
444
12
Page 1 of 1
10 items per page
1 - 1 of 1 items

localhost:18162/Tbl1/Edit/2080
Application name
Tbl1
Home
About
Contact
صفحه
Edit
Edit
tbl1
Name
a
Family
444
Age
12
Salary
Coment
Save
Back to List

localhost:18162/Tbl1/Edit/2080
Application name
Tbl1
Home
About
Contact
List of Tbl1
بازگشت به صفحه
Index
پس از
Edit
و عدم تغییر لینک
Create
DeleteSelected
ExportView
Print
Drag a column header and drop it here to group by that column
Edit/Detail/Delete
Name
Family
Age
Salary
Edit
Details
Delete
a
444
12
Page 1 of 1
10 items per page
1 - 1 of 1 items

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۶ ۲۱:۲۵

در مثالی که زده شد، فقط لینک‌هایی که دارای ویژگی withpjax هستند تحت کنترل این افزونه قرار می‌گیرند و نه هیچ لینک دیگری در برنامه و نه هیچ روش بازگشت دیگری:

```
$(document).pjax('a[withpjax]', '#pjaxContainer', { timeout: 5000 });
```

البته این یک مثال است و اگر مثلا withpjax آن‌را حذف کنید:

```
$(document).pjax('a', '#pjaxContainer', { timeout: 5000 });
```

تمام لینک‌های صفحه تحت کنترل خواهند بود. شبیه به حالتی که عنوان کردید صفحه بندی گرید به هم خورده. برای اینکه این نوع تداخل‌ها رخ ندهند و هر لینکی در صفحه توسط این افزونه پردازش نشود، بهتر است از روش پیشنهادی استفاده کنید.

نویسنده: رمزینه
تاریخ: ۱۳۹۳/۰۳/۱۶ ۲۱:۳۸

نمی دونم Pjax این مشکلا رو چطوری بوجود میاره، اما از وقتی ازش استفاده کردم، توی Save کردن در صورتی که از گزینه‌های خود یک Textbox که خود Browser پیشنهاد میده استفاده بشه، چندتا چندتا ذخیره میشه:

و اینم خروجی یک Save

نویسنده: رمزینه
تاریخ: ۱۳۹۳/۰۳/۱۶ ۲۱:۴۳

ممنون از جوابتون اما مشکل من همینجاست که اصلاً به هیچ خصوصیتی ندادم

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

اینو گذاشتم اما چون توی

```
@Ajax.beginform(...)
```

هست و توی

```
<div id="pjaxContainer">
  @RenderBody()
</div>
```

که توی layout هست داره بصورت pjax عمل میکنه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۶ ۲۲:۲

- لازم هست با نحوه‌ی دیباگ برنامه‌های Ajax و برنامه‌های مبتنی بر jQuery آشنا شوید (جهت بررسی اینکه مشخص شود بدون تنظیمات این افزونه، آیا عملیات انجام شده، Ajax ایی است یا Pjax ایی). اطلاعات بیشتر
- کار pjax فقط ارائه محتوای صفحات است. اگر فعال هم نباشد، برنامه بدون مشکل کار می‌کند و صفحات آن نمایش داده خواهند شد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۶ ۲۲:۱۳

با عکس نمی‌شود یک برنامه را از راه دور دیباگ کرد .

نویسنده: احسان شیروان
تاریخ: ۱۳۹۳/۰۳/۱۸ ۱۱:۳۹

تصور بنده بر اینه که شما باید برای داشتن فرم همون Html.BeginForm معمولی رو استفاده کنید و نه Ajax.BeginForm چون به صورت توکار pjax داره از Ajax بهره می‌بره ممکنه یه تداخلی هم این وسط پیش بیاد

نویسنده: مهدی
تاریخ: ۱۳۹۳/۰۳/۱۹ ۰:۵۳

مشکل همینجاست فکر کنم که از beginForm چه عادی چه Ajax ای استفاده میشه
در حالت Ajax که مشکل اساسی هست و چندتا چندتا INsert میشه و یه جورایی همینطور تو حلقه میمونه
از Html.beginForm استفاده میکنم رفرش میشه
از هیچکدوم استفاده نکنم نمیدونم چطوری اطلاعات صفحه رو به یه Action توی کنترلر با دکمه Submit ارسال کنم!

نویسنده: فخاری
تاریخ: ۱۳۹۳/۰۷/۱۹ ۱۷:۱۸

با سلام
در ajax.actionlink میشد مثلا یه loading نمایش داد خواستم ببینم در pjax هم این امکان وجود داره؟

با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۱۹ ۱۷:۴۱

به مستندات آن مراجعه کنید (^). قسمت رویدادهای آن یک مثال loading دارد:

```
$(document).on('pjax:send', function() {
  $('#loading').show()
})
$(document).on('pjax:complete', function() {
  $('#loading').hide()
})
```

نویسنده: فخاری
تاریخ: ۱۳۹۳/۰۷/۲۰ ۱۲:۲۰

با سلام؛ وقتی که از Pjax در برنامه استفاده می‌شود و مثلا فرم لاگین را با آن اجرا می‌کنیم اگر از دستور

```
@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}
```

در ویو استفاده کنیم دوبار به اکشن login می‌رود که در دفعه اول به صورت Pjax برای ما فرم را می‌آورد ولی در دفعه دوم به صورت معمولی
در مثال زیر این مورد مفصلا مطرح شده
دانلود مثال
خواستم ببینم که مشکل از چی می‌تونه باشه؟

نویسنده: فخاری
تاریخ: ۱۳۹۳/۰۸/۰۵ ۱۰:۴۰

با سلام
زمانی که از pjax استفاده می‌کنیم اگر قرار باشه ویویی رو که دارای یک فرم برای ارسال اطلاعات به سرور است را نمایش دهیم با خطای زیر رو به رو می‌شویم.

و بعد از نمایش خطا کل صفحه رفرش می‌شود و مثل اینه که اصلا pjax کار نداده.
همه‌ی مراحل را هم در مثال قبل که ارسال شد انجام داده ام ولی باز به اینصورت جواب میده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۵ ۱۰:۴۵

صفحه‌ی کامل internal error در همینجا قابل مشاهده است؛ به همراه ریز خطای آن. روی + آن کلیک کنید و بعد محتوای برگه‌ی response را جهت یافتن متن استثنای حاصل بررسی کنید.

نویسنده: فخاری
تاریخ: ۱۳۹۳/۰۸/۰۵ ۱۱:۶

این خطا برای یه لحظه به من نمایش داده می‌شود و بعد کل خطاها پاک می‌شود و از نو صفحه بارگذاری می‌شود.
راهی هست که در زمان نمایش خطا به این صورت که سریع پاک می‌شود به اطلاعات آن دسترسی داشت؟

و اینکه اصلا این عمل صحیح است که جاهایی که قراره اطلاعات رو از کاربر دریافت کنیم از Pjax استفاده شود؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۵ ۱۱:۲۵

internal server error صرفا به معنای بروز استثنایی در کدهای سمت سرور شما است. برای لاگ کردن دقیق ریز جزئیات آن‌ها از ELAMH استفاده کنید. نسخه‌ی ساده شده‌ی آن برای ASP.NET MVC در اینجا

نویسنده: فخاری
تاریخ: ۱۳۹۳/۰۸/۰۶ ۱۲:۱

با سلام
از کمک شما ممنون
بالاخره خطا رو پیدا کردم
The following sections have been defined but have not been rendered for the layout page
""~/Views/Shared/_PjaxLayout.cshtml": "Scripts.
ولی دلیلش چی می‌تونه باشه مگه فقط نمیاد قسمت مثلا main در کد زیر را جایگذاری کنه؟

```
<div id="main">
            @RenderBody()
        </div>
```

```
//**********
@Scripts.Render("~/bundles/jquery")
```

```
@Scripts.Render("~/bundles/bootstrap")
```

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

و برای فراخوانی لینک‌های pjax نوشته شده:

```
<script type="text/javascript">
        $(function () {
            $(document).pjax('a[withpjax]', '#main', { timeout: 5000 });
```

و لینک هم به اینصورت:

```
@Html.ActionLink("ارتباط با ما","Contact", "Home"
                                        , null,new { withpjax="with-pjax" })
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۶ ۱۲:۲۹

یعنی فایل _PjaxLayout.cshtml هم نیاز به یک سری تعاریف section را دارد؛ مانند:

```
@RenderSection("Scripts", false)
```

کلا View ایی که قرار است رندر شود، اگر دارای تعاریف section اختصاصی هست، باید معادل آن‌ها در فایل layout متناظر، تعریف RenderSection وجود داشته باشد.

نویسنده: شقایق اشتری
تاریخ: ۱۳۹۳/۰۸/۱۹ ۱۲:۱۳

با سلام.
من وقتی از Pjax در یک View که هیچ Layout یی ندارد استفاده می‌کنم به خوبی جواب می‌دهد و صفحه رفرش نمی‌شود .(قسمت پیج بندی یک لیست از اطلاعات)
اما وقتی View را در یک Layout قرار می‌دهم صفحه رفرش می‌شود. جای اسکریپت‌ها را در view و Layout تغییر دادم هیچ تاثیری نداشت . مشکل کار من کجاست ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۹ ۱۲:۲۳

مراجعه کنید به توضیحات قسمت «ب) تغییر فایل ViewStart برنامه ». اگر Layout یک View ذکر نشود، اطلاعات آن‌را از ViewStart دریافت می‌کند. اگر آن‌را صریحا ذکر کنید، همان کاری که در ViewStart برای تشخیص هدر Request.Headers["X-"] PJAX انجام شده، در این حالت باید به صورت دستی انجام و اضافه شود.

نویسنده: شقایق اشتری
تاریخ: ۱۳۹۳/۰۸/۱۹ ۱۲:۳۵

من Layout را در View به این شکل مقدار دهی می‌کنم : Layout="" . که در این حالت pjax به خوبی جواب می‌دهد. البته من اون دو خط اسکریپت فایل‌های js و اون تکه کد را در view قرار دادم. من فقط از pjax برای پیج بندی لیست اطلاعاتم فقط در این view میخوام استفاده کنم.
اما من میخوام وقتی view خود را در Layout میگذارم جواب دهد.

لیست محصولات من در یک Partial قرار دارد که این Partial هم در View هست .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۹ ۱۳:۵۴

برای دیباگ کار، بررسی کنید :
- آیا لینک‌هایی که بر روی آن‌ها کلیک می‌شود، ویژگی withpjax را دارند؟ آیا اسکریپت متناظر با آن به صفحه پیوست شده؟
- آیا خطایی مشاهده نمی‌شود ؟
- آیا header مربوط به X-PJAX در درخواست ارسالی به سرور مطابق تصاویر فوق وجود دارند؟
- آیا در سمت سرور بر اساس هدر X-PJAX دریافتی، فایل layout صحیحی تنظیم می‌شود؟

عنوان: استفاده از افزونه‌ی jsTree در ASP.NET MVC
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۹ ۷:۵
آدرس: www.dotnettips.info
گروه‌ها: MVC, jQuery

jsTree یکی از افزونه‌های بسیار محبوب jQuery جهت نمایش ساختارهای سلسله مراتبی، خود ارجاع دهنده و تو در تو است. روش ابتدایی استفاده از آن تعریف یک سری ul و li ثابت در صفحه و سپس فراخوانی این افزونه بر روی آن‌ها است که سبب نمایش درخت‌واره‌ای این اطلاعات خواهد شد. روش پیشرفته‌تر آن به همراه کار با داده‌های JSON و دریافت پویای اطلاعات از سرور است که در ادامه به بررسی آن خواهیم پرداخت.

دریافت افزونه‌ی jsTree

برای دریافت افزونه‌ی jsTree می‌توان به مخزن کد آن در Github مراجعه کرد و همچنین مستندات آن‌را در سایت jstree.com قابل مطالعه هستند.

تنظیمات مقدماتی jsTree

در این مطلب فرض شده‌است که فایل jstree.min.js، در پوشه‌ی Scripts و فایل‌های CSS آن در پوشه‌ی Content\themes\default کپی شده‌اند.

به این ترتیب layout برنامه چنین شکلی را خواهد یافت:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>

    <link href="~/Content/Site.css" rel="stylesheet" />
    <link href="~/Content/themes/default/style.min.css" rel="stylesheet" />
    <script src="~/Scripts/jquery.min.js"></script>
    <script src="~/Scripts/jstree.min.js"></script>
</head>
<body dir="rtl">
    @RenderBody()

    @RenderSection("scripts", required: false)
</body>
</html>
```

نمایش راست به چپ اطلاعات

در کدهای این افزونه به تگ body و ویژگی dir آن برای تشخیص راست به چپ بودن محیط دقت می‌شود. به همین جهت این تعریف را در layout فوق ملاحظه می‌کنید. برای مثال اگر به فایل jstree.contextmenu.js (موجود در مجموعه سورس‌های این افزونه) مراجعه کنید، یک چنین تعریفی قابل مشاهده است:

```
right_to_left = $("body").css("direction") === "rtl";
```

تهیه ساختاری جهت ارائه‌ی خروجی JSON

با توجه به اینکه قصد داریم به صورت پویا با این افزونه کار کنیم، نیاز است بتوانیم ساختار سلسله مراتبی مدنظر را با فرمت JSON ارائه دهیم. در ادامه کلاس‌هایی که معادل فرمت JSON قابل قبول توسط این افزونه را تولید می‌کنند، ملاحظه می‌کنید:

```
using System.Collections.Generic;

namespace MvcJSTree.Models
{
    public class JsTreeNode
    {
        public string id { set; get; } // نام این خواص باید با مستندات هماهنگ باشد
        public string text { set; get; }
        public string icon { set; get; }
        public JsTreeNodeState state { set; get; }
        public List<JsTreeNode> children { set; get; }
        public JsTreeNodeLiAttributes li_attr { set; get; }
        public JsTreeNodeAAttributes a_attr { set; get; }

        public JsTreeNode()
        {
            state = new JsTreeNodeState();
            children = new List<JsTreeNode>();
            li_attr = new JsTreeNodeLiAttributes();
            a_attr = new JsTreeNodeAAttributes();
        }
    }

    public class JsTreeNodeAAttributes
    {
        // به هر تعداد و نام اختیاری می‌توان خاصیت تعریف کرد
        public string href { set; get; }
    }

    public class JsTreeNodeLiAttributes
    {
        // به هر تعداد و نام اختیاری می‌توان خاصیت تعریف کرد
        public string data { set; get; }
    }
```

```
    public class JsTreeNodeState
    {
        public bool opened { set; get; }
        public bool disabled { set; get; }
        public bool selected { set; get; }

        public JsTreeNodeState()
        {
            opened = true;
        }
    }
}
```

در اینجا به چند نکته باید دقت داشت:
- هر چند اسامی مانند a_attr، مطابق اصول نامگذاری دات نت نیستند، ولی این نام‌ها را تغییر ندهید. زیرا این افزونه دقیقا به همین نام‌ها و با همین املاء نیاز دارد .
- id، می‌تواند دقیقا معادل id یک رکورد در بانک اطلاعاتی باشد. Text عنوان گره‌ای (node) است که نمایش داده می‌شود. icon در اینجا مسیر یک فایل png است جهت نمایش در کنار عنوان هر گره. توسط state می‌توان مشخص کرد که زیر شاخه‌ی جاری به صورت باز نمایش داده شود یا بسته. به کمک خاصیت children می‌توان زیر شاخه‌ها را تا هر سطح و تعدادی که نیاز است تعریف نمود.
- خاصیت‌های li_attr و a_attr کاملا دلخواه هستند. برای مثال در اینجا دو خاصیت href و data را در کلاس‌های مرتبط با آن‌ها مشاهده می‌کنید. می‌توانید در اینجا به هر تعداد ویژگی سفارشی دیگری که جهت تعریف یک گره نیاز است، خاصیت اضافه کنید.

ساده‌ترین مثالی که از ساختار فوق می‌تواند استفاده کند، اکشن متد زیر است:

```
[HttpPost]
        public ActionResult GetTreeJson()
        {
            var nodesList = new List<JsTreeNode>();

            var rootNode = new JsTreeNode
            {
                id = "dir",
                text = "Root 1",
                icon = Url.Content("~/Content/images/tree_icon.png"),
                a_attr = { href = "http://www.bing.com" }
            };
            nodesList.Add(rootNode);

            nodesList.Add(new JsTreeNode
            {
                id = "test1",
                text = "Root 2",
                icon = Url.Content("~/Content/images/tree_icon.png"),
                a_attr = { href = "http://www.bing.com" }
            });

            return Json(nodesList, JsonRequestBehavior.AllowGet);
        }
```

در ابتدا لیست گره‌ها تعریف می‌شود و سپس برای نمونه در این مثال، دو گره تعریف شده‌اند و در ادامه با فرمت JSON در اختیار افزونه قرار گرفته‌اند.
بنابراین ساختارهای خود ارجاع دهنده را به خوبی می‌توان با این افزونه وفق داد.

فعال سازی اولیه سمت کلاینت افزونه jsTree

برای استفاده‌ی پویای از این افزونه در سمت کلاینت، فقط نیاز به یک DIV خالی است:

```
<div id="jstree">
</div>
```

سپس jstree را بر روی این DIV فراخوانی می‌کنیم:

```
$('#jstree').jstree({
                "core": {
                    "multiple": false,
                    "check_callback": true,
                    'data': {
                        'url': '@getTreeJsonUrl',
                        "type": "POST",
                        "dataType": "json",
                        "contentType": "application/json; charset=utf8",
                        'data': function (node) {
                            return { 'id': node.id };
                        }
                    },
                    'themes': {
                        'variant': 'small',
                        'stripes': true
                    }
                },
                "types": {
                    "default": {
                        "icon": '@Url.Content("~/Content/images/bookmark_book_open.png")'
                    },
                },
                "plugins": ["contextmenu", "dnd", "state", "types", "wholerow", "sort", "unique"],
                "contextmenu": {
                    "items": function (o, cb) {
                        var items = $.jstree.defaults.contextmenu.items();
                        items["create"].label = "ایجاد زیر شاخه";
                        items["rename"].label = "تغییر نام";
                        items["remove"].label = "حذف";
                        var cpp = items["ccp"];
                        cpp.label = "ویرایش";
                        var subMenu = cpp["submenu"];
                        subMenu["copy"].label = "کپی";
                        subMenu["paste"].label = "پیست";
                        subMenu["cut"].label = "برش";
                        return items;
                    }
                }
            });
```

توضیحات

- multiple : false به این معنا است که نمی‌خواهیم کاربر بتواند چندین گره را با نگه داشتن دکمه‌ی کنترل انتخاب کند.
- check_callback : true کدهای مرتبط با منوی کلیک سمت راست ماوس را فعال می‌کند.
- در قسمت data کار تبادل اطلاعات با سرور جهت دریافت فرمت JSON ایی که به آن اشاره شد، انجام می‌شود. متغیر getTreeJsonUrl یک چنین شکلی را می‌تواند داشته باشد:

```
@{
ViewBag.Title = "Demo";
var getTreeJsonUrl =  Url.Action(actionName: "GetTreeJson", controllerName: "Home");
}
```

- در قسمت themes مشخص کرده‌ایم که از قالب small آن به همراه نمایش یک درمیان پس زمینه‌ی روشن و خاکستری استفاده شود. قالب large نیز دارد.
- در قسمت types که مرتبط است با افزونه‌ای به همین نام، آیکن پیش فرض یک نود جدید ایجاد شده را مشخص کرده‌ایم.
- گزینه‌ی plugins، لیست افزونه‌های اختیاری این افزونه را مشخص می‌کند. برای مثال contextmenu منوی کلیک سمت راست ماوس را فعال می‌کند، dnd همان کشیدن و رها کردن گره‌ها است در زیر شاخه‌های مختلف. افزونه‌ی state، انتخاب جاری کاربر را در سمت کلاینت ذخیره و در مراجعه‌ی بعدی او بازیابی می‌کند. با ذکر افزونه‌ی wholerow سبب می‌شویم که انتخاب یک گره، معادل انتخاب یک ردیف کامل از صفحه باشد. افزونه‌ی sort کار مرتب سازی خودکار اعضای یک زیر شاخه را انجام می‌دهد. افزونه‌ی unique سبب می‌شود تا در یک زیر شاخه نتوان دو عنوان یکسان را تعریف کرد.
- در قسمت contextmenu نحوه‌ی بومی سازی گزینه‌های منوی کلیک راست ماوس را مشاهده می‌کنید. در حالت پیش فرض، عناوینی مانند rename، create و امثال آن نمایش داده می‌شوند که به نحو فوق می‌توان آن‌را تغییر داد.

با همین حد تنظیم، این افزونه کار نمایش سلسله مراتبی اطلاعات JSON ایی دریافت شده از سرور را انجام می‌دهد.

ذخیره سازی گره‌های جدید و تغییرات سلسله مراتب پویای تعریف شده در سمت سرور

همانطور که عنوان شد، اگر افزونه‌ی اختیاری contextmenu را فعال کنیم، امکان افزودن، ویرایش و حذف گره‌ها و زیر شاخه‌ها را خواهیم یافت. برای انتقال این تغییرات به سمت سرور، باید به نحو ذیل عمل کرد:

```
$('#jstree').jstree({
              // تمام تنظیمات مانند قبل
            }).on('delete_node.jstree', function (e, data) {
                })
                .on('create_node.jstree', function (e, data) {
                })
                .on('rename_node.jstree', function (e, data) {
                })
                .on('move_node.jstree', function (e, data) {
                })
                .on('copy_node.jstree', function (e, data) {
                })
                .on('changed.jstree', function (e, data) {
                })
                .on('dblclick.jstree', function (e) {
                })
                .on('select_node.jstree', function (e, data) {
                });
```

در اینجا نحوه‌ی تحت کنترل قرار دادن رخدادهای مختلف این افزونه را مشاهده می‌کنید. برای مثال در callback مرتبط با delete_node کار حذف یک گره اطلاع رسانی می‌شود. create_node مربوط است به ایجاد یک گره یا زیر شاخه‌ی جدید. rename_node پس از تغییر نام یک گره فراخوانی خواهد شد. move_node مربوط است به کشیدن و رها کردن یک گره در یک زیر شاخه‌ی دیگر. copy_node برای copy/paste یک گره تعریف شده‌است. Changed یک callback عمومی است. dblclick برای عکس العمل نشان دادن به رخداد دوبار کلیک کردن بر روی یک گره می‌تواند بکار گرفته شود. select_node با انتخاب یک گره فعال می‌شود.

در تمام این حالات، جایی که data در اختیار ما است، می‌توان یک چنین ساختار جاوا اسکریپتی را برای ارسال به سرور طراحی کرد:

```
function postJsTreeOperation(operation, data, onDone, onFail) {
            $.post('@doJsTreeOperationUrl',
                {
                    'operation': operation,
                    'id': data.node.id,
                    'parentId': data.node.parent,
                    'position': data.position,
                    'text': data.node.text,
                    'originalId': data.original ? data.original.id : data.node.original.id,
                    'href': data.node.a_attr.href
                })
                .done(function (result) {
                    onDone(result);
                })
                .fail(function (result) {
                    alert('failed.....');
                    onFail(result);
                });
        }
```

به این ترتیب در سمت سرور می‌توان id یک گره، متن تغییر یافته آن، والد گره و بسیاری از مشخصات دیگر را دریافت و ثبت کرد. پس از تعریف متد postJsTreeOperation فوق، آن‌را باید به callbackهایی که پیشتر معرفی شدند، اضافه کرد؛ برای مثال:

```
.on('create_node.jstree', function (e, data) {
                    postJsTreeOperation('CreateNode', data,
                        function (result) {
                            data.instance.set_id(data.node, result.id);
                        },
```

```
                        function (result) {
                            data.instance.refresh();
                        });
                })
```

در اینجا متد postJsTreeOperation، یک Operation خاص را مانند CreateNode (تعریف شده در enum ایی به نام JsTreeOperation در سمت سرور) به همراه data، به سرور post می‌کند.
و معادل سمت سرور دریافت کننده‌ی این اطلاعات، اکشن متد ذیل می‌تواند باشد:

```
[HttpPost]
        public ActionResult DoJsTreeOperation(JsTreeOperationData data)
        {
            switch (data.Operation)
            {
                case JsTreeOperation.CopyNode:
                case JsTreeOperation.CreateNode:
                    //todo: save data
                    var rnd = new Random(); // آی دی رکورد پس از ثبت در بانک اطلاعاتی دریافت و بازگشت
داده شود
                    return Json(new { id = rnd.Next() }, JsonRequestBehavior.AllowGet);

                case JsTreeOperation.DeleteNode:
                    //todo: save data
                    return Json(new { result = "ok" }, JsonRequestBehavior.AllowGet);

                case JsTreeOperation.MoveNode:
                    //todo: save data
                    return Json(new { result = "ok" }, JsonRequestBehavior.AllowGet);

                case JsTreeOperation.RenameNode:
                    //todo: save data
                    return Json(new { result = "ok" }, JsonRequestBehavior.AllowGet);

                default:
                    throw new InvalidOperationException(string.Format("{0} is not supported.",
data.Operation));
            }
        }
```

که در آن ساختار JsTreeOperationData به نحو ذیل تعریف شده‌است:

```
namespace MvcJSTree.Models
{
    public enum JsTreeOperation
    {
        DeleteNode,
        CreateNode,
        RenameNode,
        MoveNode,
        CopyNode
    }

    public class JsTreeOperationData
    {
        public JsTreeOperation Operation { set; get; }
        public string Id { set; get; }
        public string ParentId { set; get; }
        public string OriginalId { set; get; }
        public string Text { set; get; }
        public string Position { set; get; }
        public string Href { set; get; }
    }
}
```

این ساختار دقیقا با اعضای شیء جاوا اسکریپتی که متد postJsTreeOperation به سمت سرور ارسال می‌کند، تطابق دارد.
در اینجا Href را نیز مشاهده می‌کنید. همانطور که عنوان شد، اعضای JsTreeNodeAAttributes اختیاری هستند. بنابراین اگر این اعضاء را تغییر دادید، باید خواص JsTreeOperationData و همچنین اعضای شیء تعریف شده در postJsTreeOperation را نیز تغییر دهید تا با هم تطابق پیدا کنند.

چند نکته‌ی تکمیلی

اگر می‌خواهید که با دوبار کلیک بر روی یک گره، کاربر به href آن هدایت شود، می‌توان از کد ذیل استفاده کرد:

```
var selectedData;
                // ...
                 .on('dblclick.jstree', function (e) {
                     var href = selectedData.node.a_attr.href;
                     alert('selected node: ' + selectedData.node.text + ', href:' + href);

                     // auto redirect
                     if (href) {
                         window.location = href;
                     }

                     // activate edit mode
                     //var inst = $.jstree.reference(selectedData.node);
                     //inst.edit(selectedData.node);
                 })
                 .on('select_node.jstree', function (e, data) {
                     //alert('selected node: ' + data.node.text);
                     selectedData = data;
                 });
```

در callback مرتبط با select_node می‌توان به گره انتخابی درستی یافت. سپس می‌توان این گره را در callback متناظر با dblclick برای یافتن href و مقدار دهی window.location که معادل redirect سمت کاربر است، بکار برد.
حتی اگر خواستید که با دوبار کلیک بر روی یک گره، گزینه‌ی ویرایش آن فعال شود، کدهای آن را به صورت کامنت مشاهده می‌کنید.

مثال کامل این بحث را از اینجا می‌توانید دریافت کنید:

MvcJSTree.zip

نظرات خوانندگان

نویسنده: ناصر پورعلی
تاریخ: ۱۳۹۳/۰۵/۰۶ ۱۵:۵

با سلام

من دارم از همین jstree مثال شما استفاده میکنم.

و json زیر رو هم به سمت کلاینت برمیگردونم به صورت صحیح :


```
[{"id":"OrganizationTree","text":"ساختار
سازمانی","icon":"/Content/images/tree_icon.png","state":{"opened":true,"disabled":false,"selected":fals
e},"children":[{"id":"2","text":"آنات","icon":"/Content/images/nuclear.png","state":{"opened":true,"dis
abled":false,"selected":false},"children":[{"id":"4","text":"آموزش","icon":"/Content/images/nuclear.pn
g","state":{"opened":true,"disabled":false,"selected":false},"children":[],"li_attr":{"data":null},"a_a
ttr":{"href":null}},{"id":"5","text":"هیات
مدیره","icon":"/Content/images/nuclear.png","state":{"opened":true,"disabled":false,"selected":false},"
children":[],"li_attr":{"data":null},"a_attr":{"href":null}}],"li_attr":{"data":null},"a_attr":{"href":
null}},{"id":"1","text":"پرنیان","icon":"/Content/images/nuclear.png","state":{"opened":true,"disabled":
false,"selected":false},"children":[{"id":"1","text":"BPM","icon":"/Content/images/nuclear.png","state"
:{"opened":true,"disabled":false,"selected":false},"children":[],"li_attr":{"data":null},"a_attr":{"hre
f":null}},{"id":"3","text":"پشتیبانی","icon":"/Content/images/nuclear.png","state":{"opened":true,"disab
led":false,"selected":false},"children":[],"li_attr":{"data":null},"a_attr":{"href":null}},{"id":"2","t
ext":"فروش","icon":"/Content/images/nuclear.png","state":{"opened":true,"disabled":false,"selected":fa
lse},"children":[],"li_attr":{"data":null},"a_attr":{"href":null}}],"li_attr":{"data":null},"a_attr":{"
href":null}}],"li_attr":{"data":null},"a_attr":{"href":null}}]
```


منتهی همش خطای too much recursion میگیرم، در صورتی که حلقه ای رو هم نمی‌بینم که ایجاد شده باشه داخل رشته json تولید شده.

اینهم تنظیمات فراخوانی

```
<script>
            $(function () {
                $('#jstree').jstree({
                    "core": {
                        "multiple": true,
                        "check_callback": true,
                        'data': {
                            'url': '@getTreeJsonUrl',
                            "type": "POST",
                            "dataType": "json",
                            "contentType": "application/json; charset=utf8",
                            'data': function (node) {
                                return { 'id': node.id };
                            }
                        },
                        'themes': {
                            'variant': 'large',
                            'stripes': false
```

```
                        }
                    },
                    "types": {
                        "default": {
                            "icon": '@Url.Content("~/Content/images/bookmark_book_open.png")'
                        },
                    },
                    "plugins": ["contextmenu", "dnd", "state", "types", "checkbox", "wholerow", "sort",
"unique", "real_checkboxes"],
                    "contextmenu": {
                        "items": function (o, cb) {
                            var items = $.jstree.defaults.contextmenu.items();
                            items["create"].label = "ایجاد زیر شاخه";
                            items["rename"].label = "تغییر نام";
                            items["remove"].label = "حذف";
                            var cpp = items["ccp"];
                            cpp.label = "ویرایش";
                            var subMenu = cpp["submenu"];
                            subMenu["copy"].label = "کپی";
                            subMenu["paste"].label = "پیست";
                            subMenu["cut"].label = "برش";
                            return items;
                        }
                    }
                });
            });
        </script>
```

فکر میکنین مشکل از کجا باشه؟

جالب اینجاست که اگه فقط یه سطح پایین برم مشکلی نیست!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۰۷ ۲:۳۳

گره‌های تعریف شده unique ID ندارند. این unique ID در کل tree معنا پیدا می‌کند و الزاما ارتباطی به ID رکورد شما در یک جدول خاص بانک اطلاعاتی ندارد.

نویسنده: ناصر پورعلی
تاریخ: ۱۳۹۳/۰۵/۱۱ ۱۶:۱۳

ممنون، مشکل قبلی حل شد.
مشکل دیگه ام اینه که من مجبورم از jquery 1.6.1 استفاده کنم، آیا نسخه ای از jstree هست که با این نسخه jquery به خوبی کار کنه؟ ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۱۱ ۱۸:۲۸

از jQuery>=1.9 تعدادی از متدهای قدیمی آن مانند live حذف شدند. راه حلی که برای آن وجود دارد استفاده از پروژه‌ای است به نام jQuery migrate . این پروژه متدهای حذف شده را بر اساس API جدید بازنویسی کرده. بنابراین افزونه‌های به روز نشده قدیمی، بدون مشکل با نگارش‌های جدید jQuery کار خواهند کرد.
استفاده از آن هم ساده‌است. تنها کاری که باید انجام دهید، تعریف آخرین نگارش jQuery و سپس افزودن jQuery migrate است:

```
<script src="jquery.js"></script>
<script src="jquery-migrate-1.2.1.js"></script>
```

نویسنده: سعیده
تاریخ: ۱۳۹۳/۰۷/۲۴ ۱۵:۴۶

با سلام و تشکر از آموزشتون.

من از jstree در یک صفحه‌ی html ای استفاده کردم اما دیتا رو از طریق webmethod و ajaxcall دریافت میکنم.
اما نمیدونم چطور باید دیتا رو به‌تری ویو bind کنم.شما برای درست بودن فرمت دیتا یک کلاس تعریف کردین و با اکشن متد دیتا رو بایند کردین...در حال حاضر دیتای من به صورت Id/Title/ParentId هستش. ممنون میشم اگر من رو راهنمایی کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۲۴ ۱۹:۰

برای دریافت JSON، این روزها دیگر کسی از فایل‌های asmx استفاده نمی‌کند. امکان استفاده از ASP.NET Web API با وب فرم‌ها هم وجود دارد. اطلاعات بیشتر
بعد از آن تنها کاری که باید انجام شود، بازگشت مستقیم خروجی GetTreeJson مثال فوق است و سایر مفاهیم آن یکی هست.

عنوان: **افزونه rtlColResizable جهت تغییر سایز ستون‌های جدول‌های راست به چپ HTML**
نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۳/۰۴/۲۰ ۱۱:۵
آدرس: www.dotnettips.info
گروه‌ها: jQuery, Persian

برای تغییر سایز ستون‌های جداول HTML با استفاده از ماوس، افزونه‌های زیادی تدارک دیده شده است که از جمله مطرح‌ترین آن‌ها می‌توان به colResizable اشاره کرد. حتی اگر از DataGridهای مطرح وب هم استفاده کرده باشید، اکثر آن‌ها از تغییر سایز ستون‌ها توسط کاربر پشتیبانی می‌کنند. اما مشکل بزرگی که در همه‌ی آن‌ها مشترک است این است که فقط از چیدمان‌های چپ به راست پشتیبانی می‌کنند و به محض اینکه شما ساختار راست به چپ را به جدول مورد نظر اعمال کنید، عملکرد تغییر سایز ستون‌ها با اشکال مواجه می‌شود.

به همین جهت برای تغییر سایز ستون‌ها توسط کاربر، افزونه ای برای jQuery تدارک دیدم که با جداول معمول HTML که همان tableها هستند سازگار است و به راحتی به هر جدولی می‌توان آن را اعمال کرد. **کدهای افزونه‌ی rtlColResizable:**

```
(function ($, undefined) {

    $.fn.extend({
        'rtlColResizable': function (options) {
            var defaults = {
                //Default values for the plugin's options here
            };

            options = $.extend(defaults, options);

            var isMouseButtonPressed;
            var $resizingElement = undefined;
            var resizingElementStartWidth;
            var mouseCursorStartX;
            var isCursorInResizingPosition;

            var addResizingCursorStyle = function ($element) {
                $element.css({
                    'cursor': 'col-resize',
                    'user-select': 'none',
                    '-o-user-select': 'none',
                    '-ms-user-select': 'none',
                    '-moz-user-select': 'none',
                    '-khtml-user-select': 'none',
                    '-webkit-user-select': 'none',
                });
            };

            var removeResizingCursorStyle = function ($element) {
                $element.css({
                    'cursor': 'default',
                    'user-select': 'text',
                    '-o-user-select': 'text',
                    '-ms-user-select': 'text',
                    '-moz-user-select': 'text',
                    '-khtml-user-select': 'text',
                    '-webkit-user-select': 'text',
                });
            };

            var canResize = function (e) {
                return (e.offsetX || e.clientX - $(e.target).offset().left) < 10;
            };

            return this.each(function () {

                var opts = options;
                var tableColumns = $(this).find('th');

                tableColumns.filter(':not(:last-child)').mousedown(function (e) {

                    $resizingElement = $(this);
                    isMouseButtonPressed = true;
                    mouseCursorStartX = e.pageX;
                    resizingElementStartWidth = $resizingElement.width();

                });
```

```
                tableColumns.mousemove(function (e) {

                    if (canResize(e)) {
                        addResizingCursorStyle($(e.target));
                        isCursorInResizingPosition = true;

                    } else if (!isMouseButtonPressed) {
                        removeResizingCursorStyle($(e.target));
                        isCursorInResizingPosition = false;
                    }

                    if (isCursorInResizingPosition && isMouseButtonPressed) {

                        $resizingElement.width(resizingElementStartWidth + (mouseCursorStartX -
e.pageX));
                    }

                });

                $(document).mouseup(function () {
                    if (isMouseButtonPressed) {
                        removeResizingCursorStyle($resizingElement);
                        isMouseButtonPressed = false;
                    }

                });

            });
        }
    });

})(jQuery);
```

نحوه‌ی استفاده:

این افزونه با تگ Table در HTML سازگار است. فقط تنها موردی که باید رعایت شود این است که در هنگام تعریف ساختار جدول، باید استاندارد تعریف ستون‌ها یا همان Headerها را رعایت کنید و از تگ‌های thead و th استفاده کنید.

نمونه ای از نحوه‌ی استفاده از آن را در کدهای زیر می‌بینید:

```
<!DOCTYPE html>
<html>
<head>
    <title>rtlColResizable Sample</title>
</head>
<body>
    <table id="myTable">
        <thead>
            <tr>
                <th>ستون1</th> <!-- نام ستون‌ها را حتما در این تگ باید تعریف کنید -->
                <th>ستون2</th>
                <th>ستون3</th>
            </tr>
        </thead>
        <tr>
            <td>داده مربوط به ستون 1</td>
            <td>داده مربوط به ستون 2 </td>
            <td>داده مربوط به ستون 3</td>
        </tr>
    </table>

    <script type="text/javascript" src="jquery-1.9.1.min.js"></script>
    <script type="text/javascript" src="jquery.rtlColResizable.js"></script>
    <script>

        $('table#myTable').rtlColResizable();

    </script>
</body>
</html>
```

دریافت نمونه کدی از نحوه‌ی استفاده از این افزونه

RTL-Table-Column-Resize-plugin.rar

عنوان: معرفی Kendo UI
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۴ ۱:۳۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Kendo UI

Kendo UI چیست؟

Kendo UI یک فریم ورک جاوا اسکریپتی ساخت برنامه‌های مدرن و تعاملی وب است و برای رسیدن به این مقصود، از HTML 5، CSS 3، JavaScript و jQuery کمک می‌گیرد.

امکانات فراهم شده توسط Kendo UI

1) انواع و اقسام ویجت‌ها: کنترل‌های وب تهیه شده برفراز jQuery
ویجت‌های آن در سه گروه کلی قرار می‌گیرند:
- گروه وب، مانند گرید، tree-view و غیره.
- گروه DataViz که جهت نمایش بصری اطلاعات و ترسیم انواع و اقسام نمودارها کاربرد دارد.
- گروه موبایل که با استفاده از فناوری adaptive rendering، در سیستم عامل‌های مختلف موبایل، مانند اندروید و آی او اس، ظاهری بومی و هماهنگ با آن‌ها را ارائه می‌دهد.

2) منبع داده سمت کاربر (Client side data source)
منبع داده سمت کاربر Kendo UI، از انواع و اقسام منابع داده محلی مانند آرایه‌های جاوا اسکریپتی تا منابع داده راه دور، مانند XML، JSON و JSONP، جهت نمایش اطلاعات و data binding پشتیبانی می‌کند. این منبع داده، مواردی مانند صفحه بندی، مرتب سازی اطلاعات و گروه بندی آن‌ها را نیز فراهم می‌کند. به علاوه با عملیات ثبت، ویرایش و حذف اطلاعات نیز هماهنگی کاملی را دارد.

3) به همراه یک فریم ورک MVVM توکار است
این فریم ورک MVVM مواردی مانند two way data binding و همچنین declarative binding را نیز پشتیبانی می‌کند.

4) امکان تعویض قالب
5) پویا نمایی، کشیدن و رها کردن
6) فریم ورک اعتبارسنجی

چرا Kendo UI؟

- مهم‌ترین مزیت کار با Kendo UI، فراهم آوردن تمام نیازهای توسعه‌ی یک برنامه‌ی مدرن وب، تنها در یک بسته است. به این ترتیب دیگر نیازی نیست تا گرید را از یک‌جا، tree-view را از جایی دیگر و کتابخانه‌های رسم نمودار را از منبعی ناهمگون با سایر عناصر برنامه دریافت و استفاده کنید؛ در اینجا تمام این‌ها در قالب یک بسته‌ی آماده برای شما فراهم شده‌است و همچنین با یکدیگر سازگاری کاملی دارند.
- تمام ویجت‌های آن برای نمایش سریع با کارآیی بالا طراحی شده‌اند.
- پشتیبانی خوب آن. این فریم ورک محصول شرکتی است که به صورت تخصصی کار تهیه کامپوننت‌های وب و دسکتاپ را انجام می‌دهد.

مرورگرهای پشتیبانی شده

یکی دیگر از مزایای مهم کار با Kendo UI پشتیبانی گسترده‌ی آن از اکثر مرورگرهای موجود است. این فریم ورک با مرورگرهای زیر سازگار است:
- IE 7 به بعد
- فایرفاکس 10 به بعد
- تمام نگارش‌های کروم
- اپرا 10 به بعد

- سفاری 4 به بعد

مجوز استفاده از Kendo UI

Kendo UI با سه مجوز ذیل ارائه می‌شود:

- 30 روزه آزمایشی رایگان
- تجاری
- سورس باز با مجوز Apache

پیشتر نسخه‌ی تجاری آن تحت مجوز GPL نیز در دسترس بود. اما اخیرا مجوز GPL آن حذف شده و به Apache تغییر یافته است. اما باید در نظر داشت که نسخه‌ی سورس باز آن شامل کنترل‌های مهمی مانند «گرید» نیست و این موارد تنها در نسخه‌ی تجاری آن لحاظ شده‌اند.

مثال‌های Kendo UI

پس از دریافت بسته‌ی کامل آن، پوشه‌هایی مانند styles، js و امثال آن قابل مشاهده هستند؛ به همراه پوشه‌ی examples آن که حداقل 86 پوشه‌ی دیگر در آن جهت ارائه مثال‌هایی از نحوه‌ی کاربرد المان‌های مختلف آن تدارک دیده شده‌اند.

نحوه‌ی افزودن Kendo UI به صفحه

از آنجائیکه Kendo UI یک فریم ورک جاوا اسکریپتی است، همانند سایر برنامه‌های وب، افزودن تعاریف فایل‌های js، css و تصاویر مرتبط با آن، برای شروع به کار کفایت می‌کند. برای این منظور ابتدا پوشه‌های js و styles بسته‌ی دریافتی آن‌را به برنامه‌ی خود اضافه کنید (این پوشه‌ها در فایل پیوست انتهای بحث موجود هستند).

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>

    <!--KendoUI: Web-->
    <link href="styles/kendo.common.min.css" rel="stylesheet" type="text/css" />
    <link href="styles/kendo.default.min.css" rel="stylesheet" type="text/css" />
    <script src="js/jquery.min.js" type="text/javascript"></script>
    <script src="js/kendo.web.min.js" type="text/javascript"></script>

    <!--KendoUI: DataViz-->
    <link href="styles/kendo.dataviz.min.css" rel="stylesheet" type="text/css" />
    <script src="js/kendo.dataviz.min.js" type="text/javascript"></script>

    <!--KendoUI: Mobile-->
    <link href="styles/kendo.mobile.all.min.css" rel="stylesheet" type="text/css" />
    <script src="js/kendo.mobile.min.js" type="text/javascript"></script>

    <script type="text/javascript">
        $(function() {
            $("#pickDate").kendoDatePicker();
        });
    </script>
</head>
<body>
    <span>
        Pick a date: <input id="pickDate" type="text"/>
    </span>
</body>
</html>
```

در اینجا یک مثال ساده‌ی استفاده از date picker کندو یو آی را ملاحظه می‌کنید. در قسمت head صفحه، نحوه‌ی ثبت سه گروه اسکریپت و شیوه نامه، مشخص شده‌اند. اگر نیاز به کامپوننت‌های وب آن‌را دارید باید اجزایی مانند kendo.common.min.css، jquery.min.js، kendo.default.min.css و kendo.web.min.js به صفحه اضافه شوند. اگر نیاز به رسم نمودار هست، فایل‌ها kendo.dataviz.min.css و kendo.dataviz.min.js باید تعریف شوند و برای فعال سازی اجزای موبایل آن فایل‌های kendo.mobile.all.min.css و kendo.mobile.min.js نیاز است به صفحه پیوست شوند. در هر سه حالت ذکر jquery.min.js الزامی است.

دریافت سورس کامل این قسمت که حاوی فایل‌های اصلی kendoui.professional.2014.2.1008 نیز می‌باشد:

KendoUI01.7z

نظرات خوانندگان

نویسنده: محمد
تاریخ: ۱۳۹۳/۰۸/۱۴ ۱۴:۳۸

برای دریافت پکیج کامل Kendo از این آدرس استفاده کنید.

نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۳/۰۸/۱۴ ۲۰:۵

آیا امکان استفاده از Razor Wrapper هم به صورت رایگان وجود دارد یا اینکه نیازمند تهیه بسته کامل می‌باشد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۴ ۲۰:۱۸

تجاری هست و توصیه هم نمی‌شود. چون نهایتا برای بسیاری از کارها باید به پشت صحنه‌ی این ویجت‌ها و امکانات مراجعه کنید؛ یعنی نیاز است مستقیما اسکریپت نویسی کنید و با ساختار واقعی آن‌ها آشنا باشید.

نویسنده: سعیدجلالی
تاریخ: ۱۳۹۳/۰۸/۱۷ ۸:۵۶

بله امکان استفاده از wrapper در نسخه asp.net.mvc.commercial وجود دارد
استفاده از اون هم خیلی ساده‌تر و خواناتر از جاوا اسکریپت هست. نظر آقای نصیری هم محترم است ولی در مواقع خواص میتونید همزمان هم از جاوااسکریپت استفاده کنید هم از wrapper یک نمونه رو در زیر با هم مقایسه میکنیم
با استفاده از جاوا اسکریپت

```
<input id="pickDate" type="text"/>

<script type="text/javascript">
  $(function() {
        $("#pickDate").kendoDatePicker();
  });
</script>
```

با استفاده از wrapper

```
@(Html.Kendo().DatePicker().Name("pickDate"))
```

در ضمن اینکه توی wrapper امکان استفاده از Intellisense و امکان تعریف ارتباط اغلب کامپوننت‌های وب به مدل با استفاده از forهای نمونه معادل کامپوننت فراهم شده است مانند wrapper زیر

```
@(Html.Kendo().DatePickerFor(m => m.HireDate).Name("pickDate1"))
```

نویسنده: Ara
تاریخ: ۱۳۹۳/۰۸/۱۷ ۱۹:۳۱

به دوستان پیشنهاد می‌شه این رو هم ببینید
استفاده از Kendo UI بهمراه AngularJs خیلی خوبه !
ما تو پروژه تجاری تجاری استفاده کردیم و خیلی راضی هستیم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۲۱ ۱۲:۲۹

نکته‌ای در مورد دریافت آخرین نگارش‌های Kendo UI

شماره نگارش‌های مختلف Kendo UI یک چنین شکلی را دارند: 2014.3.1119

برای دریافت فایل‌های js و css نگارشی خاص، از الگوی ذیل استفاده کنید:

```
http://cdn.kendostatic.com/version/js/file name.js
http://cdn.kendostatic.com/version/styles/file name.css
```

برای مثال:

```
http://cdn.kendostatic.com/2014.3.1119/js/kendo.all.min.js

http://cdn.kendostatic.com/2014.3.1119/styles/kendo.default.min.css
http://cdn.kendostatic.com/2014.3.1119/styles/kendo.common.min.css
http://cdn.kendostatic.com/2014.3.1119/styles/kendo.dataviz.default.min.css
http://cdn.kendostatic.com/2014.3.1119/styles/kendo.dataviz.min.css
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۲۱ ۱۴:۳۴

جهت اطلاع

مثال‌های این سری را از مخزن کد ذیل نیز می‌توانید دریافت کنید:

KendoUI-Samples

عنوان: بررسی ساختار ویجت‌های وب Kendo UI
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۴ ۱۴:۱۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Kendo UI

ویجت‌های وب Kendo UI کدامند؟

ویجت‌های وب Kendo UI مجموعه‌ای از کنترل‌های سفارشی HTML 5 هستند که برفراز jQuery تهیه شده‌اند. این کنترل‌ها برای برنامه‌های وب و همچنین برنامه‌های دسکتاپ لمسی طراحی شده‌اند.
بهترین روش برای مشاهده‌ی این مجموعه، مراجعه به فایل examples\index.html پوشه‌ی اصلی Kendo UI است که لیست کاملی از این ویجت‌ها را به همراه مثال‌های مرتبط ارائه می‌دهد. تعدادی از اعضای این مجموعه شامل کنترل‌های ذیل هستند:

Window, TreeView, Tooltip, ToolBar, TimePicker, TabStrip, Splitter, Sortable, Slider, Gantt, Scheduler, ProgressBar, PanelBar, NumericTextBox, Notification, MultiSelect, Menu, MaskedTextBox, ListView, PivotGrid, Grid, Editor, DropDownList, DateTimePicker, DatePicker, ComboBox, ColorPicker, Calendar, Button, AutoComplete

نحوه‌ی استفاده کلی از ویجت‌های وب Kendo UI

با توجه به اینکه کنترل‌های Kendo UI مبتنی بر jQuery هستند، نحوه‌ی استفاده از آن‌ها، مشابه سایر افزونه‌های جی‌کوئری است.
ابتدا المانی به صفحه اضافه می‌شود:

```
<input id="pickDate" type="text"/>
```

سپس این المان را در رویداد document ready، به یکی از کنترل‌های Kendo UI مزین خواهیم کرد. برای مثال تزئین یک TextBox معمولی با یک Date Picker:

```
<script type="text/javascript">
  $(function() {
        $("#pickDate").kendoDatePicker();
  });
</script>
```

روش دیگری به نام declarative initialization نیز برای اعمال ویجت‌های وب Kendo UI قابل استفاده است که از ویژگی‌های data-* مرتبط با HTML 5 کمک می‌گیرد. برای نمونه، کدهای جاوا اسکریپتی فوق را می‌توان با ویژگی data-role ذیل جایگزین کرد:

```
<input id="dateOfBirth" type="text" data-role="datepicker" />
```

اگر در این حالت برنامه را اجرا کنید، تفاوتی را مشاهده نخواهید کرد.
برای فعال سازی حالت declarative initialization باید به دو نکته‌ی مهم دقت داشت:
الف) در مطلب معرفی Kendo UI اسکریپت‌های ذیل برای آماده سازی Kendo Ui معرفی شدند:

```
<!--KendoUI: Web-->
<link href="styles/kendo.common.min.css" rel="stylesheet" type="text/css" />
<link href="styles/kendo.default.min.css" rel="stylesheet" type="text/css" />
<script src="js/jquery.min.js" type="text/javascript"></script>
<script src="js/kendo.web.min.js" type="text/javascript"></script>

<!--KendoUI: DataViz-->
<link href="styles/kendo.dataviz.min.css" rel="stylesheet" type="text/css" />
<script src="js/kendo.dataviz.min.js" type="text/javascript"></script>

<!--KendoUI: Mobile-->
<link href="styles/kendo.mobile.all.min.css" rel="stylesheet" type="text/css" />
<script src="js/kendo.mobile.min.js" type="text/javascript"></script>
```

باید دقت داشت که در آن واحد نمی‌توان تمام این بسته‌ها را با هم بکار برد؛ چون برای مثال فایل‌های جداگانه ویجت‌های وب و موبایل با هم تداخل ایجاد می‌کنند . بجای اینکار بهتر است از فایل‌های kendo.all.min.js (که حاوی تمام اسکریپت‌های لازم است) و cssهای عنوان شده استفاده کرد:

```
<link href="styles/kendo.common.min.css" rel="stylesheet" type="text/css" />
<link href="styles/kendo.default.min.css" rel="stylesheet" type="text/css" />
<script src="js/jquery.min.js" type="text/javascript"></script>
<script src="js/kendo.all.min.js" type="text/javascript"></script>
```

ب) data-roleها توسط متد kendo.init فعال می‌شوند.

یک مثال کامل:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>

    <link href="styles/kendo.common.min.css" rel="stylesheet" type="text/css" />
    <link href="styles/kendo.default.min.css" rel="stylesheet" type="text/css" />
    <script src="js/jquery.min.js" type="text/javascript"></script>
    <script src="js/kendo.all.min.js" type="text/javascript"></script>

    <script type="text/javascript">
        $(function () {
            $("#pickDate").kendoDatePicker();
        });

        $(function () {
            // initialize any widgets in the #container div
            kendo.init($("#container"));
        });
    </script>
</head>
<body>
    <span>
        Pick a date: <input id="pickDate" type="text" />
    </span>

    <div id="container">
        <input id="dateOfBirth" type="text" data-role="datepicker" />
        <div id="colors"
             data-role="colorpalette"
             data-columns="4"
             data-tile-size="{ width: 34, height: 19 }"></div>
    </div>
</body>
</html>
```

- در این مثال نحوه‌ی پیوست تمام فایل‌های لازم Kendo UI را به صورت یکجا ملاحظه می‌کنید که در ابتدای head صفحه ذکر شده‌اند.
- در اینجا pickDate به صورت معمولی فعال شده‌است.
- اما در قسمت kendo.init نام یک ناحیه یا نام یک کنترل را می‌توان ذکر کرد. برای مثال در اینجا کل ناحیه‌ی مشخص شده توسط یک div با id مساوی container به صورت یکجا با تمام کنترل‌های داخل آن فعال گردیده‌است.

بنابراین برای اعمال declarative initialization، یک ناحیه را توسط kendo.init مشخص کرده و سپس توسط data-roleها، نام ویجت وب مورد نظر را به صورت lower case مشخص می‌کنیم. همچنین فایل‌های اسکریپت مورد استفاده نیز نباید تداخلی داشته باشند.

تنظیمات ویجت‌های وب Kendo UI

تاکنون نمونه‌ی ساده‌ای از بکارگیری ویجت‌های وب Kendo UI را بررسی کردیم؛ اما این ویجت‌ها توسط تنظیمات پیش بینی شده برای آن‌ها بسیار قابل تنظیم و تغییر هستند. تنظیمات آن‌ها نیز بستگی به روش استفاده و آغاز آن‌ها دارد. برای مثال اگر این

ویجت‌ها را توسط کدهای جاوا اسکریپتی آغاز کرده‌اید، در همانجا توسط پارامترهای افزونه‌ی جی‌کوئری می‌توان تنظیمات مرتبط را اعمال کرد:

```
<script type="text/javascript">
  $(function () {
      $("#pickDate").kendoDatePicker({
              format: "yyyy/MM/dd"
        });
  });
</script>
```

که در اینجا توسط پارامتر format، نحوه‌ی دریافت تاریخ نهایی مشخص می‌شود.

در حالت declarative initialization، پارامتر format تبدیل به ویژگی data-format خواهد شد:

```
<input id="dateOfBirth" type="text"
          data-role="datepicker"
          data-format="yyyy/MM/dd" />
```

تنظیمات DataSource ویجت‌های وب

بسیاری از ویجت‌های وب Kendo UI با داده‌ها سر و کار دارند مانند Grid، Auto Complete، Combo box و غیره. این کنترل‌ها داده‌های خود را از طریق خاصیت DataSource دریافت می‌کنند. برای نمونه در اینجا یک combo box را در نظر بگیرید. در مثال اول، خاصیت dataSource کنترل ComboBox در همان افزونه‌ی جی‌کوئری تنظیم شده‌است:

```
<input id="colorPicker1" />
    <script type="text/javascript">
        $(document).ready(function () {
            $("#colorPicker1").kendoComboBox({
                dataSource: ["Blue", "Green", "Red", "Yellow"]
            });
        });
    </script>
```

و در مثال دوم، نحوه‌ی مقدار دهی ویژگی data-source را در حالت declarative initialization مشاهده می‌کنید. همانطور که عنوان شد، در این حالت ذکر متد kendo.init بر روی یک ناحیه و یا یک کنترل ویژه، جهت آغاز فعالیت آن ضروری است:

```
<input id="colorPicker2" data-role="combobox" data-source='["Blue", "Green", "Red", "Yellow"]' />
    <script type="text/javascript">
        $(document).ready(function () {
            kendo.init($("#colorPicker2"));
        });
    </script>
```

کار با رویدادهای ویجت‌های وب

نحوه‌ی کار با رویدادهای ویجت‌های وب نیز بر اساس نحوه‌ی آغاز آن‌ها متفاوت است. در مثال‌های ذیل، دو حالت متفاوت تنظیم رویداد change را توسط خواص افزونه‌ی جی‌کوئری:

```
<input id="colorPicker3" />
    <script type="text/javascript">
        function onColorChange(e) {
             alert('Color Change!');
        }

        $(document).ready(function () {
            $("#colorPicker3").kendoComboBox({
                dataSource: ["Blue", "Green", "Red", "Yellow"],
                change: onColorChange
```

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

و همچنین توسط ویژگی data-change مشاهده می‌کنید:

```
<input id="colorPicker4" data-role="combobox"
           data-source='["Blue", "Green", "Red", "Yellow"]'
           data-change="onColorChange" />

    <script type="text/javascript">
        function onColorChange(e) {
            alert('Color Change!');
        }

        $(document).ready(function () {
            kendo.init($("#colorPicker4"));
        });
    </script>
```

در هر دو حالت، انتخاب یک گزینه‌ی جدید combo box، سبب فراخوانی متد callback ایی به نام onColorChange می‌شود.

تغییر قالب ویجت‌های وب

Kendo UI همیشه یک جفت CSS را جهت تعیین قالب‌های ویجت‌های خود، مورد استفاده قرار می‌دهد. برای نمونه در مثال‌های فوق، kendo.common.min.css حاوی اطلاعات محل قرارگیری و اندازه‌ی ویجت‌ها است. شیوه نامه‌ی دوم همیشه به شکل kendo.[skin].min.css تعریف می‌شود که دارای اطلاعات رنگ و پس زمینه‌ی ویجت‌ها خواهد بود؛ مانند kendo.black.min.css، kendo.blueopal.min.css و امثال آن که در پوشه‌ی styles قابل مشاهده هستند.
همچنین باید دقت داشت که همیشه common باید پیش از skin ذکر شود؛ زیرا در تعدادی از حالات، شیوه نامه‌ی skin، اطلاعات common را بازنویسی می‌کند.
علاوه بر skinهای پیش فرض موجود در پوشه‌ی styles، امکان استفاده از یک theme builder آنلاین نیز وجود دارد: kendo-ui-themebuilder

نظرات خوانندگان

نویسنده: انصاری
تاریخ: ۱۳۹۳/۰۹/۰۱ ۸:۵۹

امکان تبدیل تاریخ (از میلادی به شمسی) در ویجت‌های وب مثل dataPicker , Calendar , Scheduler وجود دارد؟ من میخواستم از Scheduler استفاده کنم ولی با تاریخ و فرمت (ماه و روزهای هفته) فارسی، به نظر شما امکانش هست بدون دردسر (و به شکل استاندارد) این تقویم رو شمسی کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۰۱ ۱۰:۵

در اینجا بحث شده.

عنوان: کار با Kendo UI DataSource
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۵ ۲۲:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.NET Web API, jQuery, Kendo UI, json.net

Kendo UI DataSource جهت تامین داده‌های سمت کلاینت ویجت‌های مختلف KendoUI طراحی شده‌است و به عنوان یک اینترفیس استاندارد قابل استفاده توسط تمام کنترل‌های داده‌ای Kendo UI کاربرد دارد. Kendo UI DataSource امکان کار با منابع داده محلی، مانند اشیاء و آرایه‌های جاوا اسکریپتی و همچنین منابع تامین شده از راه دور، مانند JSONP، JSON و XML را دارد. به علاوه توسط آن می‌توان اعمال ثبت، ویرایش و حذف اطلاعات، به همراه صفحه بندی، گروه بندی و مرتب سازی داده‌ها را کنترل کرد.

استفاده از منابع داده محلی

در ادامه مثالی را از نحوه‌ی استفاده از یک منبع داده محلی جاوا اسکریپتی، مشاهده می‌کنید:

```
<script type="text/javascript">
        $(function () {
            var cars = [
                { "Year": 2000, "Make": "Hyundai", "Model": "Elantra" },
                { "Year": 2001, "Make": "Hyundai", "Model": "Sonata" },
                { "Year": 2002, "Make": "Toyota", "Model": "Corolla" },
                { "Year": 2003, "Make": "Toyota", "Model": "Yaris" },
                { "Year": 2004, "Make": "Honda", "Model": "CRV" },
                { "Year": 2005, "Make": "Honda", "Model": "Accord" },
                { "Year": 2000, "Make": "Honda", "Model": "Accord" },
                { "Year": 2002, "Make": "Kia", "Model": "Sedona" },
                { "Year": 2004, "Make": "Fiat", "Model": "One" },
                { "Year": 2005, "Make": "BMW", "Model": "M3" },
                { "Year": 2008, "Make": "BMW", "Model": "X5" }
            ];

            var carsDataSource = new kendo.data.DataSource({
                data: cars
            });

            carsDataSource.read();
            alert(carsDataSource.total());
        });
    </script>
```

در اینجا cars آرایه‌ای از اشیاء جاوا اسکریپتی بیانگر ساختار یک خودرو است. سپس برای معرفی آن به Kendo UI، کار با مقدار دهی خاصیت data مربوط به new kendo.data.DataSource شروع می‌شود.
ذکر new kendo.data.DataSource به تنهایی به معنای مقدار دهی اولیه است و در این حالت منبع داده مورد نظر، استفاده نخواهد شد. برای مثال اگر متد total آن‌را جهت یافتن تعداد عناصر موجود در آن فراخوانی کنید، صفر را بازگشت می‌دهد. برای شروع به کار با آن، نیاز است ابتدا متد read را بر روی این منبع داده مقدار دهی شده، فراخوانی کرد.

استفاده از منابع داده راه دور

در برنامه‌های کاربردی، عموما نیاز است تا منبع داده را از یک وب سرور تامین کرد. در اینجا نحوه‌ی خواندن اطلاعات JSON بازگشت داده شده از جستجوی توئیتر را مشاهده می‌کنید:

```
<script type="text/javascript">
        $(function () {
            var twitterDataSource = new kendo.data.DataSource({
                transport: {
                    read: {
                        url: "http://search.twitter.com/search.json",
                        dataType: "jsonp",
                        contentType: 'application/json; charset=utf-8',
                        type: 'GET',
                        data: { q: "#kendoui" }
                    },
                    schema: { data: "results" }
```

```
                },
                error: function (e) {
                    alert(e.errorThrown.stack);
                }
            });
        });
    </script>
```

در قسمت transport، جزئیات تبادل اطلاعات با سرور راه دور مشخص می‌شود؛ برای مثال url ارائه دهنده‌ی سرویس، dataType بیانگر نوع داده مورد انتظار و data کار مقدار دهی پارامتر مورد انتظار توسط سرویس توئیتر را انجام می‌دهد. در اینجا چون صرفا عملیات خواندن اطلاعات صورت می‌گیرد، خاصیت read مقدار دهی شده‌است.
در قسمت schema مشخص می‌کنیم که اطلاعات JSON بازگشت داده شده توسط توئیتر، در فیلد results آن قرار دارد.

کار با منابع داده OData

علاوه بر فرمت‌های یاد شده، Kendo UI DataSource امکان کار با اطلاعاتی از نوع OData را نیز دارا است که تنظیمات ابتدایی آن به صورت ذیل است:

```
<script type="text/javascript">
            var moviesDataSource = new kendo.data.DataSource({
                type: "odata",
                transport: {
                    read: "http://demos.kendoui.com/service/Northwind.svc/Orders"
                },
                error: function (e) {
                    alert(e.errorThrown.stack);
                }
            });
        });
    </script>
```

همانطور که ملاحظه می‌کنید، تنظیمات ابتدایی آن اندکی با حالت remote data پیشین متفاوت است. در اینجا ابتدا نوع داده‌ی بازگشتی مشخص می‌شود و در قسمت transport، خاصیت read آن، آدرس سرویس را دریافت می‌کند.

یک مثال: دریافت اطلاعات از ASP.NET Web API

یک پروژه‌ی جدید ASP.NET را آغاز کنید. تفاوتی نمی‌کند که Web forms باشد یا MVC؛ از این جهت که مباحث Web API در هر دو یکسان است.
سپس یک کنترلر جدید Web API را به نام ProductsController با محتوای زیر ایجاد کنید:

```
using System.Collections.Generic;
using System.Web.Http;

namespace KendoUI02
{
    public class Product
    {
        public int Id { set; get; }
        public string Name { set; get; }
    }

    public class ProductsController : ApiController
    {
        public IEnumerable<Product> Get()
        {
            var products = new List<Product>();
            for (var i = 1; i <= 100; i++)
            {
                products.Add(new Product { Id = i, Name = "Product " + i });
            }
            return products;
        }
    }
}
```

در این مثال، هدف صرفا ارائه یک خروجی ساده JSON از طرف سرور است.

در ادامه نیاز است تعریف مسیریابی ذیل نیز به فایل Global.asax.cs برنامه اضافه شود تا بتوان به آدرس api/products در سایت، دسترسی یافت:

```
using System;
using System.Web.Http;
using System.Web.Routing;

namespace KendoUI02
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            RouteTable.Routes.MapHttpRoute(
               name: "DefaultApi",
               routeTemplate: "api/{controller}/{id}",
               defaults: new { id = RouteParameter.Optional }
               );
        }
    }
}
```

در ادامه فایلی را به نام Index.html (یا در یک View و یا یک فایل aspx دلخواه)، محتوای ذیل را اضافه کنید:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta charset="utf-8" />
    <title>Kendo UI: Implemeting the Grid</title>

    <link href="styles/kendo.common.min.css" rel="stylesheet" type="text/css" />
    <link href="styles/kendo.default.min.css" rel="stylesheet" type="text/css" />
    <script src="js/jquery.min.js" type="text/javascript"></script>
    <script src="js/kendo.all.min.js" type="text/javascript"></script>
</head>
<body>

    <div id="report-grid"></div>
    <script type="text/javascript">
        $(function () {
            var productsDataSource = new kendo.data.DataSource({
                transport: {
                    read: {
                        url: "api/products",
                        dataType: "json",
                        contentType: 'application/json; charset=utf-8',
                        type: 'GET'
                    }
                },
                error: function (e) {
                    alert(e.errorThrown.stack);
                },
                pageSize: 5,
                sort: { field: "Id", dir: "desc" }
            });

            $("#report-grid").kendoGrid({
                dataSource: productsDataSource,
                autoBind: true,
                scrollable: false,
                pageable: true,
                sortable: true,
                columns: [
                    { field: "Id", title: "#" },
                    { field: "Name", title: "Product" }
                ]
            });
        });
    </script>
</body>
</html>
```

- ابتدا فایل‌های اسکریپت و CSS مورد نیاز Kendo UI اضافه شده‌اند.
- گرید صفحه، در محل div ایی با id مساوی report-grid تشکیل خواهد شد.
- سپس DataSource ایی که به آدرس api/products اشاره می‌کند، تعریف شده و در آخر productsDataSource را توسط یک kendoGrid نمایش داده‌ایم.
- نحوه‌ی تعریف productsDataSource، در قسمت استفاده از منابع داده راه دور ابتدای بحث توضیح داده شد. در اینجا فقط دو خاصیت pageSize و sort نیز به آن اضافه شده‌اند. این دو خاصیت بر روی نحوه‌ی نمایش گرید نهایی تاثیر گذار هستند. pageSize تعداد رکورد هر صفحه را مشخص می‌کند و sort نحوه‌ی مرتب سازی را بر اساس فیلد Id و در حالت نزولی قرار می‌دهد.
- در ادامه، ابتدایی‌ترین حالت کار با kendoGrid را ملاحظه می‌کنید.
- تنظیم dataSource و autoBind: true (حالت پیش فرض)، سبب خواهند شد تا به صورت خودکار، اطلاعات JSON از مسیر api/products خوانده شوند.
- سه خاصیت بعدی صفحه بندی و مرتب سازی خودکار ستون‌ها را فعال می‌کنند.
- در آخر هم دو ستون گرید، بر اساس نام‌های خواص کلاس Product تعریف شده‌اند.

سورس کامل این قسمت را از اینجا می‌توانید دریافت کنید:

KendoUI02.zip

نظرات خوانندگان

نویسنده: ژوپیتر
تاریخ: ۱۳۹۳/۱۱/۰۹ ۱۰:۲۷

سلام
چرا زمان اجرا به جای نمایش اطلاعات گرید، پیام undefined داده می‌شود؟ بنده از MVC استفاده کردم و کاملا مطابق مقاله مسیریابی و ... را اعمال کردم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۰۹ ۱۱:۱۰

روی چه سطری پیام خطا دریافت کردید؟
حین کار با کتابخانه‌های جاوا اسکریپتی باید مدام کنسول developer مرورگر را باز نگه دارید تا بتوانید خطاها را بهتر بررسی و دیباگ کنید.

نویسنده: ژوپیتر
تاریخ: ۱۳۹۳/۱۱/۱۱ ۹:۳

خطا توسط error handler، گرفته می‌شود وقتی این بخش نیز حذف می‌شود مرورگر فقط منتظر دریافت اطلاعات از api/products می‌ماند و خطایی از کد گرفته نمی‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۱۱ ۱۰:۳

خطای 404 به معنای یافتن نشدن مسیر تنظیمی "api/products" :url در برنامه شما است.

مطابق تصویر، مسیر /api/product **home** در حال جستجو است که به ریشه‌ی سایت اشاره نمی‌کند.
- آیا در ASP.NET MVC از یک اکشن متد برای بازگرداندن لیست جی‌سون محصولات استفاده کرده‌اید؟ اگر بله، از مطلب « نحوه صحیح تولید Url در ASP.NET MVC » کمک بگیرید؛ مثلا آدرس آن چنین شکلی را پیدا خواهد کرد:

```
@Url.Action("method_name", "Home")
```

- اگر وب API است در یک برنامه‌ی MVC، از روش زیر استفاده کنید:

```
'@Url.RouteUrl("DefaultApi", new { httproute = "", controller = "products" })'
```

و البته فرض بر این است که مسیریابی DefaultApi پیشتر در برنامه‌ی شما ثبت شده‌است:

```
routes.MapHttpRoute(
    name: "DefaultApi",
    routeTemplate: "api/{controller}/{id}",
    defaults: new { id = RouteParameter.Optional }
);
```

نویسنده: ژوپیتر
تاریخ: ۱۳۹۳/۱۱/۱۱ ۱۱:۸

- ممنون؛ از وب API استفاده شده بود که با راهنمایی شما حل شد. ولی استفاده از خروجی Json کنترلر به‌نظرم بهتر و ساده‌تر اومد. آیا تفاوتی محسوسی بین این دو روش وجود داره؟
- آیا امکان استفاده مستقیم اشیا Strongly Typed هم در توابع این کتابخانه وجود داره؟ (منظورم همون @model به صورت مستقیم یا با واسطه است).

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۱۱ ۱۱:۲۰

- خیر. اگر هدف صرفا بازگشت جی‌سون است، تفاوت خاصی ندارند. فقط Web API به صورت پیش فرض از JSON.NET استفاده می‌کند؛ اطلاعات بیشتر و همچنین با وب فرم‌ها هم سازگار است.
- بله: استفاده ازExpressionها جهت ایجاد Strongly typed view در ASP.NET MVC

عنوان: **صفحه بندی، مرتب سازی و جستجوی پویای اطلاعات به کمک Kendo UI Grid**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۶ ۱۶:۳۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.NET Web API, jQuery, DynamicLINQ, Kendo UI, json.net

پس از آشنایی مقدماتی با Kendo UI DataSource ، اکنون می‌خواهیم از آن جهت صفحه بندی، مرتب سازی و جستجوی پویای سمت سرور استفاده کنیم. در مثال قبلی، هر چند صفحه بندی فعال بود، اما پس از دریافت تمام اطلاعات، این اعمال در سمت کاربر انجام و مدیریت می‌شد.

مدل برنامه

در اینجا قصد داریم لیستی را با ساختار کلاس Product در اختیار Kendo UI گرید قرار دهیم:

```
namespace KendoUI03.Models
{
    public class Product
    {
        public int Id { set; get; }
        public string Name { set; get; }
        public decimal Price { set; get; }
        public bool IsAvailable { set; get; }
    }
}
```

پیشنیاز تامین داده مخصوص Kendo UI Grid

برای ارائه اطلاعات مخصوص Kendo UI Grid، ابتدا باید درنظر داشت که این گرید، درخواست‌های صفحه بندی خود را با فرمت ذیل ارسال می‌کند. همانطور که مشاهده می‌کنید، صرفا یک کوئری استرینگ با فرمت JSON را دریافت خواهیم کرد:

```
/api/products?{"take":10,"skip":0,"page":1,"pageSize":10,"sort":[{"field":"Id","dir":"desc"}]}
```

سپس این گرید نیاز به سه فیلد، در خروجی JSON نهایی خواهد داشت:

```
{
"Data":
[
{"Id":1500,"Name":"1500 نام","Price":2499.0,"IsAvailable":false},
{"Id":1499,"Name":"1499 نام","Price":2498.0,"IsAvailable":true}
],
"Total":1500,
"Aggregates":null
}
```

فیلد Data که رکوردهای گرید را تامین می‌کنند. فیلد Total که بیانگر تعداد کل رکوردها است و Aggregates که برای گروه بندی بکار می‌رود.

می‌توان برای تمام این‌ها، کلاس و Parser تهیه کرد و یا ... پروژه‌ی سورس بازی به نام Kendo.DynamicLinq نیز چنین کاری را میسر می‌سازد که در ادامه از آن استفاده خواهیم کرد. برای نصب آن تنها کافی است دستور ذیل را صادر کنید:

```
PM> Install-Package Kendo.DynamicLinq
```

Kendo.DynamicLinq به صورت خودکار System.Linq.Dynamic را نیز نصب می‌کند که از آن جهت صفحه بندی پویا استفاده خواهد شد.

تامین کننده‌ی داده سمت سرور

همانند مطلب کار با Kendo UI DataSource ، یک ASP.NET Web API Controller جدید را به پروژه اضافه کنید و همچنین مسیریابی‌های مخصوص آن‌را به فایل global.asax.cs نیز اضافه نمائید.

```
using System.Linq;
using System.Net.Http;
using System.Web.Http;
using Kendo.DynamicLinq;
using KendoUI03.Models;
using Newtonsoft.Json;

namespace KendoUI03.Controllers
{
    public class ProductsController : ApiController
    {
        public DataSourceResult Get(HttpRequestMessage requestMessage)
        {
            var request = JsonConvert.DeserializeObject<DataSourceRequest>(
                requestMessage.RequestUri.ParseQueryString().GetKey(0)
            );

            var list = ProductDataSource.LatestProducts;
            return list.AsQueryable()
                       .ToDataSourceResult(request.Take, request.Skip, request.Sort, request.Filter);
        }
    }
}
```

تمام کدهای این کنترلر همین چند سطر فوق هستند. با توجه به ساختار کوئری استرینگی که در ابتدای بحث عنوان شد، نیاز است آن‌را توسط کتابخانه‌ی JSON.NET تبدیل به یک نمونه از DataSourceRequest نمائیم. این کلاس در Kendo.DynamicLinq تعریف شده‌است و حاوی اطلاعاتی مانند take و skip کوئری LINQ نهایی است.
ProductDataSource.LatestProducts صرفا یک لیست جنریک تهیه شده از کلاس Product است. در نهایت با استفاده از متد الحاقی جدید ToDataSourceResult ، به صورت خودکار مباحث صفحه بندی سمت سرور به همراه مرتب سازی اطلاعات، صورت گرفته و اطلاعات نهایی با فرمت DataSourceResult بازگشت داده می‌شود. DataSourceResult نیز در Kendo.DynamicLinq تعریف شده و سه فیلد یاد شده‌ی Data، Total و Aggregates را تولید می‌کند.

تا اینجا کارهای سمت سرور این مثال به پایان می‌رسد.

تهیه View نمایش اطلاعات ارسالی از سمت سرور

اعمال مباحث بومی سازی

```
<head>
    <meta charset="utf-8" />
    <meta http-equiv="Content-Language" content="fa" />
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

    <title>Kendo UI: Implemeting the Grid</title>

    <link href="styles/kendo.common.min.css" rel="stylesheet" type="text/css" />
    <!--شیوه نامه‌ی مخصوص راست به چپ سازی-->
    <link href="styles/kendo.rtl.min.css" rel="stylesheet" />
    <link href="styles/kendo.default.min.css" rel="stylesheet" type="text/css" />
    <script src="js/jquery.min.js" type="text/javascript"></script>
    <script src="js/kendo.all.min.js" type="text/javascript"></script>

    <!--محل سفارشی سازی پیام‌ها و مسایل بومی-->
    <script src="js/cultures/kendo.culture.fa-IR.js" type="text/javascript"></script>
    <script src="js/cultures/kendo.culture.fa.js" type="text/javascript"></script>
    <script src="js/messages/kendo.messages.en-US.js" type="text/javascript"></script>

    <style type="text/css">
        body {
            font-family: tahoma;
            font-size: 9pt;
        }
    </style>

    <script type="text/javascript">
        // جهت استفاده از فایل: kendo.culture.fa-IR.js
        kendo.culture("fa-IR");
    </script>
</head>
```

- در اینجا چند فایل js و css جدید اضافه شده‌اند. فایل kendo.rtl.min.css جهت تامین مباحث RTL توکار Kendo UI کاربرد دارد.
- سپس سه فایل kendo.culture.fa.js، kendo.culture.fa-IR.js و kendo.messages.en-US.js نیز اضافه شده‌اند. فایل‌های fa و fa-Ir آن هر چند به ظاهر برای ایران طراحی شده‌اند، اما نام ماه‌های موجود در آن عربی است که نیاز به ویرایش دارد. به همین جهت به سورس این فایل‌ها، جهت ویرایش نهایی نیاز خواهد بود که در پوشه‌ی src\js\cultures مجموعه‌ی اصلی Kendo UI موجود هستند (ر.ک. فایل پیوست).
- فایل kendo.messages.en-US.js حاوی تمام پیام‌های مرتبط با Kendo UI است. برای مثال«رکوردهای 10 تا 15 از 1000 ردیف» را در اینجا می‌توانید به فارسی ترجمه کنید.
- متد kendo.culture کار مشخص سازی فرهنگ بومی برنامه را به عهده دارد. برای مثال در اینجا به fa-IR تنظیم شده‌است. این مورد سبب خواهد شد تا از فایل kendo.culture.fa-IR.js استفاده گردد. اگر مقدار آن‌را به fa تنظیم کنید، از فایل kendo.culture.fa.js کمک گرفته خواهد شد.

راست به چپ سازی گرید

تنها کاری که برای راست به چپ سازی Kendo UI Grid باید صورت گیرد، محصور سازی div آن در یک div با کلاس مساوی k-

rtl است:

```
<div class="k-rtl">
        <div id="report-grid"></div>
    </div>
```

k-rtl و تنظیمات آن در فایل kendo.rtl.min.css قرار دارند که در ابتدای head صفحه تعریف شده‌است.

تامین داده و نمایش گرید

در ادامه کدهای کامل DataSource و Kendo UI Grid را ملاحظه می‌کنید:

```
<script type="text/javascript">
        $(function () {
            var productsDataSource = new kendo.data.DataSource({
                transport: {
                    read: {
                        url: "api/products",
                        dataType: "json",
                        contentType: 'application/json; charset=utf-8',
                        type: 'GET'
                    },
                    parameterMap: function (options) {
                        return kendo.stringify(options);
                    }
                },
                schema: {
                    data: "Data",
                    total: "Total",
                    model: {
                        fields: {
                            "Id": { type: "number" }, //تعیین نوع فیلد برای جستجوی پویا مهم است
                            "Name": { type: "string" },
                            "IsAvailable": { type: "boolean" },
                            "Price": { type: "number" }
                        }
                    }
                },
                error: function (e) {
                    alert(e.errorThrown);
                },
                pageSize: 10,
                sort: { field: "Id", dir: "desc" },
                serverPaging: true,
                serverFiltering: true,
                serverSorting: true
            });

            $("#report-grid").kendoGrid({
                dataSource: productsDataSource,
                autoBind: true,
                scrollable: false,
                pageable: true,
                sortable: true,
                filterable: true,
                reorderable: true,
                columnMenu: true,
                columns: [
                    { field: "Id", title: "شماره", width: "130px" },
                    { field: "Name", title: "نام محصول" },
                    {
                        field: "IsAvailable", title: "موجود است",
                        template: '<input type="checkbox" #= IsAvailable ? checked="checked" : "" #
disabled="disabled" ></input>'
                    },
                    { field: "Price", title: "قیمت", format: "{0:c}" }
                ]
            });
        });
    </script>
```

- با تعاریف مقدماتی Kendo UI DataSource پیشتر آشنا شده‌ایم و قسمت read آن جهت دریافت اطلاعات از سمت سرور کاربرد

دارد.

- در اینجا ذکر contentType الزامی است. زیرا ASP.NET Web API بر این اساس است که تصمیم می‌گیرد، خروجی را به صورت JSON ارائه دهد یا XML.

- با استفاده از parameterMap، سبب خواهیم شد تا پارامترهای ارسالی به سرور، با فرمت صحیحی تبدیل به JSON شده و بدون مشکل به سرور ارسال گردند.

- در قسمت schema باید نام فیلدهای موجود در DataSourceResult دقیقا مشخص شوند تا گرید بداند که data را باید از چه فیلدی استخراج کند و تعداد کل ردیف‌ها در کدام فیلد قرار گرفته‌است.

- نحوه‌ی تعریف model را نیز در اینجا ملاحظه می‌کنید. ذکر نوع فیلدها در اینجا بسیار مهم است و اگر قید نشوند، در حین جستجوی پویا به مشکل برخواهیم خورد. زیرا پیش فرض نوع تمام فیلدها string است و در این حالت نمی‌توان عدد 1 رشته‌ای را با یک فیلد از نوع int در سمت سرور مقایسه کرد.

- در اینجا serverFiltering، serverPaging و serverSorting نیز به true تنظیم شده‌اند. اگر این مقدار دهی‌ها صورت نگیرد، این اعمال در سمت کلاینت انجام خواهند شد.

پس از تعریف DataSource، تنها کافی است آن‌را به خاصیت dataSource یک kendoGrid نسبت دهیم.

- autoBind: true سبب می‌شود تا اطلاعات DataSource بدون نیاز به فراخوانی متد read آن به صورت خودکار دریافت شوند.

- با تنظیم scrollable: false، اعلام می‌کنیم که قرار است تمام رکوردها در معرض دید قرارگیرند و اسکرول پیدا نکنند.

- pageable: true صفحه بندی را فعال می‌کند. این مورد نیاز به تنظیم pageSize: 10 در قسمت DataSource نیز دارد.

- با sortable: true مرتب سازی ستون‌ها با کلیک بر روی سرستون‌ها فعال می‌گردد.

- filterable: true به معنای فعال شدن جستجوی خودکار بر روی فیلدها است. کتابخانه‌ی Kendo.DynamicLinq حاصل آن‌را در سمت سرور مدیریت می‌کند.

- reorderable: true سبب می‌شود تا کاربر بتواند محل قرارگیری ستون‌ها را تغییر دهد.

- ذکر columnMenu: true اختیاری است. اگر ذکر شود، امکان مخفی سازی انتخابی ستون‌ها نیز مسیر خواهد شد.

- در آخر ستون‌های گرید مشخص شده‌اند. با تعیین "format: "{0:c} سبب نمایش فیلدهای قیمت با سه رقم جدا کننده خواهیم شد. مقدار ریال آن از فایل فرهنگ جاری تنظیم شده دریافت می‌گردد. با استفاده از template تعریف شده نیز سبب نمایش فیلد bool به صورت یک checkbox خواهیم شد.

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید:

KendoUI03.zip

نظرات خوانندگان

نویسنده: حمیدرضا کبیری
تاریخ: ۱۳۹۳/۰۸/۱۶ ۱۹:۳

آیا kendo UI کاملا از زبان فارسی پشتیبانی میکند ؟
برای calender آن ، به تقویم شمی گزینه ای موجود هست ؟
این گزینه با ورژن ۲۰۱۴/۱/۳۱۸ مطابقت دارد ، آیا با ورژنهای جدید مشکلی نخواهد داشت ؟

نویسنده: احمد رجبی
تاریخ: ۱۳۹۳/۰۸/۱۶ ۲۰:۱۵

میتوانید با اضافه کردن این اسکریپت تمامی قسمتهای kendo را به زبان فارسی ترجمه کنید.

نویسنده: سعیدجلالی
تاریخ: ۱۳۹۳/۰۸/۱۷ ۸:۴۴

با تشکر از مطلب مفید شما من از wrapper mvc مجموعه kendo استفاده میکنم
توی مطالب شما در مورد استفاده از Kendo.DynamicLinq صحبت شد خواستم بدونم آیا وقتی از wrapper هم استفاده میکنیم استفاده از این پکیج لازم هست؟

```
@(Html.Kendo().Grid(Model)
      .Name("gridKendo")
      .Columns(columns =>
      {
          columns.Bound(c => c.Name).Width(50);
          columns.Bound(c => c.Family).Width(100);
          columns.Bound(c => c.Tel);
      })
      .Pageable(pager => pager
          .Input(true)
          .Numeric(true)
          .Info(true)
          .PreviousNext(true)
          .Refresh(true)
          .PageSizes(true)
      )
      )
```

چون من با استفاده از telerik profiler وقتی درخواست رو بررسی مکنم توی دستور sql چنین دستوری رو در انتها مشاهده میکنم:
صفحه اول:

```
SELECT *
FROM (
    SELECT
    FROM table a
)
WHERE ROWNUM <= :TAKE
```

صفحات بعد:

```
SELECT *
FROM (
    SELECT
        a.*,
        ROWNUM OA_ROWNUM
    FROM (
        FROM table a
```

```
    ) a
    WHERE ROWNUM <= :TAKE
)
WHERE OA_ROWNUM > :SKIP
```

پایگاه داده اوراکل است.

نویسنده: سعیدجلالی
تاریخ: ۱۳۹۳/۰۸/۱۷ ۹:۱۲

امکان فارسی شدن تمام بخش‌ها وجود دارد.
تقویم هم فارسی شده است در این سایت برای نسخه‌های جدیدتر هم باید دوتا فایل جاوا اسکریپت all و mvc رو خودتون تغییر بدهید (با توجه به الگوی انجام شده در فایل فارسی شده فوق)
ولی برای تقویم زمانبدی scheduler من فارسی ندیده ام

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۷ ۹:۳۱

مطلب فوق نه وابستگی خاصی به وب فرم‌ها دارد و نه ASP.NET MVC. ویو آن یک فایل HTML ساده‌است و سمت سرور آن فقط یک کنترلر ASP.NET Web API که با تمام مشتقات ASP.NET سازگار است. در این حالت یک نفر می‌تواند ASP.NET نگارش خودش را خلق کند؛ بدون اینکه نگران جزئیات وب فرم‌ها باشد یا ASP.NET MVC. ضمنا دانش جاوا اسکریپتی آن هم قابل انتقال است؛ چون اساسا Kendo UI برای فناوری سمت سرور خاصی طراحی نشده‌است و حالت اصل آن با PHP، Java و امثال آن هم کار می‌کند.

نویسنده: میثم آقااحمدی
تاریخ: ۱۳۹۳/۰۸/۱۷ ۱۳:۱۷

در کنترلر این خط باعث بارگذاری تمامی داده‌ها می‌شود

```
var list = ProductDataSource.LatestProducts;
```

آیا راه حلی وجود دارد که دیتای به تعداد همان pagesize از پایگاه خوانده شود؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۷ ۱۳:۲۸

- این فقط یک مثال هست و منبع داده‌ای صرفا جهت دموی ساده‌ی برنامه. فقط برای اینکه با یک کلیک بتوانید برنامه را اجرا کنید و نیازی به برپایی و تنظیم بانک اطلاعاتی و امثال آن نداشته باشد.
- شما در کدها و کوئری‌های مثلا EF در اصل با یک سری IQueryable کار می‌کنید. همینجا باید متد الحاقی ToDataSourceResult را اعمال کنید تا نتیجه‌ی نهایی در حداقل بار تعداد رفت و برگشت و با کوئری مناسبی بر اساس پارامترهای دریافتی به صورت خودکار تولید شود. در انتهای کار بجای مثلا ToList بنویسید ToDataSourceResult.

نویسنده: امین
تاریخ: ۱۳۹۳/۰۸/۱۷ ۱۴:۴۱

سلام من در ویو خودم نمیتونم اطلاعاتم رو تو kendo.grid ببینم و برای من یک لیست استرینگ در ویو نمایش داده میشه و به این شکل در کنترلر و ویو کد نویسی کردم .

```
public class EFController : Controller    {
        //
        // GET: /EF/

        public ActionResult AjaxConnected([DataSourceRequest] DataSourceRequest request )
```

```
        {
            using (var dbef=new dbTestEntities())
            {
                IQueryable<Person> persons = dbef.People;
                DataSourceResult result = persons.ToDataSourceResult(request);
                return Json(result.Data,JsonRequestBehavior.AllowGet);
            }
        }


    }
```

و ویو

```
@{    ViewBag.Title = "AjaxConnected";
}


<h2>AjaxConnected</h2>
@(Html.Kendo().Grid<TelerikMvcApp2.Models.Person>(  )
      .Name("Grid")
      .DataSource(builder => builder
          .Ajax()
          .Read(operationBuilder => operationBuilder.Action("AjaxConnected", "EF"))


      )
      .Columns(factory =>
      {
          factory.Bound(person => person.personId);
          factory.Bound(person => person.Name);
          factory.Bound(person => person.LastName);
      })
      .Pageable()
      .Sortable())
```

و یک لیست استرینگ بهم در عمل خروجی میده و از خود قالب kendogrid خبری نیست . من اطلاعات رو به طور json پاس میدم و ajaxi میگیرم.

حالا قبلش همچین خطلایی داشتم که به allowget ایراد میگرفت ولی در کل با JsonRequestBehavior.AllowGet حل شد و حالا فقط یه لسیت بهم خروجی میده! و از ظاهر گرید خبری نیست. و اگر به جای json نوشته بشه view و با ویو return کنم ظاهر kendogrid رو دارم اما خروجی دارای مقداری نیست! اینم خروجی استرینگ من :(

```
[{"personId":1,"Name":"Amin","LastName":"Saadati"},  {"personId":2,"Name":"Fariba","LastName":"Ghochani
"},{"personId":4,"Name":"Milad","LastName":"Rahman  i"},{"personId":5,"Name":"rima","LastName":"rad"},
{"personId":6,"Name":"ali","LastName":"kiva"},{"pe  rsonId":7,"Name":"sahel","LastName":"abasi"},{"per
sonId":8,"Name":"medi","LastName":"ghaem"},{"perso  nId":9,"Name":"mino","LastName":"kafash"},{"person
Id":10,"Name":"behzad","LastName":"tizro"},{"perso  nId":11,"Name":"toti","LastName":"saadati"},{"pers
onId":12,"Name":"parinaz","LastName":"karami"},{"p  ersonId":13,"Name":"sadegh","LastName":"hojati"},{
"personId":14,"Name":"milad","LastName":"ebadipor"  },{"personId":15,"Name":"farid","LastName":"riazi"
},{"personId":16,"Name":"said","LastName":"abdoli"  },{"personId":17,"Name":"behzad","LastName":"ariaf
ar"},{"personId":18,"Name":"jamshid","LastName":"k  otahi"}]
```

این سوال رو در چند سایت پرسیدم و به جوابی برایش نرسیدم. و نمیدونم ایراد کدهای نوشته شده ام کجاست!
متشکرم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۷ ۱۵:۲

- قصد پشتیبانی از wrapperهای آن‌را ندارم. لطفا خارج از موضوع سؤال نپرسید. اگر کسی دوست داشت در این زمینه مطلب منتشر کند، خوب. ولی من چنین قصدی ندارم.
- عرض کردم اگر از wrapperها استفاده کنید، به علت عدم درک زیر ساخت اصلی Kendo UI، قادر به دیباگ کار نخواهید بود.
- اگر متن را مطالعه کنید در قسمت «پیشنیاز تامین داده مخصوص Kendo UI Grid » دقیقا شکل نهایی خروجی JSON مورد نیاز ارائه شده‌است. این خروجی در سه فیلد data، total و aggregate قرار می‌گیرد. شما الان فقط قسمت data آن‌را بازگشت

داده‌اید؛ بجای اصل و کل آن. نام این سه فیلد هم مهم نیست؛ اما هر چیزی که تعیین می‌شوند، باید در قسمت data source در خاصیت schema آن مانند مثالی که در مطلب جاری آمده (در قسمت «تامین داده و نمایش گرید »)، دقیقا مشخص شوند، تا Kendo UI بداند که اطلاعات مختلف را باید از چه فیلدهایی از JSON خروجی دریافت کند.

نویسنده: وحید محمّدطاهری
تاریخ: ۱۳۹۳/۱۰/۰۷ ۱۴:۲۴

با سلام و خدا قوت

آقای نصیری، model ای که باید در قسمت schema تعریف بشه چطوری میشه اونو دینامیک تولید کرد.
من یک چنین حالتی رو ایجاد کردم ولی نمی‌دونم چطوری باید اسم ستونو براش مشخص کنم.

```
public class Field
    {
        public string Type { get; set; }
    }
```

```
public class Fields : System.Collections.ObjectModel.Collection<Field>
    {
        public System.Collections.IEnumerable ToList()
        {
            return System.Linq.Enumerable.ToList( System.Linq.Enumerable.Select( this ,
                                                                                  field => new {
field.Type } ) );
        }
    }
```

این قسمت اطلاعاتی است که برای ایجاد گرید باز گردانده می‌شود.

```
return new InitializeInfo
                  {
                          ...
                          Model = GetModel()
                  };
```

```
private Model GetModel ()
        {
            return new Model{ Fields = GetFields().ToList() };
        }
```

متد GetColumns شامل 3 ستون می‌باشد که نوع، عنوان و سایر مشخصات رو توش تعریف کردم

```
private Fields GetFields()
        {
            var fields = new Fields();
            foreach ( var column in GetColumns() )
            {
                fields.Add( new Field { Type = column.DataType } );
            }
            return fields;
        }
```

الان خروجی که تولید میشه اینجوریه

```
"model": {

    "fields": [
        {
            "type": "string"
```

```
        },
        {
            "type": "string"
        },
        {
            "type": "datetime"
        }
    ]
}
```

ممنون میشم یه راهنمایی کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۷ ۱۴:۴۱

- پویا هست و خروجی دسترسی هم گرفتید. زمانیکه تعریف می‌کنید:

```
new Field { Type = column.DataType }
```

یعنی در لیست نهایی، خاصیتی با نام ثابت Type و با مقدار متغیر column.DataType را تولید کن (نام خاصیت، مقدار ثابت نام خاصیت را در JSON نهایی تشکیل می‌دهد).
+ نیازی هم به این همه پیچیدگی نداشت. تمام کارهایی را که انجام دادید با تهیه خروجی ساده List<Field> از یک متد دلخواه، یکی هست و نیازی به anonymous type کار کردن نبود.
- به همان کلاس فیلد، خواص دیگر مورد نیاز را اضافه کنید (عنوان و سایر مشخصات یک فیلد) و در نهایت لیست ساده List<Field> را بازگشت دهید. هر خاصیت کلاس Field، یک ستون گرید را تشکیل می‌دهد.
- همچنین دقت داشته باشید اگر از روش مطلب جاری استفاده می‌کنید، اطلاعات ستون‌های نهایی باید در فیلد Data نهایی قرار گیرند (قسمت «پیشنیاز تامین داده مخصوص Kendo UI Grid» در بحث).

نویسنده: وحید محمّدطاهری
تاریخ: ۱۳۹۳/۱۰/۰۷ ۱۵:۴۸

با تشکر از پاسختون

درسته این به صورت پویا تولید میشه ولی شکل model ای که شما در این مطلب توضیح دادید با این چیزی که کد من تولید می‌کنه فرق میکنه

برای شما اول نام فیلد هست بعد نوع اون فیلد، در حالی که نحوه تولید داینامیک اینو نمی‌دونم چطوری باید باشه.

```
model: {
                    fields: {
                        "Id": { type: "number" }, //تعیین نوع فیلد برای جستجوی پویا مهم است
                        "Name": { type: "string" },
                        "IsAvailable": { type: "boolean" },
                        "Price": { type: "number" }
                    }
                }
      }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۷ ۱۶:۱۹

باید از Dictionary استفاده کنید برای تعریف خواص پویا:

```
public class Field
{
  [JsonExtensionData]
  public Dictionary<string, object> Property { get; set; }
```

```
}

public class FieldType
{
  public string Type { get; set; }
}
```

و بعد نحوه استفاده از آن به صورت زیر خواهد بود:

```
var data = new
{
  model = new
  {
   fields = new List<Field>
   {
    new Field
    {
      Property = new Dictionary<string, object>
      {
       {"Id", new FieldType { Type = "number" }},
       {"Name", new FieldType { Type = "string" }}
      }
    }
   }
  }
};
var dataJson = JsonConvert.SerializeObject(data, Formatting.Indented);
```

با این خروجی:

```
{
  "model": {
    "fields": [
      {
        "Id": {
          "Type": "number"
        },
        "Name": {
          "Type": "string"
        }
      }
    ]
  }
}
```

- اگر از Web API استفاده می‌کنید، ذکر سطر JsonConvert.SerializeObject ضروری نیست و به صورت توکار از JSON.NET استفاده می‌کند.
- اگر از ASP.NET MVC استفاده می‌کنید، نیاز است از آن کمک بگیرید . از این جهت که خاصیت JsonExtensionData سبب می‌شود تا نام ثابت خاصیت Property، از خروجی نهایی حذف شود و اعضای دیکشنری، جزئی از خاصیت‌های موجود شوند.
- نکته‌ی « گرفتن خروجی CamelCase از JSON.NET » را هم باید مد نظر داشته باشید.

نویسنده: ژوپیتر
تاریخ: ۱۳۹۳/۱۱/۱۲ ۱۲:۴۷

در صورتی بخواهیم dataSource مربوطه را از همان کنترلر MVC دریافت کنیم، با توجه به اینکه درخواست ارسال شده توسط گرید پارامتریک است، راهکار چیست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۱۲ ۱۳:۳۰

دو سری مثال رسمی kendo-examples-asp-net و kendo-examples-asp-net-mvc در مورد کار با گرید و یک سری از اجزای مهم آن وجود دارند. سری MVC آن دقیقا از Kendo.DynamicLinq مطرح شده در مطلب جاری، استفاده کرده‌است. برای مثال با این

کنترلر و این View .

عنوان: استفاده از Kendo UI templates
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۸ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Kendo UI

در مطلب « صفحه بندی، مرتب سازی و جستجوی پویای اطلاعات به کمک Kendo UI Grid » در انتهای بحث، ستون IsAvailable به صورت زیر تعریف شد:

```
columns: [
               {
                   field: "IsAvailable", title: "موجود است",
                   template: '<input type="checkbox" #= IsAvailable ? checked="checked" : "" #
disabled="disabled" ></input>'
                }
]
```

Templates، جزو یکی از پایه‌های Kendo UI Framework هستند و توسط آن‌ها می‌توان قطعات با استفاده‌ی مجدد HTML ایی را طراحی کرد که قابلیت یکی شدن با اطلاعات جاوا اسکریپتی را دارند.
همانطور که در این مثال نیز مشاهده می‌کنید، قالب‌های Kendo UI از syntax (#) Hash استفاده می‌کنند. در اینجا قسمت‌هایی از قالب که با علامت # محصور می‌شوند، در حین اجرا، با اطلاعات فراهم شده جایگزین خواهند شد.
برای رندر مقادیر ساده می‌توان از # =# استفاده کرد. از # :# برای رندر اطلاعات HTML-encoded کمک گرفته می‌شود و # # برای رندر کدهای جاوا اسکریپتی کاربرد دارد. از حالت HTML-encoded برای نمایش امن اطلاعات دریافتی از کاربران و جلوگیری از حملات XSS استفاده می‌شود.
اگر در این بین نیاز است # به صورت معمولی رندر شود، در حالت کدهای جاوا اسکریپتی به صورت #\\ و در HTML ساده به صورت #\ باید مشخص گردد.

مثالی از نحوه‌ی تعریف یک قالب Kendo UI

```
<!--دریافت اطلاعات از منبع محلی-->
    <script id="javascriptTemplate" type="text/x-kendo-template">
        <ul>
            # for (var i = 0; i < data.length; i++) { #
            <li>#= data[i] #</li>
            # } #
        </ul>
    </script>

    <div id="container1"></div>
    <script type="text/javascript">
        $(function () {
            var data = ['User 1', 'User 2', 'User 3'];
            var template = kendo.template($("#javascriptTemplate").html());
            var result = template(data); //Execute the template
            $("#container1").html(result); //Append the result
        });
    </script>
```

این قالب ابتدا در تگ script محصور می‌شود و سپس نوع آن مساوی text/x-kendo-template قرار می‌گیرد. در ادامه توسط یک حلقه‌ی جاوا اسکریپتی، عناصر آرایه‌ی فرضی data خوانده شده و با کمک Hash syntax در محل‌های مشخص شده قرار می‌گیرند.
در ادامه باید این قالب را رندر کرد. برای این منظور یک div با id مساوی container1 را جهت تعیین محل رندر نهایی اطلاعات مشخص می‌کنیم. سپس متد kendo.template بر اساس id قالب اسکریپتی تعریف شده، یک شیء قالب را تهیه کرده و سپس با ارسال آرایه‌ای به آن، سبب اجرای آن می‌شود. خروجی نهایی، یک قطعه کد HTML است که در محل container1 درج خواهد شد.
همانطور که ملاحظه می‌کنید، متد kendo.template، نهایتا یک رشته را دریافت می‌کند. بنابراین همینجا و به صورت inline نیز می‌توان یک قالب را تعریف کرد.

کار با منابع داده راه دور

فرض کنید مدل برنامه به صورت ذیل تعریف شده‌است:

```
namespace KendoUI04.Models
{
    public class Product
    {
        public int Id { set; get; }
        public string Name { set; get; }
        public decimal Price { set; get; }
        public bool IsAvailable { set; get; }
    }
}
```

و لیستی از آن توسط یک ASP.NET Web API کنترلر، به سمت کاربر ارسال می‌شود:

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using KendoUI04.Models;

namespace KendoUI04.Controllers
{
    public class ProductsController : ApiController
    {
        public IEnumerable<Product> Get()
        {
            return ProductDataSource.LatestProducts.Take(10);
        }
    }
}
```

در سمت کاربر و در View برنامه خواهیم داشت:

```
<!--دریافت اطلاعات از سرور-->
    <div>
        <div id="container2"><ul></ul></div>
    </div>

    <script id="template1" type="text/x-kendo-template">
        <li> #=Id# - #:Name# - #=kendo.toString(Price, "c")#</li>
    </script>

    <script type="text/javascript">
        $(function () {
            var producatsTemplate1 = kendo.template($("#template1").html());

            var productsDataSource = new kendo.data.DataSource({
                transport: {
                    read: {
                        url: "api/products",
                        dataType: "json",
                        contentType: 'application/json; charset=utf-8',
                        type: 'GET'
                    }
                },
                error: function (e) {
                    alert(e.errorThrown);
                },
                change: function () {
                    $("#container2 > ul").html(kendo.render(producatsTemplate1, this.view()));
                }
            });
            productsDataSource.read();
        });
    </script>
```

ابتدا یک div با id مساوی container2 جهت تعیین محل نهایی رندر قالب template1 در صفحه تعریف می‌شود.
هرچند خروجی دریافتی از سرور نهایتا یک آرایه از اشیاء Product است، اما در template1 اثری از حلقه‌ی جاوا اسکریپتی مشاهده نمی‌شود. در اینجا چون از متد kendo.render استفاده می‌شود، نیازی به ذکر حلقه نیست و به صورت خودکار، به تعداد

عناصر آرایه دریافتی از سرور، قطعه HTML قالب را تکرار می‌کند.
در ادامه برای کار با سرور از یک Kendo UI DataSource استفاده شده‌است. قسمت transport/read آن، کار تعریف محل دریافت اطلاعات را از سرور مشخص می‌کند. رویدادگران change آن اطلاعات نهایی دریافتی را توسط متد view در اختیار متد kendo.render قرار می‌دهد. در نهایت، قطعه‌ی HTML رندر شده‌ی نهایی حاصل از اجرای قالب، در بین تگ‌های ul مربوط به container2 درج خواهد شد.
رویدادگران change زمانیکه data source، از اطلاعات راه دور و یا یک آرایه‌ی جاوا اسکریپتی پر می‌شود، فراخوانی خواهد شد. همچنین مباحث مرتب سازی اطلاعات، صفحه بندی و تغییر صفحه، افزودن، ویرایش و یا حذف اطلاعات نیز سبب فراخوانی آن می‌گردند. متد view ایی که در این مثال فراخوانی شد، صرفا در روال رویدادگردان change دارای اعتبار است و آخرین تغییرات اطلاعات و آیتم‌های موجود در data source را باز می‌گرداند.

یک نکته‌ی تکمیلی: فعال سازی intellisense کدهای جاوا اسکریپتی Kendo UI

اگر به پوشه‌ی اصلی مجموعه‌ی Kendo UI مراجعه کنید، یکی از آن‌ها vsdoc نام دارد که داخل آن فایل‌های min.intellisense.js و vsdoc.js مشهود هستند.
اگر از ویژوال استودیوهای قبل از 2012 استفاده می‌کنید، نیاز است فایل‌های vsdoc.js متناظری را به پروژه اضافه نمائید؛ دقیقا در کنار فایل‌های اصلی js موجود. اگر از ویژوال استودیوی 2012 و یا بالاتر استفاده می‌کنید باید از فایل‌های intellisense.js متناظر استفاده کنید. برای مثال اگر از kendo.all.min.js کمک می‌گیرید، فایل متناظر با آن kendo.all.min.intellisense.js خواهد بود.
بعد از اینکار نیاز است فایلی به نام _references.js را به پوشه‌ی اسکریپت‌های خود با این محتوا اضافه کنید (برای VS 2012 به بعد):

```
/// <reference path="jquery.min.js" />
/// <reference path="kendo.all.min.js" />
```

نکته‌ی مهم اینجا است که این فایل به صورت پیش فرض از مسیر /Scripts/_references.js ~ خوانده می‌شود. برای اضافه کردن مسیر دیگری مانند /js/_references.js ~ باید آن‌را به تنظیمات ذیل اضافه کنید:

```
Tools menu -> Options -> Text Editor -> JavaScript -> Intellisense -> References
```

گزینه‌ی Reference Group را به Implicit (Web) تغییر داده و سپس مسیر جدیدی را اضافه نمائید.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:
KendoUI04.zip

عنوان: فرمت کردن اطلاعات نمایش داده شده به کمک Kendo UI Grid
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۹ ۱۳:۲۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Kendo UI

پیشنیازهای بحث:

- « صفحه بندی، مرتب سازی و جستجوی پویای اطلاعات به کمک Kendo UI Grid »
- « استفاده از Kendo UI templates »

صورت مساله

می‌خواهیم به یک چنین تصویری برسیم؛ که دارای گروه بندی اطلاعات است، فرمت شرطی روی ستون قیمت آن اعمال شده و تاریخ نمایش داده شده در آن نیز شمسی است. همچنین برای مثال ستون قیمت آن دارای ته جمع صفحه بوده و به علاوه یک دکمه‌ی سفارشی به نوار ابزار آن اضافه شده‌است.

مباحث قسمت سمت سرور این مثال با مطلب « صفحه بندی، مرتب سازی و جستجوی پویای اطلاعات به کمک Kendo UI Grid » دقیقا یکی است. فقط یک خاصیت AddDate نیز در اینجا اضافه شده‌است.

تغییر نحوه‌ی نمایش pager

اگر به قسمت pager تصویر فوق دقت کنید، یک دکمه‌ی refresh، تعداد موارد هر صفحه و امکان وارد کردن دستی شماره صفحه، در آن پیش بینی شده‌است. این موارد را با تنظیمات ذیل می‌توان فعال کرد:

```
$("#report-grid").kendoGrid({
                // ...
                pageable: {
                    previousNext: true, // default true
                    numeric: true, // default true
                    buttonCount: 5, // default 10
                    refresh: true, // default false
                    input: true, // default false
                    pageSizes: true // default false
                },
```

بومی سازی پیغام‌های گرید

پیغام‌های فارسی را که در تصویر فوق مشاهده می‌کنید، حاصل پیوست فایل kendo.fa-IR.js هستند:

```
<!--https://github.com/loudenvier/kendo-global/blob/master/lang/kendo.fa-IR.js-->
<script src="js/messages/kendo.fa-IR.js" type="text/javascript"></script>
```

گروه بندی اطلاعات

برای گروه بندی اطلاعات در Kendo UI Grid دو قسمت باید تغییر کنند.
ابتدا باید فیلد پیش فرض گروه بندی در قسمت data source گرید تعریف شود:

```
var productsDataSource = new kendo.data.DataSource({
                // ...
                group: { field: "IsAvailable" },
                // ...
            });
```

همین تنظیم، گروه بندی را فعال خواهد کرد. اگر علاقمند باشید که به کاربران امکان تغییر دستی گروه بندی را بدهید، خاصیت groupable را نیز true کنید.

```
$("#report-grid").kendoGrid({
// ...
groupable: true, // allows the user to alter what field the grid is grouped by
// ...
```

در این حالت با کشیدن و رها کردن یک سرستون، به نوار ابزار مرتبط با گروه بندی، گروه بندی گرید بر اساس این فیلد انتخابی به صورت خودکار انجام می‌شود.

اضافه کردن ته جمع‌های ستون‌ها

این ته جمع‌ها که aggregate نام دارند باید در دو قسمت فعال شوند:

```
var productsDataSource = new kendo.data.DataSource({
                //...
                aggregate: [
                    { field: "Name", aggregate: "count" },
                    { field: "Price", aggregate: "sum" }
                ]
                //...
            });
```

ابتدا در قسمت data source مشخص می‌کنیم که چه تابع تجمعی قرار است به ازای یک فیلد خاص استفاده شود.
سپس این متدها را می‌توان مطابق فرمت hash syntax قالب‌های Kendo UI در قسمت footerTemplate هر ستون تعریف کرد:

```
$("#report-grid").kendoGrid({
                // ...
                columns: [
                    {
                        field: "Name", title: "نام محصول",
                        footerTemplate: "تعداد: #=count#"
                    },
                    {
                        field: "Price", title: "قیمت",
                        footerTemplate: "جمع: #=kendo.toString(sum,'c0')#"
                    }
                ]
                // ...
            });
```

فرمت شرطی اطلاعات

در ستون قیمت، می‌خواهیم اگر قیمتی بیش از 2490 بود، با پس زمینه‌ی قهوه‌ای و رنگ زرد نمایش داده شود. برای این منظور می‌توان یک قالب Kendo UI سفارشی را طراحی کرد:

```
<script type="text/x-kendo-template" id="priceTemplate">
        #if( Price > 2490 ) {#
        <span style="background:brown; color:yellow;">#=kendo.toString(Price,'c0')#</span>
        #} else {#
        #= kendo.toString(Price,'c0')#
        #}#
    </script>
```

سپس نحوه‌ی استفاده‌ی از آن به صورت ذیل خواهد بود:

```
$("#report-grid").kendoGrid({
                //...
                columns: [
                    {
                        field: "Price", title: "قیمت",
                        template: kendo.template($("#priceTemplate").html()),
                        footerTemplate: "جمع: #=kendo.toString(sum,'c0')#"
                    }
                ]
                //...
            });
```

توسط متد kendo.template امکان انتساب یک قالب سفارشی به خاصیت template یک ستون وجود دارد.

فرمت تاریخ میلادی به شمسی در حین نمایش

برای تبدیل سمت کلاینت تاریخ میلادی به شمسی از کتابخانه‌ی moment-jalaali.js کمک گرفته شده‌است:

```
<!--https://github.com/moment/moment/-->
<script src="js/cultures/moment.min.js" type="text/javascript"></script>
<!--https://github.com/jalaali/moment-jalaali-->
<script src="js/cultures/moment-jalaali.js" type="text/javascript"></script>
```

پس از آن تنها کافی است متد فرمت این کتابخانه را در قسمت template ستون تاریخ و توسط hash syntax قالب‌های Kendo UI بکار برد:

```
$("#report-grid").kendoGrid({
                //...
                columns: [
                    {
                        field: "AddDate", title: "تاریخ ثبت",
                        template: "#=moment(AddDate).format('jYYYY/jMM/jDD')#"
                    }
                ]
                //...
            });
```

اضافه کردن یک دکمه به نوار ابزار گرید

نوار ابزار Kendo UI Grid را نیز می‌توان توسط یک قالب سفارشی آن مقدار دهی کرد:

```
$("#report-grid").kendoGrid({
                // ...
                toolbar: [
                    { template: kendo.template($("#toolbarTemplate").html()) }
                ]
                // ...
            });
```

برای نمونه toolbarTemplate فوق را به نحو ذیل تعریف کرده‌ایم:

```
<script>
        // این اطلاعات برای تهیه خروجی سمت سرور مناسب هستند
        function getCurrentGridFilters() {
            var dataSource = $("#report-grid").data("kendoGrid").dataSource;
            var gridState = {
                page: dataSource.page(),
                pageSize: dataSource.pageSize(),
                sort: dataSource.sort(),
                group: dataSource.group(),
                filter: dataSource.filter()
            };
            return kendo.stringify(gridState);
        }
    </script>

    <script id="toolbarTemplate" type="text/x-kendo-template">
        <a class="k-button" href="\#" onclick="alert('gridState: ' + getCurrentGridFilters());">نوار
ابزار سفارشی</a>
    </script>
```

دکمه‌ی اضافه شده، وضعیت فیلتر data source متصل به گرید را بازگشت می‌دهد. برای مثال مشخص می‌کند که در چه صفحه‌ای با چه تعداد رکورد قرار داریم و همچنین وضعیت مرتب سازی، فیلتر و غیره چیست. از این اطلاعات می‌توان در سمت سرور برای تهیه‌ی خروجی‌های PDF یا اکسل استفاده کرد. وضعیت فیلتر اطلاعات مشخص است. بر همین مبنا کوئری گرفته و سپس می‌توان نتیجه‌ی آن‌را تبدیل به منبع داده تهیه خروجی مورد نظر کرد.

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید:

KendoUI05.zip

نظرات خوانندگان

نویسنده: سروش
تاریخ: ۱۳۹۳/۰۸/۲۰ ۸:۵۱

با سلام هنگام Bulid پروژه با خطای زیر مواجه می‌شم

Error 1 Unable to locate 'C:\Users\Administrator\Desktop\KendoUI05\KendoUI05\.nuget\NuGet.exe' KendoUI05

دلیلش چیه ؟ البته Update-Package -Reinstall را انجام داده ام
متشکرم از شما

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۰ ۹:۵

- روی solution کلیک راست کنید و گزینه‌ی Enable NuGet Package Restore را انتخاب کنید.
- یا فایل NuGet.targets پوشه‌ی nuget. را باز کرده و دریافت خودکار nuget.exe را فعال کنید:

```
تغییر از
<DownloadNuGetExe Condition=" '$(DownloadNuGetExe)' == '' ">false</DownloadNuGetExe>
به
<DownloadNuGetExe Condition=" '$(DownloadNuGetExe)' == '' ">true</DownloadNuGetExe>
```

- و یا فایل nuget.exe را از این آدرس دریافت کنید و در پوشه‌ی nuget. کپی کنید.

نویسنده: سروش
تاریخ: ۱۳۹۳/۰۸/۲۰ ۹:۱۵

مرسی. فقط برای تغییر قالب KendoUi میشه کاری کرد برای ترکیب رنگهاش ؟ مثل JqueryUi Custom

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۰ ۹:۳۲

در انتهای مطلب «بررسی ساختار ویجت‌های وب Kendo UI » قسمت «تغییر قالب ویجت‌های وب»، توضیح داده شده.

نویسنده: وحید محمّدطاهری
تاریخ: ۱۳۹۳/۱۰/۰۹ ۱۷:۳۰

با سلام و خدا قوت

وقتی تو تنظیمات دیتاسورس serverGrouping:true تنظیم می‌کنم با خطای e.slice مواجه میشم.
تو مثالی که شما قراردادید هم اینو وقتی تنظیم کردم باز همین مشکل بوجود اومد.

ممنون میشم راهنماییم کنید.

نویسنده: شاکری
تاریخ: ۱۳۹۳/۱۰/۱۳ ۱۲:۳

سلام
من برای نمایش تاریخ شمسی کالچر رو تغییر میدم اما تاریخ‌ها در گرید تغییری نمیکنه در صورتی که اگه مثلا به صورت

```
@DateTime.Now
```

در خارج از گرید تاریخ رو نمایش بدم تاریخ شمسی نمایش داده میشه!
شما برای نمایش تاریخ شمسی در گرید کندو از چه روشی استفاده می‌کنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۳ ۱۲:۱۸

- خروجی تاریخ JSON استاندارد ، نمی‌تواند شمسی باشد و Kendo UI اطلاعات خودش را به فرمت JSON دریافت می‌کند. اطلاعاتی که به کمک Razor نمایش دادید، نهایتا یک رشته معمولی است و نه یک تاریخ استاندارد ISO 8601 که توسط JSON.NET بازگشت داده می‌شود.

- امکان « تهیه یک JsonConverter سفارشی » در JSON.NET وجود دارد؛ مثلا تاریخ را تبدیل به یک رشته‌ی دلخواه کنید و بازگشت دهید.

+ قسمت «فرمت تاریخ میلادی به شمسی در حین نمایش» را در متن فوق مطالعه کنید. این تبدیل سمت کلاینت هست و نه سمت سرور (به کمک کتابخانه‌ی moment-jalaali.js).

عنوان: فعال سازی عملیات CRUD در Kendo UI Grid
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۱ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.NET Web API, jQuery, Kendo UI

پیشنیاز بحث

- « فرمت کردن اطلاعات نمایش داده شده به کمک Kendo UI Grid »

Kendo UI Grid دارای امکانات ثبت، ویرایش و حذف توکاری است که در ادامه نحوه‌ی فعال سازی آن‌ها را بررسی خواهیم کرد. مثالی که در ادامه بررسی خواهد شد، در تکمیل مطلب « فرمت کردن اطلاعات نمایش داده شده به کمک Kendo UI Grid » است.

تنظیمات Data Source سمت کاربر

برای فعال سازی صفحه بندی سمت سرور ، با قسمت read منبع داده Kendo UI پیشتر آشنا شده بودیم. جهت فعال سازی قسمت‌های ثبت اطلاعات جدید (create)، به روز رسانی رکوردهای موجود (update) و حذف ردیفی مشخص (destroy) نیاز است تعاریف قسمت‌های متناظر را که هر کدام به آدرس مشخصی در سمت سرور اشاره می‌کنند، اضافه کنیم:

```
var productsDataSource = new kendo.data.DataSource({
                transport: {
                    read: {
                        url: "api/products",
                        dataType: "json",
                        contentType: 'application/json; charset=utf-8',
                        type: 'GET'
                    },
                    create: {
                        url: "api/products",
                        contentType: 'application/json; charset=utf-8',
                        type: "POST"
                    },
```

```
                    update: {
                        url: function (product) {
                            return "api/products/" + product.Id;
                        },
                        contentType: 'application/json; charset=utf-8',
                        type: "PUT"
                    },
                    destroy: {
                        url: function (product) {
                            return "api/products/" + product.Id;
                        },
                        contentType: 'application/json; charset=utf-8',
                        type: "DELETE"
                    },
                    //...
                },
                schema: {
                    //...
                    model: {
                        id: "Id", // define the model of the data source. Required for validation and
property types.
                        fields: {
                            "Id": { type: "number", editable: false }, //تعیین نوع فیلد برای جستجوی پویا
مهم است
                            "Name": { type: "string", validation: { required: true } },
                            "IsAvailable": { type: "boolean" },
                            "Price": { type: "number", validation: { required: true, min: 1 } },
                            "AddDate": { type: "date", validation: { required: true } }
                        }
                    }
                },
                batch: false, // enable batch editing - changes will be saved when the user clicks the
"Save changes" button
                //...
            });
```

- همانطور که ملاحظه می‌کنید، حالت‌های update و destroy بر اساس Id ردیف انتخابی کار می‌کنند. این Id را باید در قسمت model مربوط به اسکیمای تعریف شده، دقیقا مشخص کرد. عدم تعریف فیلد id، سبب خواهد شد تا عملیات update نیز در حالت create تفسیر شود.

- به علاوه در اینجا به ازای هر فیلد، مباحث اعتبارسنجی نیز اضافه شده‌اند؛ برای مثال فیلدهای اجباری با required: true مشخص گردیده‌اند.

- اگر فیلدی نباید ویرایش شود (مانند فیلد Id)، خاصیت editable آن‌را false کنید.

- در data source امکان تعریف خاصیتی به نام batch نیز وجود دارد. حالت پیش فرض آن false است. به این معنا که در حالت ویرایش، تغییرات هر ردیفی، یک درخواست مجزا را به سمت سرور سبب خواهد شد. اگر آن‌را true کنید، تغییرات تمام ردیف‌ها در طی یک درخواست به سمت سرور ارسال می‌شوند. در این حالت باید به خاطر داشت که پارامترهای سمت سرور، از حالت یک شیء مشخص باید به لیستی از آن‌ها تغییر یابند.

مدیریت سمت سرور ثبت، ویرایش و حذف اطلاعات

در حالت ثبت، متد Post، توسط آدرس مشخص شده در قسمت create منبع داده گرید، فراخوانی می‌گردد:

```
namespace KendoUI06.Controllers
{
    public class ProductsController : ApiController
    {
        public HttpResponseMessage Post(Product product)
        {
            if (!ModelState.IsValid)
                return Request.CreateResponse(HttpStatusCode.BadRequest);

            var id = 1;
            var lastItem = ProductDataSource.LatestProducts.LastOrDefault();
            if (lastItem != null)
            {
                id = lastItem.Id + 1;
            }
            product.Id = id;
            ProductDataSource.LatestProducts.Add(product);
```

```
            var response = Request.CreateResponse(HttpStatusCode.Created, product);
            response.Headers.Location = new Uri(Url.Link("DefaultApi", new { id = product.Id }));
            // گرید آی دی جدید را به این صورت دریافت می‌کند
            response.Content = new ObjectContent<DataSourceResult>(
                new DataSourceResult { Data = new[] { product } }, new JsonMediaTypeFormatter());
            return response;
        }
    }
}
```

نکته‌ی مهمی که در اینجا باید به آن دقت داشت، نحوه‌ی بازگشت Id رکورد جدید ثبت شده‌است. در این مثال، قسمت schema منبع داده سمت کاربر به نحو ذیل تعریف شده‌است:

```
var productsDataSource = new kendo.data.DataSource({
                //...
                schema: {
                    data: "Data",
                    total: "Total",
                }
                //...
            });
```

از این جهت که خروجی متد Get بازگرداننده‌ی اطلاعات صفحه بندی شده ، از نوع DataSourceResult است و این نوع، دارای خواصی مانند Data، Total و Aggergate است:

```
namespace KendoUI06.Controllers
{
    public class ProductsController : ApiController
    {
        public DataSourceResult Get(HttpRequestMessage requestMessage)
        {
            var request = JsonConvert.DeserializeObject<DataSourceRequest>(
                requestMessage.RequestUri.ParseQueryString().GetKey(0)
            );

            var list = ProductDataSource.LatestProducts;
            return list.AsQueryable()
                       .ToDataSourceResult(request.Take, request.Skip, request.Sort, request.Filter);
        }
    }
}
```

بنابراین در متد Post نیز باید بر این اساس، response.Content را از نوع لیستی از DataSourceResult تعریف کرد تا Kendo UI Grid بداند که Id رکورد جدید را باید از فیلد Data، همانند تنظیمات schema منبع داده خود، دریافت کند.

```
response.Content = new ObjectContent<DataSourceResult>(
                                new DataSourceResult { Data = new[] { product } }, new
JsonMediaTypeFormatter());
```

اگر این تنظیم صورت نگیرد، Id رکورد جدید را در گرید، مساوی صفر مشاهده خواهید کرد و عملا بدون استفاده خواهد شد؛ زیرا قابلیت ویرایش و حذف خود را از دست می‌دهد.

متدهای حذف و به روز رسانی سمت سرور نیز چنین امضایی را خواهند داشت:

```
namespace KendoUI06.Controllers
{
    public class ProductsController : ApiController
    {
        public HttpResponseMessage Delete(int id)
        {
            var item = ProductDataSource.LatestProducts.FirstOrDefault(x => x.Id == id);
            if (item == null)
                return Request.CreateResponse(HttpStatusCode.NotFound);
```

```
            ProductDataSource.LatestProducts.Remove(item);

            return Request.CreateResponse(HttpStatusCode.OK, item);
        }

        [HttpPut] // Add it to fix this error: The requested resource does not support http method
'PUT'
        public HttpResponseMessage Update(int id, Product product)
        {
            var item = ProductDataSource.LatestProducts
                                        .Select(
                                            (prod, index) =>
                                                new
                                                {
                                                    Item = prod,
                                                    Index = index
                                                })
                                        .FirstOrDefault(x => x.Item.Id == id);
            if (item == null)
                return Request.CreateResponse(HttpStatusCode.NotFound);


            if (!ModelState.IsValid || id != product.Id)
                return Request.CreateResponse(HttpStatusCode.BadRequest);

            ProductDataSource.LatestProducts[item.Index] = product;
            return Request.CreateResponse(HttpStatusCode.OK);
        }
    }
}
```

حالت Update از HTTP Verb خاصی به نام Put استفاده می‌کند و ممکن است در این بین خطای The requested resource does not support http method 'PUT' را دریافت کنید. برای رفع آن ابتدا بررسی کنید که آیا Web.config برنامه دارای تعاریف ExtensionlessUrlHandler هست یا خیر. همچنین مزین کردن این متد با ویژگی HttpPut، مشکل را برطرف می‌کند.

تنظیمات Kendo UI Grid جهت فعال سازی CRUD

در ادامه کلیه تغییرات مورد نیاز جهت فعال سازی CRUD را در Kendo UI، به همراه مباحث بومی سازی عبارات متناظر با دکمه‌ها و صفحات خودکار مرتبط، مشاهده می‌کنید:

```
$("#report-grid").kendoGrid({
                //....
                editable: {
                    confirmation: "آیا مایل به حذف ردیف انتخابی هستید؟",
                    destroy: true, // whether or not to delete item when button is clicked
                    mode: "popup", // options are "incell", "inline", and "popup"
                    //template: kendo.template($("#popupEditorTemplate").html()), // template to use
for pop-up editing
                    update: true, // switch item to edit mode when clicked?
                    window: {
                        title: "مشخصات محصول"   // Localization for Edit in the popup window
                    }
                },
                columns: [
                //....
                    {
                        command: [
                            { name: "edit", text: "ویرایش" },
                            { name: "destroy", text: "حذف" }
                        ],
                        title: " ", width: "160px"
                    }
                ],
                toolbar: [
                    { name: "create", text: "افزودن ردیف جدید" },
                    { name: "save", text: "ذخیره‌ی تمامی تغییرات" },
                    { name: "cancel", text: "لغو کلیه‌ی تغییرات" },
                    { template: kendo.template($("#toolbarTemplate").html()) }
                ],
                messages: {
                    editable: {
                        cancelDelete: "لغو",
                        confirmation: "آیا مایل به حذف این رکورد هستید؟",
```

```
                    confirmDelete: "حذف"
                },
                commands: {
                    create: "افزودن ردیف جدید",
                    cancel: "لغو کلیه‌ی تغییرات",
                    save: "ذخیره‌ی تمامی تغییرات",
                    destroy: "حذف",
                    edit: "ویرایش",
                    update: "ثبت",
                    canceledit: "لغو"
                }
            }
        });
```

- ساده‌ترین حالت CRUD در Kendo UI با مقدار دهی خاصیت editable آن به true آغاز می‌شود. در این حالت، ویرایش درون سلولی یا incell فعال خواهد شد که مباحث batching ابتدای بحث، فقط در این حالت کار می‌کند. زمانیکه incell editing فعال است، کاربر می‌تواند تمام ردیف‌ها را ویرایش کرده و در آخر کار بر روی دکمه‌ی «ذخیره‌ی تمامی تغییرات» موجود در نوار ابزار، کلیک کند. در سایر حالات، هر بار تنها یک ردیف را می‌توان ویرایش کرد.
- برای فعال سازی تولید صفحات خودکار ویرایش و افزودن ردیف‌ها، نیاز است خاصیت editable را به نحوی که ملاحظه می‌کنید، مقدار دهی کرد. خاصیت mode آن سه حالت incell (پیش فرض)، inline و popup را پشتیبانی می‌کند.
- اگر حالت‌های inline و یا popup را فعال کردید، در انتهای ستون‌های تعریف شده، نیاز است ستون ویژه‌ای به نام command را مطابق تعاریف فوق، تعریف کنید. در این حالت دو دکمه‌ی ویرایش و ثبت، فعال می‌شوند و اطلاعات خود را از تنظیمات data source گرید دریافت می‌کنند. دکمه‌ی ویرایش در حالت incell کاربردی ندارد (چون در این حالت کاربر با کلیک درون یک سلول می‌تواند آن‌را مانند برنامه‌ی اکسل ویرایش کند). اما دکمه‌ی حذف در هر سه حالت قابل استفاده است.
- به نوار ابزار گرید، سه دکمه‌ی افزودن ردیف‌های جدید، ذخیره‌ی تمامی تغییرات و لغو تغییرات صورت گرفته، اضافه شده‌اند. این دکمه‌ها استاندارد بوده و در اینجا نحوه‌ی بومی سازی پیام‌های مرتبط را نیز مشاهده می‌کنید. همانطور که عنوان شد، دکمه‌های «تمامی تغییرات» در حالت فعال سازی batching در منبع داده و استفاده از incell editing معنا پیدا می‌کند. در سایر حالات این دو دکمه کاربردی ندارند. اما دکمه‌ی افزودن ردیف‌های جدید در هر سه حالت کاربرد دارد و یکسان است.

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید

KendoUI06.zip

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۱ ۹:۴۵

یک نکته‌ی تکمیلی

در مثال فوق از ASP.NET Web API استفاده شده‌است. اگر علاقمند به استفاده از WCF و یا حتی فایل‌های asmx قدیمی هم باشید، اینکار میسر است. مثال‌هایی را در این زمینه، در اینجا می‌توانید مشاهده کنید.

نویسنده: شروین ایرانی
تاریخ: ۱۳۹۳/۱۱/۱۹ ۸:۷

در بحث Grid در پلتفرم KendoUI آیا راهی هست که بتونیم بوسیله کوکی براحتی آخرین وضعیت گرید را ذخیره کنیم تا در مراجعه بعدی بصورت اتوماتیک وضعیت قبلی را لود کنه؟ متشکرم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۱۹ ۱۰:۳

Preserve Grid state in a cookie

فایل آن: PreserveState.zip + یک مثال

نویسنده: ژوپیتر
تاریخ: ۱۳۹۳/۱۱/۱۹ ۱۰:۳۳

سلام،

هنگام تغییر خطای زیر را دریافت می‌کنم، هر چند تغییرات ذخیره می‌شوند و فقط این خطا بی‌جهت داده می‌شود. از این روش‌ها (+ و +) برای دریافت اطلاعات استفاده کردم. به نظر شما مشکل کجاست و یا چطور می‌شه دیباگ کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۲۰ ۱۱:۱۲

مثال فوق را (KendoUI06) اگر برای ASP.NET MVC بازنویسی کنیم به کدهای ذیل خواهیم رسید:

```
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Mvc;
using Kendo.DynamicLinq;
using KendoUI06Mvc.Models;
using Newtonsoft.Json;

namespace KendoUI06Mvc.Controllers
{
    public class HomeController : Controller
    {
        public ActionResult Index()
        {
            return View(); // shows the page.
        }

        [HttpDelete]
        public ActionResult DeleteProduct(int id)
        {
            var item = ProductDataSource.LatestProducts.FirstOrDefault(x => x.Id == id);
            if (item == null)
                return new HttpNotFoundResult();

            ProductDataSource.LatestProducts.Remove(item);
```

```
            return Json(item);
        }

        [HttpGet]
        public ActionResult GetProducts()
        {
            var request = JsonConvert.DeserializeObject<DataSourceRequest>(
               this.Request.Url.ParseQueryString().GetKey(0)
            );

            var list = ProductDataSource.LatestProducts;
            return Json(list.AsQueryable()
                       .ToDataSourceResult(request.Take, request.Skip, request.Sort, request.Filter),
                       JsonRequestBehavior.AllowGet);
        }

        [HttpPost]
        public ActionResult PostProduct(Product product)
        {
            if (!ModelState.IsValid)
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

            var id = 1;
            var lastItem = ProductDataSource.LatestProducts.LastOrDefault();
            if (lastItem != null)
            {
                id = lastItem.Id + 1;
            }
            product.Id = id;
            ProductDataSource.LatestProducts.Add(product);

            // گرید آی دی جدید را به این صورت دریافت می‌کند
            return Json(new DataSourceResult { Data = new[] { product } });
        }

        [HttpPut] // Add it to fix this error: The requested resource does not support http method
'PUT'
        public ActionResult UpdateProduct(int id, Product product)
        {
            var item = ProductDataSource.LatestProducts
                                        .Select(
                                            (prod, index) =>
                                                new
                                                {
                                                    Item = prod,
                                                    Index = index
                                                })
                                        .FirstOrDefault(x => x.Item.Id == id);
            if (item == null)
                return new HttpNotFoundResult();


            if (!ModelState.IsValid || id != product.Id)
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

            ProductDataSource.LatestProducts[item.Index] = product;

            //Return HttpStatusCode.OK
            return new HttpStatusCodeResult(HttpStatusCode.OK);
        }
    }
}
```

در این حالت View برنامه فقط جهت ذکر آدرس‌های جدید باید اصلاح شود و نیاز به تغییر دیگری ندارد:

```
var productsDataSource = new kendo.data.DataSource({
            transport: {
                read: {
                    url: "@Url.Action("GetProducts","Home")",
                    dataType: "json",
                    contentType: 'application/json; charset=utf-8',
                    type: 'GET'
                },
                create: {
                    url: "@Url.Action("PostProduct","Home")",
                    contentType: 'application/json; charset=utf-8',
                    type: "POST"
```

```
                },
                update: {
                    url: function (product) {
                        return "@Url.Action("UpdateProduct","Home")/" + product.Id;
                        },
                        contentType: 'application/json; charset=utf-8',
                        type: "PUT"
                    },
                    destroy: {
                        url: function (product) {
                            return "@Url.Action("DeleteProduct","Home")/" + product.Id;
                        },
                        contentType: 'application/json; charset=utf-8',
                        type: "DELETE"
                    },
                    parameterMap: function (options) {
                        return kendo.stringify(options);
                    }
                },
```

نویسنده: ژوپیتر
تاریخ: ۱۳۹۳/۱۱/۲۰ ۱۴:۴۱

برای کسانی که از روش GitHub لینک داده شده استفاده کردند و مشکل بنده رو هنگام Update اطلاعات دارند: در ActionResult مربوط به Update گریدویو Kendu UI هنگام بازگشت مقدار Json به صورت null باید از عبارت رشته‌ای خالی شبیه زیر استفاده کنیم:

```
[HttpPost]
        public ActionResult Update(IEnumerable<Product> products)
        {
                //  ....
                //Return emtpy result
                return Json("");
        }
```

موفق باشید.

عنوان: اعتبار سنجی ورودی‌های کاربر در Kendo UI
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۲ ۱۵:۱۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Kendo UI

در مطلب « فعال سازی عملیات CRUD در Kendo UI Grid » با نحوه‌ی تعریف مقدماتی اعتبارسنجی فیلدهای تعریف شده، آشنا شدید:

```
fields: {
      "Price": { type: "number", validation: { required: true, min: 1 } }
}
```

در ادامه نگاهی خواهیم داشت به جزئیات تکمیلی امکانات اعتبارسنجی ورودی‌های کاربر در Kendo UI.

Kendo UI Validation و HTML 5

در HTML 5 امکان تعریف نوع‌های خاص کنترل‌های ورودی کاربر مانند search، date، range، number، url، email و color وجود دارد. برای مثال در اینجا اگر کاربر تاریخ غیرمعتبری را وارد کند، مرورگر پیام اعتبارسنجی متناظری را به او نمایش خواهد داد.
همچنین در HTML 5 امکان افزودن ویژگی required نیز به کنترل‌های ورودی پیش بینی شده‌است. اما باید درنظر داشت که مرورگرهای قدیمی از این امکانات پشتیبانی نمی‌کنند. در این حالت Kendo UI با تشویق استفاده از روش معرفی شده در HTML 5، با آن یکپارچه شده و همچنین این قابلیت‌های اعتبارسنجی HTML 5 را در مرورگرهای قدیمی نیز میسر می‌کند. Kendo UI Validation جزو نسخه‌ی سورس باز Kendo UI با مجوز Apache نیز می‌باشد.
نمونه‌ای از امکانات اعتبارسنجی توکار HTML 5 را در اینجا مشاهده می‌کنید:

```
<input type="text" name="firstName" required />
<input type="text" name="twitter" pattern="https?://(?:www\.)?twitter\.com/.+i" />
<input type="number" name="age" min="1" max="42" />
<input type="number" name="age" min="1" max="100" step="2" />
<input type="url" name="url" />
<input type="email" name="email" />
```

یکپارچه سازی اعتبارسنجی Kendo UI با اعتبارسنجی HTML 5

در اینجا یک فرم تشکیل شده با ساختار HTML 5 را ملاحظه می‌کنید. هر دو فیلد ورودی، با ویژگی استاندارد required مزین شده‌اند. همچنین توسط ویژگی type، ورودی دوم جهت دریافت آدرس ایمیل معرفی شده‌است.
چون فیلد دوم دارای دو اعتبارسنجی تعریف شده است، دارای دو ویژگی data-* برای تعریف پیام‌های اعتبارسنجی متناظر نیز می‌باشد. الگوی تعریف آن‌ها data-[rule]-msg است.

```
<div class="k-rtl">
        <form id="testView">
            <label for="firstName">نام</label>
            <input id="firstName"
                   name="firstName"
                   type="text"
                   class="k-textbox"
                   required
                   validationmessage="لطفا نامی را وارد کنید">
            <br>
            <label for="emailId">آدرس پست الکترونیک</label>
            <input id="emailId"
                   name="emailId"
                   type="email"
                   dir="ltr"
                   required
                   class="k-textbox"
                   data-required-msg="لطفا ایمیلی را وارد کنید."
                   data-email-msg="ایمیل وارد شده معتبر نیست.">
            <br>
```

```
            <input type="submit" class="k-button" value="ارسال">
        </form>
    </div>

    <script type="text/javascript">
        $(function () {
            $("form#testView").kendoValidator();
        });
    </script>
```

تنها کاری که جهت یکپارچه سازی امکانات اعتبارسنجی Kendo UI با اعتبارسنجی استاندارد HTML 5 باید انجام داد، فراخوانی متد kendoValidator بر روی ناحیه‌ی مشخص شده است.

تعیین محل نمایش پیام‌های اعتبارسنجی

پیام‌های اعتبارسنجی Kendo UI به صورت خودکار در کنار فیلد متناظر با آن نمایش داده می‌شوند. اما اگر نیاز به تعیین مکان دستی آن‌ها وجود داشت (جهت خوانایی بهتر) باید به نحو ذیل عمل کرد:

```
<input type="text" id="name" name="name" required>
     <span class="k-invalid-msg" data-for="name"></span>
```

در اینجا span با کلاس k-invalid-msg و ویژگی data-for که به name کنترل ورودی اشاره می‌کند، محل نمایش پیام اعتبارسنجی متناظر با فیلد name خواهد بود.

تعریف سراسری پیام‌های اعتبارسنجی

در مثال فوق، به ازای تک تک فیلدهای ورودی، پیام اعتبارسنجی متناظر با required وارد شد. می‌توان این پیام‌ها را حذف کرد و در قسمت messages متد kendoValidator قرار داد:

```
<script type="text/javascript">
        $(function () {
            $("form#testView").kendoValidator({
                messages: {
                    // {0} would be replaced with the input element's name
                    required: '{0} را تکمیل کنید.',
                    email: 'ایمیل وارد شده معتبر نیست.'
                }
            });
        });
    </script>
```

- به این صورت پیام‌های اعتبارسنجی required و email، به صورت یکسانی به تمام المان‌های دارای این ویژگی‌ها اعمال خواهند شد.
- در این پیام‌ها {0} با مقدار ویژگی name فیلد ورودی متناظر جایگزین می‌شود.
- اگر هم در markup و هم در تعاریف kendoValidator، پیام‌های اعتبارسنجی تعریف شوند، حق تقدم با تعاریف markup خواهد بود.

اعتبارسنجی سفارشی سمت کاربر

علاوه بر امکانات استاندارد HTML 5، امکان تعریف دستورهای اعتبارسنجی سفارشی نیز وجود دارد:

```
<script type="text/javascript">
        $(function () {
            $("form#testView").kendoValidator({
                rules: {
                    customRule1: function (input) {
                        if (!input.is("[id=firstName]"))
                            return true;

                        var re = /^[A-Za-z]+$/;
                        return re.test(input.val());
                    }
                   //, customRule1: ….
                },
                messages: {
                    // {0} would be replaced with the input element's name
                    required: '{0} را تکمیل کنید.',
                    email: 'ایمیل وارد شده معتبر نیست.',
                    customRule1: 'اعداد مجاز نیستند.'
                }
            });
        });
    </script>
```

- همانطور که ملاحظه می‌کنید، برای تعریف منطق اعتبارسنجی سفارشی، باید از خاصیت rules ورودی متد kendoValidator شروع کرد. در اینجا نام یک متد callback دلخواهی را وارد کرده و سپس بر اساس منطق اعتبارسنجی مورد نظر، باید true/false را بازگشت داد. برای نمونه در این مثال اگر کاربر در فیلد نام، عدد وارد کند، ورودی او مورد قبول واقع نخواهد شد.
- باید دقت داشت که اگر بررسی input.is صورت نگیرد، منطق تعریف شده به تمام کنترل‌های صفحه اعمال می‌شود.
- پیام متناظر با این دستور سفارشی جدید، در قسمت messages، دقیقا بر اساس نام callback method تعریف شده در قسمت rules باید تعریف شود.

فراخوانی دستی اعتبارسنجی یک فرم

در حالت پیش فرض، با کلیک بر روی دکمه‌ی ارسال، اعتبارسنجی کلیه عناصر فرم به صورت خودکار انجام می‌شود. اگر بخواهیم در این بین یک پیام سفارشی را نیز نمایش دهیم می‌توان به صورت زیر عمل کرد:

```
<script type="text/javascript">
        $(function () {
            $("form#testView").submit(function (event) {
                event.preventDefault();
                var validator = $("form#testView").data("kendoValidator");
                if (validator.validate()) {
                    alert("validated!");
                } else {
                    alert("There is invalid data in the form.");
                }
```

```
            });

            $("form#testView").kendoValidator();
        });
    </script>
```

در اینجا رخداد submit فرم بازنویسی شده و متد validate آن بر اساس kendoValidator تعریف شده، به صورت دستی فراخوانی می‌شود.

اعتبارسنجی سفارشی در DataSource

در تعریف فیلدهای مدل DataSource، امکان تعریف اعتبارسنجی‌های پیش فرضی مانند rquired، min، max و امثال آن وجود دارد که نمونه‌ای از آن‌را در بحث فعال سازی CRUD در Kendo UI Grid مشاهده کردید:

```
fields: {
   "serviceName": {
    type: "string",
    defaultValue: "Inspection",
    editable: true,
    nullable: false,
    validation: { /*...*/ }
   },
   // ...
}
```

برای تعریف اعتبارسنجی سفارشی در اینجا، همانند متد kendoValidator نیاز است یک یا چند callback متد سفارشی را طراحی کرد:

```
schema: {
            model: {
                        id: "ProductID",
                        fields: {
                                    ProductID: { editable: false, nullable: true },
                                    ProductName: {
                                        validation: {
                                            required: true,
                                            custom1: function (input) {
                                                if (input.is("[name='ProductName']") && input.val()
!= "") {
                                                    input.attr("data-custom1-msg", "نام محصول باید
با حرف بزرگ انگلیسی شروع شود");
                                                    return /^[A-Z]/.test(input.val());
                                                }

                                                return true;
                                            }
                                          // ,custom2: ...
                                        }
                                    },
                                    UnitPrice: { type: "number", validation: { required: true, min:
1} },
                                    Discontinued: { type: "boolean" },
                                    UnitsInStock: { type: "number", validation: { min: 0, required:
true} }
                                }
                            }
                        }
```

نام این متد که نهایتا true/false بر می‌گرداند، اختیاری است. نام کنترل جاری همان نام فیلد متناظر است (جهت محدود کردن بازه‌ی اعمال منطق اعتبارسنجی). برای مقدار دهی پیام اعتبارسنجی از متد input.attr و الگوی data-[validationRuleName]-msg استفاده می‌شود. ضمنا به هر تعداد لازم می‌توان در اینجا custom rule تعریف کرد.

متد input.val() مقدار کنترل جاری را بر می‌گرداند. برای دسترسی به مقدار سایر کنترل‌ها می‌توان از روش $("#fieldName").val() استفاده کرد.

نظرات خوانندگان

نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۳/۰۸/۲۲ ۱۷:۱

ممنون مطلبتون.
آیا اعتبار سنجی در حالت Inline Editing درون گرید هم به همین شکل هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۲ ۱۸:۱۱

بله. اطلاعات اعتبارسنجی فیلدهای خودش را از data source دریافت می‌کند. مثال فعال سازی CRUD در Kendo UI Grid را اجرا کنید، این مورد در آن لحاظ شده.

عنوان: Kendo UI MVVM
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۴ ۱۵:۵۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, MVVM, ASP.NET Web API, jQuery, KnockoutJS, Kendo UI

پیشنیازها

- « استفاده از Kendo UI templates »
- « اعتبار سنجی ورودی‌های کاربر در Kendo UI »
- « فعال سازی عملیات CRUD در Kendo UI Grid » جهت آشنایی با نحوه‌ی تعریف DataSource ایی که می‌تواند اطلاعات را ثبت، حذف و یا ویرایش کند.

در این مطلب قصد داریم به یک چنین صفحه‌ای برسیم که در آن در ابتدای نمایش، لیست ثبت نام‌های موجود، از سرور دریافت و توسط یک Kendo UI template نمایش داده می‌شود. سپس امکان ویرایش و حذف هر ردیف، وجود خواهد داشت، به همراه امکان افزودن ردیف‌های جدید. در این بین مدیریت نمایش لیست ثبت نام‌ها توسط امکانات binding توکار فریم ورک MVVM مخصوص Kendo UI صورت خواهد گرفت. همچنین کلیه اعمال مرتبط با هر ردیف نیز توسط data binding دو طرفه مدیریت خواهد شد.

Kendo UI MVVM

الگوی MVVM یا Model-View-ViewModel که برای اولین بار جهت کاربردهای WPF و Silverlight معرفی شد، برای ساده سازی اتصال تغییرات کنترل‌های برنامه به خواص ViewModel یک View کاربرد دارد. برای مثال با تغییر عنصر انتخابی یک DropDownList در یک View، بلافاصله خاصیت متصل به آن که در ViewModel برنامه تعریف شده‌است، مقدار دهی و به روز خواهد شد. هدف نهایی آن نیز جدا سازی منطق کدهای UI، از کدهای جاوا اسکریپتی سمت کاربر است. برای این منظور کتابخانه‌هایی مانند Knockout.js به صورت اختصاصی برای این کار تهیه شده‌اند؛ اما Kendo UI نیز جهت یکپارچگی هرچه تمامتر اجزای آن، دارای یک فریم ورک MVVM توکار نیز می‌باشد. طراحی آن نیز بسیار شبیه به Knockout.js است؛ اما با سازگاری 100 درصد با کل مجموعه.

پیاده سازی الگوی MVVM از 4 قسمت تشکیل می‌شود:

- Model که بیانگر خواص متناظر با اشیاء رابط کاربری است.
- View همان رابط کاربری است که به کاربر نمایش داده می‌شود.
- ViewModel واسطی است بین Model و View. کار آن انتقال داده‌ها و رویدادها از View به مدل است و در حالت binding دوطرفه، عکس آن نیز صحیح می‌باشد.
- Declarative data binding جهت رهایی برنامه نویس‌ها از نوشتن کدهای هماهنگ سازی اطلاعات المان‌های View و خواص ViewModel کاربرد دارد.

در ادامه این اجزا را با پیاده سازی مثالی که در ابتدای بحث مطرح شد، دنبال می‌کنیم.

تعریف Model و ViewModel

در سمت سرور، مدل ثبت نام برنامه چنین شکلی را دارد:

```
namespace KendoUI07.Models
{
    public class Registration
    {
        public int Id { set; get; }
        public string UserName { set; get; }
        public string CourseName { set; get; }
        public int Credit { set; get; }
        public string Email { set; get; }
        public string Tel { set; get; }
    }
}
```

در سمت کاربر، این مدل را به نحو ذیل می‌توان تعریف کرد:

```
<script type="text/javascript">
        $(function () {
            var model = kendo.data.Model.define({
                id: "Id",
                fields: {
                    Id: { type: 'number' }, // leave this set to 0 or undefined, so Kendo knows it is
new.
                    UserName: { type: 'string' },
                    CourseName: { type: 'string' },
                    Credit: { type: 'number' },
                    Email: { type: 'string' },
                    Tel: { type: 'string' }
                }
            });
        });
    </script>
```

و ViewModel برنامه در ساده‌ترین شکل آن اکنون چنین تعریفی را خواهد یافت:

```
<script type="text/javascript">
        $(function () {
            var viewModel = kendo.observable({
```

```
                accepted: false,
                course: new model()
            });
        });
    </script>
```

یک viewModel در Kendo UI به صورت یک observable object تعریف می‌شود که می‌تواند دارای تعدادی خاصیت و متد دلخواه باشد. هر خاصیت آن به یک عنصر HTML متصل خواهد شد. در اینجا این اتصال دو طرفه است؛ به این معنا که تغییرات UI به خواص viewModel و برعکس منتقل و منعکس می‌شوند.

اتصال ViewModel به View برنامه

تعریف فرم ثبت نام را در اینجا ملاحظه می‌کنید. فیلدهای مختلف آن بر اساس نکات اعتبارسنجی HTML 5 با ویژگی‌های خاص آن، مزین شده‌اند. جزئیات آن‌را در مطلب « اعتبار سنجی ورودی‌های کاربر در Kendo UI » پیشتر بررسی کرده‌ایم.
اگر به تعریف هر فیلد دقت کنید، ویژگی data-bind جدیدی را هم ملاحظه خواهید کرد:

```
<div id="coursesSection" class="k-rtl k-header">
        <div class="box-col">
            <form id="myForm" data-role="validator" novalidate="novalidate">
                <h3>ثبت نام</h3>
                <ul>
                    <li>
                        <label for="Id">Id</label>
                        <span id="Id" data-bind="text:course.Id"></span>
                    </li>
                    <li>
                        <label for="UserName">نام</label>
                        <input type="text" id="UserName" name="UserName" class="k-textbox"
                               data-bind="value:course.UserName"
                               required />
                    </li>
                    <li>
                        <label for="CourseName">دوره</label>
                        <input type="text" dir="ltr" id="CourseName" name="CourseName" required
                               data-bind="value:course.CourseName" />
                        <span class="k-invalid-msg" data-for="CourseName"></span>
                    </li>
                    <li>
                        <label for="Credit">مبلغ پرداختی</label>
                        <input id="Credit" name="Credit" type="number" min="1000" max="6000"
                               required data-max-msg="6000 و 1000 عددی بین" dir="ltr"
                               data-bind="value:course.Credit"
                               class="k-textbox k-input" />
                        <span class="k-invalid-msg" data-for="Credit"></span>
                    </li>
                    <li>
                        <label for="Email">پست الکترونیک</label>
                        <input type="email" id="Email" dir="ltr" name="Email"
                               data-bind="value:course.Email"
                               required class="k-textbox" />
                    </li>
                    <li>
                        <label for="Tel">تلفن</label>
                        <input type="tel" id="Tel" name="Tel" dir="ltr" pattern="\d{8}"
                               required class="k-textbox"
                               data-bind="value:course.Tel"
                               data-pattern-msg="8 رقم" />
                    </li>
                    <li>
                        <input type="checkbox" name="Accept"
                               data-bind="checked:accepted"
                               required />
                        شرایط دوره را قبول دارم.
                        <span class="k-invalid-msg" data-for="Accept"></span>
                    </li>
                    <li>
                        <button class="k-button"
                                data-bind="enabled: accepted, click: doSave"
                                type="submit">
                            ارسال
                        </button>
                        <button class="k-button" data-bind="click: resetModel">از نو</button>
```

```
                    </li>
                </ul>
                <span id="doneMsg"></span>
            </form>
        </div>
```

برای اتصال ViewModel تعریف شده به ناحیه‌ی مشخص شده با DIV ایی با Id مساوی coursesSection، می‌توان از متد kendo.bind استفاده کرد.

```
<script type="text/javascript">
        $(function () {
            var model = kendo.data.Model.define({
            // ...
            });

            var viewModel = kendo.observable({
            // ...
            });

            kendo.bind($("#coursesSection"), viewModel);
        });
    </script>
```

به این ترتیب Kendo UI به بر اساس تعریف data-bind یک فیلد، برای مثال تغییرات خواص course.UserName را به text box نام کاربر منتقل می‌کند و همچنین اگر کاربر اطلاعاتی را در این text box وارد کند، بلافاصله این تغییرات در خاصیت course.UserName منعکس خواهند شد.

```
<input type="text" id="UserName" name="UserName" class="k-textbox"
       data-bind="value:course.UserName"
       required />
```

بنابراین تا اینجا به صورت خلاصه، مدلی را توسط متد kendo.data.Model.define، معادل مدل سمت سرور خود ایجاد کردیم. سپس وهله‌ای از این مدل را به صورت یک خاصیت جدید دلخواهی در ViewModel تعریف شده توسط متد kendo.observable در معرض دید View برنامه قرار دادیم. در ادامه اتصال ViewModel و View، با فراخوانی متد kendo.bind انجام شد. اکنون برای دریافت تغییرات کنترل‌های برنامه، تنها کافی است ویژگی‌های data-bind ایی را به آن‌ها اضافه کنیم.
در ناحیه‌ی تعریف شده توسط متد kendo.bind، کلیه خواص ViewModel در دسترس هستند. برای مثال اگر به تعریف ViewModel دقت کنید، یک خاصیت دیگر به نام accepted با مقدار false نیز در آن تعریف شده‌است (این خاصیت چون صرفا کاربرد UI داشت، در model برنامه قرار نگرفت). از آن برای اتصال checkbox تعریف شده، به button ارسال اطلاعات، استفاده کرده‌ایم:

```
<input type="checkbox" name="Accept"
       data-bind="checked:accepted"
       required />

<button class="k-button"
        data-bind="enabled: accepted, click: doSave"
        type="submit">
       ارسال
</button>
```

برای مثال اگر کاربر این checkbox را انتخاب کند، مقدار خاصیت accepted، مساوی true خواهد شد. تغییر مقدار این خاصیت، توسط ViewModel بلافاصله در کل ناحیه coursesSection منتشر می‌شود. به همین جهت ویژگی enabled: accepted که به معنای مقید بودن فعال یا غیرفعال بودن دکمه بر اساس مقدار خاصیت accepted است، دکمه را فعال می‌کند، یا برعکس و برای انجام این عملیات نیازی نیست کدنویسی خاصی را انجام داد. در اینجا بین checkbox و button یک سیم کشی برقرار است.

ارسال داده‌های تغییر کرده‌ی ViewModel به سرور

تا اینجا 4 جزء اصلی الگوی MVVM که در ابتدای بحث عنوان شد، تکمیل شده‌اند. مدل اطلاعات فرم تعریف گردید. ViewModel ایی

که این خواص را به المان‌های فرم متصل می‌کند نیز در ادامه اضافه شده‌است. توسط ویژگی‌های data-bind کار Declarative data binding انجام می‌شود.

در ادامه نیاز است تغییرات ViewModel را به سرور، جهت ثبت، به روز رسانی و حذف نهایی منتقل کرد.

```
<script type="text/javascript">
        $(function () {
            var model = kendo.data.Model.define({
                //...
            });

            var dataSource = new kendo.data.DataSource({
                type: 'json',
                transport: {
                    read: {
                        url: "api/registrations",
                        dataType: "json",
                        contentType: 'application/json; charset=utf-8',
                        type: 'GET'
                    },
                    create: {
                        url: "api/registrations",
                        contentType: 'application/json; charset=utf-8',
                        type: "POST"
                    },
                    update: {
                        url: function (course) {
                            return "api/registrations/" + course.Id;
                        },
                        contentType: 'application/json; charset=utf-8',
                        type: "PUT"
                    },
                    destroy: {
                        url: function (course) {
                            return "api/registrations/" + course.Id;
                        },
                        contentType: 'application/json; charset=utf-8',
                        type: "DELETE"
                    },
                    parameterMap: function (data, type) {
                        // Convert to a JSON string.  Without this step your content will be form
encoded.
                        return JSON.stringify(data);
                    }
                },
                schema: {
                    model: model
                },
                error: function (e) {
                    alert(e.errorThrown);
                },
                change: function (e) {
                    // فراخوانی در زمان دریافت اطلاعات از سرور و یا تغییرات محلی
                    viewModel.set("coursesDataSourceRows", new
kendo.data.ObservableArray(this.view()));
                }
            });

            var viewModel = kendo.observable({
                //...
            });

            kendo.bind($("#coursesSection"), viewModel);
            dataSource.read(); // دریافت لیست موجود از سرور در آغاز کار
        });
    </script>
```

در اینجا تعریف DataSource کار با منبع داده راه دور ASP.NET Web API را مشاهده می‌کنید. تعاریف اصلی آن با تعاریف مطرح شده در مطلب « فعال سازی عملیات CRUD در Kendo UI Grid » یکی هستند. هر قسمت آن مانند read، create، update و destory به یکی از متدهای کنترلر ASP.NET Web API اشاره می‌کنند. حالت‌های update و destroy بر اساس Id ردیف انتخابی کار می‌کنند. این Id را باید در قسمت model مربوط به اسکیمای تعریف شده، دقیقا مشخص کرد. عدم تعریف فیلد id، سبب خواهد شد تا عملیات update نیز در حالت create تفسیر شود.

متصل کردن DataSource به ViewModel

تا اینجا DataSource ایی جهت کار با سرور تعریف شده‌است؛ اما مشخص نیست که اگر رکوردی اضافه شد، چگونه باید اطلاعات خودش را به روز کند. برای این منظور خواهیم داشت:

```
<script type="text/javascript">
        $(function () {
            $("#coursesSection").kendoValidator({
                // ...
            });

            var model = kendo.data.Model.define({
                // ...
            });

            var dataSource = new kendo.data.DataSource({
                // ...
            });

            var viewModel = kendo.observable({
                accepted: false,
                course: new model(),
                doSave: function (e) {
                    e.preventDefault();
                    console.log("this", this.course);
                    var validator = $("#coursesSection").data("kendoValidator");
                    if (validator.validate()) {
                        if (this.course.Id == 0) {
                            dataSource.add(this.course);
                        }
                        dataSource.sync(); // push to the server
                        this.set("course", new model()); // reset controls
                    }
                },
                resetModel: function (e) {
                    e.preventDefault();
                    this.set("course", new model());
                }
             });

            kendo.bind($("#coursesSection"), viewModel);
            dataSource.read(); // دریافت لیست موجود از سرور در آغاز کار
        });
    </script>
```

همانطور که در تعاریف تکمیلی viewModel مشاهده می‌کنید، اینبار دو متد جدید دلخواه doSave و resetModel را اضافه کرده‌ایم. در متد doSave، ابتدا بررسی می‌کنیم آیا اعتبارسنجی فرم با موفقیت انجام شده‌است یا خیر. اگر بله، توسط متد add منبع داده، اطلاعات فرم جاری را توسط شیء course که هم اکنون به تمامی فیلدهای آن متصل است، اضافه می‌کنیم. در اینجا بررسی شده‌است که آیا Id این اطلاعات صفر است یا خیر. از آنجائیکه از همین متد برای به روز رسانی نیز در ادامه استفاده خواهد شد، در حالت به روز رسانی، Id شیء ثبت شده، از طرف سرور دریافت می‌گردد. بنابراین غیر صفر بودن این Id به معنای عملیات به روز رسانی است و در این حالت نیازی نیست کار بیشتری را انجام داد؛ زیرا شیء متناظر با آن پیشتر به منبع داده اضافه شده‌است.

استفاده از متد add صرفا به معنای مطلع کردن منبع داده محلی از وجود رکوردی جدید است. برای ارسال این تغییرات به سرور، از متد sync آن می‌توان استفاده کرد. متد sync بر اساس متد add یک درخواست POST، بر اساس شیءایی که Id غیر صفر دارد، یک درخواست PUT و با فراخوانی متد remove بر روی منبع داده، یک درخواست DELETE را به سمت سرور ارسال می‌کند.

متد دلخواه resetModel سبب مقدار دهی مجدد شیء course با یک وهله‌ی جدید از شیء model می‌شود. همینقدر برای پاک کردن تمامی کنترل‌های صفحه کافی است.

تا اینجا دو متد جدید را در ViewModel برنامه تعریف کرده‌ایم. در مورد نحوه‌ی اتصال آن‌ها به View، به کدهای دو دکمه‌ی موجود در فرم دقت کنید:

```
<button class="k-button"
        data-bind="enabled: accepted, click: doSave"
        type="submit">
```

```
        ارسال
</button>
<button class="k-button" data-bind="click: resetModel">از نو</button>
```

این متدها نیز توسط ویژگی‌های data-bind به هر دکمه نسبت داده شده‌اند. به این ترتیب برای مثال با کلیک کاربر بر روی دکمه‌ی submit، متد doSave موجود در ViewModel فراخوانی می‌شود.

مدیریت سمت سرور ثبت، ویرایش و حذف اطلاعات

در حالت ثبت، متد Post توسط آدرس مشخص شده در قسمت create منبع داده، فراخوانی می‌گردد. نکته‌ی مهمی که در اینجا باید به آن دقت داشت، نحوه‌ی بازگشت Id رکورد جدید ثبت شده‌است. اگر این تنظیم صورت نگیرد، Id رکورد جدید را در لیست، مساوی صفر مشاهده خواهید کرد و منبع داده این رکورد را همواره به عنوان یک رکورد جدید، مجددا به سرور ارسال می‌کند.

```
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using KendoUI07.Models;

namespace KendoUI07.Controllers
{
    public class RegistrationsController : ApiController
    {
        public HttpResponseMessage Delete(int id)
        {
            var item = RegistrationsDataSource.LatestRegistrations.FirstOrDefault(x => x.Id == id);
            if (item == null)
                return Request.CreateResponse(HttpStatusCode.NotFound);

            RegistrationsDataSource.LatestRegistrations.Remove(item);
            return Request.CreateResponse(HttpStatusCode.OK, item);
        }

        public IEnumerable<Registration> Get()
        {
            return RegistrationsDataSource.LatestRegistrations;
        }

        public HttpResponseMessage Post(Registration registration)
        {
            if (!ModelState.IsValid)
                return Request.CreateResponse(HttpStatusCode.BadRequest);

            var id = 1;
            var lastItem = RegistrationsDataSource.LatestRegistrations.LastOrDefault();
            if (lastItem != null)
            {
                id = lastItem.Id + 1;
            }
            registration.Id = id;
            RegistrationsDataSource.LatestRegistrations.Add(registration);

            // ارسال آی دی مهم است تا از ارسال رکوردهای تکراری جلوگیری شود
            return Request.CreateResponse(HttpStatusCode.Created, registration);
        }

        [HttpPut] // Add it to fix this error: The requested resource does not support http method
'PUT'
        public HttpResponseMessage Update(int id, Registration registration)
        {
            var item = RegistrationsDataSource.LatestRegistrations
                                        .Select(
                                            (prod, index) =>
                                                new
                                                {
                                                    Item = prod,
                                                    Index = index
                                                })
                                        .FirstOrDefault(x => x.Item.Id == id);
            if (item == null)
```

```
                return Request.CreateResponse(HttpStatusCode.NotFound);


            if (!ModelState.IsValid || id != registration.Id)
                return Request.CreateResponse(HttpStatusCode.BadRequest);

            RegistrationsDataSource.LatestRegistrations[item.Index] = registration;
            return Request.CreateResponse(HttpStatusCode.OK);
        }
    }
}
```

در اینجا بیشتر امضای این متدها مهم هستند، تا منطق پیاده سازی شده در آن‌ها. همچنین بازگشت Id رکورد جدید، توسط متد Post نیز بسیار مهم است و سبب می‌شود تا DataSource بداند با فراخوانی متد sync آن، باید عملیات Post یا create انجام شود یا Put و update.

نمایش آنی اطلاعات ثبت شده در یک لیست

ردیف‌های اضافه شده به منبع داده را می‌توان بلافاصله در همان سمت کلاینت توسط Kendo UI Template که قابلیت کار با ViewModelها را دارد، نمایش داد:

```
<div id="coursesSection" class="k-rtl k-header">
        <div class="box-col">
            <form id="myForm" data-role="validator" novalidate="novalidate">
                        <!--فرم بحث شده در ابتدای مطلب-->
            </form>
        </div>
        <div id="results">
            <table class="metrotable">
                <thead>
                    <tr>
                        <th>Id</th>
                        <th>نام</th>
                        <th>دوره</th>
                        <th>هزینه</th>
                        <th>ایمیل</th>
                        <th>تلفن</th>
                        <th></th>
                        <th></th>
                    </tr>
                </thead>
                <tbody data-template="row-template" data-bind="source: coursesDataSourceRows"></tbody>
                <tfoot data-template="footer-template" data-bind="source: this"></tfoot>
            </table>
            <script id="row-template" type="text/x-kendo-template">
                <tr>
                    <td data-bind="text: Id"></td>
                    <td data-bind="text: UserName"></td>
                    <td dir="ltr" data-bind="text: CourseName"></td>
                    <td>
                        #: kendo.toString(get("Credit"), "c0") #
                    </td>
                    <td data-bind="text: Email"></td>
                    <td data-bind="text: Tel"></td>
                    <td><button class="k-button" data-bind="click: deleteCourse">حذف</button></td>
                    <td><button class="k-button" data-bind="click: editCourse">ویرایش</button></td>
                </tr>
            </script>
            <script id="footer-template" type="text/x-kendo-template">
                <tr>
                    <td colspan="3"></td>
                    <td>
                        جمع کل: #: kendo.toString(totalPrice(), "c0") #
                    </td>
                    <td colspan="2"></td>
                    <td></td>
                    <td></td>
                </tr>
            </script>
        </div>
    </div>
```

در ناحیه‌ی coursesSection که توسط متد kendo.bind به viewModel برنامه متصل شده‌است، یک جدول را برای نمایش ردیف‌های ثبت شده توسط کاربر اضافه کرده‌ایم. thead آن بیانگر سر ستون جدول است. قسمت tbody و tfoot این جدول توسط دو Kendo UI Template مقدار دهی شده‌اند. هر کدام نیز منبع داده‌اشان را از view model دریافت می‌کنند. در row-template معادل خواص شیء course را مشاهده می‌کنید. در footer-template متد totalPrice برای نمایش جمع ستون هزینه اضافه شده‌است. بنابراین مطابق این قسمت از View، به یک خاصیت جدید coursesDataSourceRows و سه متد deleteCourse، editCourse و totalPrice نیاز است:

```
<script type="text/javascript">
        $(function () {
            // ...
            var viewModel = kendo.observable({
                accepted: false,
                course: new model(),
                coursesDataSourceRows: new kendo.data.ObservableArray([]),
                doSave: function (e) {
                       // ...
                },
                resetModel: function (e) {
                      // ...
                },
                totalPrice: function () {
                    var sum = 0;
                    $.each(this.get("coursesDataSourceRows"), function (index, item) {
                        sum += item.Credit;
                    });
                    return sum;
                },
                deleteCourse: function (e) {
                    // the current data item is passed as the "data" field of the event argument
                    var course = e.data;
                    dataSource.remove(course);
                    dataSource.sync(); // push to the server
                },
                editCourse: function(e) {
                    // the current data item is passed as the "data" field of the event argument
                    var course = e.data;
                    this.set("course", course);
                }
            });

            kendo.bind($("#coursesSection"), viewModel);
            dataSource.read(); // دریافت لیست موجود از سرور در آغاز کار
        });
    </script>
```

نحوه‌ی اتصال خاصیت جدید coursesDataSourceRows که به عنوان منبع داده ردیف‌های row-template عمل می‌کند، به این صورت است:

- ابتدا خاصیت دلخواه coursesDataSourceRows به viewModel اضافه می‌شود تا در ناحیه‌ی coursesSection در دسترس قرار گیرد.

- سپس اگر به انتهای تعریف DataSource دقت کنید، داریم:

```
<script type="text/javascript">
        $(function () {
            var dataSource = new kendo.data.DataSource({
                //...
                change: function (e) {
                    // فراخوانی در زمان دریافت اطلاعات از سرور و یا تغییرات محلی
                    viewModel.set("coursesDataSourceRows", new
kendo.data.ObservableArray(this.view()));
                }
            });
        });
    </script>
```

متد change آن، هر زمانیکه اطلاعاتی در منبع داده تغییر کنند یا اطلاعاتی به سمت سرور ارسال یا دریافت گردد، فراخوانی می‌شود. در همینجا فرصت خواهیم داشت تا خاصیت coursesDataSourceRows را جهت نمایش اطلاعات موجود در منبع داده،

مقدار دهی کنیم. همین مقدار دهی ساده سبب اجرای row-template برای تولید ردیف‌های جدول می‌شود. استفاده از new kendo.data.ObservableArray سبب خواهد شد تا اگر اطلاعاتی در فرم برنامه تغییر کند، این اطلاعات بلافاصله در لیست گزارش برنامه نیز منعکس گردد.

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید:

KendoUI07.zip

عنوان: **یکپارچه سازی سیستم اعتبارسنجی ASP.NET MVC با Kendo UI validator**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۷ ۱۱:۵۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, MVC, jQuery, Kendo UI

روش پیش فرض اعتبارسنجی برنامه‌های ASP.NET MVC، استفاده از دو افزونه‌ی jquery.validate و jquery.validate.unobtrusive است.

```
<script src="~/Scripts/jquery.validate.min.js" type="text/javascript"></script>
    <script src="~/Scripts/jquery.validate.unobtrusive.min.js" type="text/javascript"></script>
```

کار اصلی اعتبارسنجی، توسط افزونه‌ی jquery.validate انجام می‌شود و فایل jquery.validate.unobtrusive صرفا یک وفق دهنده و مترجم ویژگی‌های خاص ASP.NET MVC به jquery.validate است.

عدم سازگاری پیش فرض jquery.validate با بعضی از ویجت‌های Kendo UI

در حالت استفاده از Kendo UI، این سیستم هنوز هم کار می‌کند؛ اما با یک مشکل. اگر برای مثال از kendoComboBox استفاده کنید، اعتبارسنجی‌های تعریف شده در برنامه، توسط jquery.validate دیده نخواهند شد. برای مثال فرض کنید یک چنین مدلی در اختیار View برنامه قرار گرفته است:

```
public class OrderDetailViewModel
    {
        [StringLength(15)]
        [Required]
        public string Destination { get; set; }
    }
```

با این View که در آن به فیلد Destination، یک kendoComboBox متصل شده‌است:

```
@model Mvc4TestViewModel.Models.OrderDetailViewModel

@using (Ajax.BeginForm(actionName: "Index", controllerName: "Home",
                       ajaxOptions:new AjaxOptions(),
                       htmlAttributes: new { id = "Form1", name ="Form1" }, routeValues: new { }
                       ))
{
    @Html.AntiForgeryToken()
    @Html.ValidationSummary(true)

    <fieldset>
        <legend>OrderDetail</legend>
        <div class="editor-label">
            @Html.LabelFor(model => model.Destination)
        </div>
        <div class="editor-field">
            @Html.TextBoxFor(model => model.Destination, new { @class = "k-textbox" })
            @Html.ValidationMessageFor(model => model.Destination)
        </div>

        <p>
            <button class="k-button" type="submit" title="Sumbit">
                Sumbit
            </button>
        </p>
    </fieldset>
}

@section JavaScript
{
    <script type="text/javascript">
        $(function () {
            $("#Destination").kendoComboBox({
                dataSource: [
                    "loc 1",
                    "loc 2"
                ]
```

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

اگر برنامه را اجرا کنید و بر روی دکمه‌ی submit کلیک نمائید، ویژگی Required عمل نخواهد کرد و عملا در سمت کاربر اعتبارسنجی رخ نمی‌دهد.

```
◢ <div class="editor-field">
  ◢ <span class="k-widget k-combobox k-header k-textbox" style="">
    ◢ <span tabindex="-1" class="k-dropdown-wrap k-state-default" unselectable="on">
        <input name="Destination_input" tabindex="0" class="k-input k-textbox" role="combobox" aria-
        busy="false" aria-disabled="false" aria-expanded="false" aria-readonly="false" aria-
        activedescendant="Destination_option_selected" aria-owns="Destination_listbox" style="width: 100%;"
        aria-autocomplete="list" type="text" autocomplete="off"></input>
      ▷ <span tabindex="-1" class="k-select" unselectable="on">…</span>
      </span>
      <input name="Destination" class="k-textbox input-validation-error" id="Destination" aria-
      disabled="false" aria-invalid="true" aria-readonly="false" aria-required="true" aria-
      describedby="Destination-error" style="display: none;" type="text" data-val-required="The Destination
      field is required." data-val="true" data-val-length-max="15" data-val-length="The field Destination must
      be a string with a maximum length of 15." data-role="combobox" value=""></input>
    </span>
  ▷ <span class="field-validation-error" data-valmsg-replace="true" data-valmsg-for="Destination">…</span>
  </div>
```

همانطور که در تصویر مشاهده می‌کنید، با اتصال kendoComboBox به یک فیلد، این فیلد در حالت مخفی قرار می‌گیرد و ویجت کندو یو آی بجای آن نمایش داده خواهد شد. در این حالت چون در فایل jquery.validate.js چنین تنظیمی وجود دارد:

```
$.extend( $.validator, {
    defaults: {
     //…
     ignore: ":hidden",
```

به صورت پیش فرض از اعتبارسنجی فیلدهای مخفی صرفنظر می‌شود.
راه حل آن نیز ساده‌است. تنها باید خاصیت ignore را بازنویسی کرد و تغییر داد:

```
<script type="text/javascript">
        $(function () {
            var form = $('#Form1');
            form.data('validator').settings.ignore = ''; // default is ":hidden".
        });
    </script>
```

در اینجا صرفا خاصیت ignore فرم یک، جهت درنظر گرفتن فیلدهای مخفی تغییر کرده‌است. اگر می‌خواهید این تنظیم را به تمام فرم‌ها اعمال کنید، می‌توان از دستور ذیل استفاده کرد:

```
<script type="text/javascript">
    $.validator.setDefaults({
        ignore: ""
    });
</script>
```

یکپارچه کردن سیستم اعتبارسنجی Kendo UI با سیستم اعتبارسنجی ASP.NET MVC

در مطلب « اعتبار سنجی ورودی‌های کاربر در Kendo UI » با زیرساخت اعتبارسنجی Kendo UI آشنا شدید. برای اینکه بتوان این سیستم را با ASP.NET MVC یکپارچه کرد، نیاز است دو کار صورت گیرد:

الف) تعریف فایل kendo.aspnetmvc.js به صفحه اضافه شود:

```
<script src="~/Scripts/kendo.aspnetmvc.js" type="text/javascript"></script>
```

ب) همانند قبل، متد kendoValidator بر روی فرم فراخوانی شود تا سیستم اعتبارسنجی Kendo UI در این ناحیه فعال گردد:

```
<script type="text/javascript">
        $(function () {
            $("form").kendoValidator();
        });
    </script>
```

پس از آن خواهیم داشت:

فایل kendo.aspnetmvc.js که در بسته‌ی مخصوص Kendo UI تهیه شده برای ASP.NET MVC موجود است (در پوشه‌ی js آن)، عملکردی مشابه فایل jquery.validate.unobtrusive مایکروسافت دارد. کار آن وفق دادن و ترجمه‌ی اعتبارسنجی unobtrusive به روش Kendo UI است.

این فایل را از اینجا می‌توانید دریافت کنید:

kendo.mvc.zip

البته باید دقت داشت که در حال حاضر فقط ویژگی‌های ذیل از ASP.NET MVC توسط kendo.aspnetmvc.js پشتیبانی می‌شوند :

```
Required
```

```
StringLength
Range
RegularExpression
```

برای تکمیل آن می‌توان از یک پروژه‌ی سورس باز به نام Moon.Validation for KendoUI Validator استفاده کرد. برای مثال remote validation مخصوص Kendo UI را اضافه کرده‌است .

نظرات خوانندگان

نویسنده: امیر
تاریخ: ۱۳۹۳/۰۸/۲۸ ۱۳:۰

سئوالی برای من پیش اومد امکان استفاده از مدل mvc داخل کنترل‌های جاوااسکریپت کندو هست ؟ مثلا اگه combobox انتخاب بشه آیدیش در مدل اصلی انتخاب بشه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۸ ۱۳:۴۶

« استفاده ازExpressionها جهت ایجاد Strongly typed view در ASP.NET MVC »

نویسنده: شاکری
تاریخ: ۱۳۹۳/۱۰/۰۲ ۱۰:۳۷

من برای فعال سازی remote validation از پروژه Moon استفاده کردم اما باز هم عمل نمیکنه!

```
[Moon.Web.Validation.RemoteValidator("IsUserExist", "Admin", HttpMethod = "POST", ErrorMessage = "نام کاربری قبلا ثبت شده است.")]
        public string Username { get; set; }
```

```
@Html.EditorFor(model => model.Username)
  @Html.ValidationMessageFor(model => model.Username)
```

```
[HttpPost]
        public ActionResult IsUserExist(string Username)
        {
            if (userSerive.isUser(Username)) return Json(true);
            return Json(false);
        }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۳ ۱۵:۲۸

کتابخانه‌ی ذکر شده را حذف و سپس به روش زیر برای فعال سازی remote validation عمل کنید:

```
$.validator.methods.remote = function () { /* disabled */ };
            $("form").kendoValidator({
                onfocusout: true,
                onkeyup: true,
                rules: {
                    remote: function (input) {
                        var remoteAttr = input.attr("data-val-remote-url");
                        if (typeof remoteAttr === typeof undefined || remoteAttr === false) {
                            return true;
                        }

                        var isInvalid = true;
                        var data = {};
                        data[input.attr('name')] = input.val();

                        $.ajax({
                            url: remoteAttr,
                            mode: "abort",
                            port: "validate" + input.attr('name'),
                            dataType: "json",
                            type: input.attr("data-val-remote-type"),
                            data: data,
                            async: false,
                            success: function (response) {
                                isInvalid = response;
                            }
```

```
                    });
                    return !isInvalid;
                }
            },
            messages: {
                remote: function (input) {
                    return input.data('val-remote');
                }
            }
        });
```

- در اینجا در ابتدا متد remote کتابخانه‌ی jQuery Validator غیرفعال می‌شود. سپس یک rule جدید، به kendoValidator به نام دلخواه remote اضافه شده‌است. چون rule‌های kendoValidator اعمال async را پشتیبانی نمی‌کنند، در درخواست ajax آن async: false تنظیم شده‌است. به این ترتیب سطر پس از ajax، پس از پایان کار عملیات ajax فراخوانی می‌شود و در این حالت kendoValidator بدون مشکل کار خواهد کرد.
- سمت سرور آن هم مانند قبل به همراه استفاده از ویژگی Remote است که از آن صرفا برای مقدار دهی data-val-remote-url و val-remote که در rule جدید استفاده می‌شوند، کمک گرفته خواهد شد.

نویسنده: شاکری
تاریخ: ۱۳۹۳/۱۰/۰۴ ۱۱:۲۸

خیلی ممنون بابت پاسخگویی
این روش در حالتی جواب میده که ما اطلاعات رو از طریق فرم ارسال کنیم.
اما من برای درج یا ویرایش از template در گرید کندو به صورت popup استفاده کردم که در این حالت از فرم برای ارسال اطلاعات استفاده نمی‌کند!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۴ ۱۱:۳۷

مراجعه کنید به مطلب « اعتبار سنجی ورودی‌های کاربر در Kendo UI ». در انتهای مطلب در مورد «اعتبارسنجی سفارشی در DataSource» بحث شده‌است. Kendo UI Grid هم اطلاعات اعتبارسنجی فیلدهای خودش را از data source دریافت می‌کند و ... اصول طراحی اعتبارسنجی آن، هیچ تفاوتی با نکته‌ی عنوان شده ندارد (یک custom rule سفارشی را به نام remote، دقیقا مانند همین مثال می‌توانید اضافه کنید).

نویسنده: شاکری
تاریخ: ۱۳۹۳/۱۰/۰۴ ۱۲:۵۴

باز هم ممنون بابت پاسخگویی
در حالتی که از popup template داخل گرید استفاده کنیم برای remote validation میشه از مثال شما با اندکی تغییرات استفاده کرد.

```
(function($, kendo) {
        $.extend(true, kendo.ui.validator, {
            rules: {
                remote: function(input) {
                    var remoteAttr = input.attr("data-val-remote-url");
                    if (typeof remoteAttr === typeof undefined || remoteAttr === false) {
                        return true;
                    }

                    var isInvalid = true;
                    var data = {};
                    data[input.attr('name')] = input.val();

                    $.ajax({
                        url: remoteAttr,
                        mode: "abort",
                        port: "validate" + input.attr('name'),
                        dataType: "json",
                        type: input.attr("data-val-remote-type"),
```

```
                        data: data,
                        async: false,
                        success: function(response) {
                            isInvalid = response;
                        }
                    });
                    return !isInvalid;
                }
            },
            messages: {
                remote: function(input) {
                    return input.data('val-remote');
                }
            }
        });
    })(jQuery, kendo);
```

عنوان: بررسی ویجت Kendo UI File Upload
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۰۳ ۱۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Kendo UI

Kendo UI به همراه یک ویجت وب مخصوص ارسال فایل‌ها به سرور نیز هست. این ویجت قابلیت ارسال چندین فایل با هم را به صورت Ajax ایی دارا است و همچنین کاربران می‌توانند فایل‌ها را با کشیدن و رها کردن بر روی آن، به لیست فایل‌های قابل ارسال اضافه کنند.
ارسال فایل Ajax ایی آن توسط HTML5 File API صورت می‌گیرد که در تمام مرورگرهای جدید پشتیبانی خوبی از آن وجود دارد. در مرورگرهای قدیمی‌تر، به صورت خودکار همان حالت متداول ارسال همزمان فایل‌ها را فعال می‌کند (یا همان post back معمولی).

فعال سازی مقدماتی kendoUpload

ابتدایی‌ترین حالت کار با kendoUpload، فعال سازی حالت post back معمولی است؛ به شرح زیر:

```
<form method="post" action="submit" enctype="multipart/form-data">
  <div>
    <input name="files" id="files" type="file" />
    <input type="submit" value="Submit" class="k-button" />
  </div>
</form>
<script>
  $(document).ready(function() {
     $("#files").kendoUpload();
  });
</script>
```

در این حالت صرفا input با نوع file، با ظاهری سازگار با سایر کنترل‌های Kendo UI به نظر می‌رسد و عملیات ارسال فایل، همانند قبل به همراه یک post back است. این روش برای حالتی مفید است که بخواهید یک فایل را به همراه سایر عناصر فرم در طی یک مرحله به سمت سرور ارسال کنید.

فعال سازی حالت ارسال فایل Ajax ایی kendoUpload

برای فعال سازی ارسال Ajax ایی فایل‌ها در Kendo UI نیاز است خاصیت async آن‌را به نحو ذیل مقدار دهی کرد:

```
<script type="text/javascript">
        $(function () {
            $("#files").kendoUpload({
                name: "files",
                async: { // async configuration
                    saveUrl: "@Url.Action("Save", "Home")", // the url to save a file is '/save'
                    removeUrl: "@Url.Action("Remove", "Home")", // the url to remove a file is
'/remove'
                    autoUpload: false, // automatically upload files once selected
                    removeVerb: 'POST'
                },
                multiple: true,
                showFileList: true
            });
        });
    </script>
```

در اینجا دو آدرس ذخیره سازی فایل‌ها و همچنین حذف آن‌ها را مشاهده می‌کنید. امضای این دو اکشن متد در ASP.NET MVC به صورت ذیل هستند:

```
[HttpPost]
        public ActionResult Save(IEnumerable<HttpPostedFileBase> files)
        {
```

```
            if (files != null)
            {
                // ...
                // Process the files and save them
                // ...
            }

            // Return an empty string to signify success
            return Content("");
        }

        [HttpPost]
        public ContentResult Remove(string[] fileNames)
        {
            if (fileNames != null)
            {
                foreach (var fullName in fileNames)
                {
                    // ...
                    // delete the files
                    // ...
                }
            }

            // Return an empty string to signify success
            return Content("");
        }
```

در هر دو حالت، لیستی از فایل‌ها توسط kendoUpload به سمت سرور ارسال می‌شوند. در حالت Save، محتوای این فایل‌ها جهت ذخیره سازی بر روی سرور در دسترس خواهد بود. در حالت Remove، صرفا نام این فایل‌ها برای حذف از سرور، توسط کاربر ارسال می‌شوند.

دو دکمه‌ی حذف با کارکردهای متفاوت در ویجت kendoUpload وجود دارند. در ابتدای کار، پیش از ارسال فایل‌ها به سرور:

کلیک بر روی دکمه‌ی حذف در این حالت، صرفا فایلی را از لیست سمت کاربر حذف می‌کند.

پس از ارسال فایل‌ها به سرور:

اما پس از پایان عملیات ارسال، اگر کاربر بر روی دکمه‌ی حذف کلیک کند، توسط آدرس مشخص شده توسط خاصیت removeUrl، نام فایل‌های مورد نظر، برای حذف از سرور ارسال می‌شوند.

چند نکته‌ی تکمیلی

- تنظیم خاصیت autoUpload به true سبب می‌شود تا پس از انتخاب فایل‌ها توسط کاربر، بلافاصله و به صورت خودکار عملیات ارسال فایل‌ها به سرور آغاز شوند. اگر به false تنظیم شود، دکمه‌ی ارسال فایل‌ها در پایین لیست نمایش داده خواهد شد.
- شاید علاقمند باشید تا removeVerb را به DELETE تغییر دهید؛ بجای POST. به همین منظور می‌توان خاصیت removeVerb در اینجا مقدار دهی کرد.
- با تنظیم خاصیت multiple به true، کاربر قادر خواهد شد تا توسط صفحه‌ی دیالوگ انتخاب فایل‌ها، قابلیت انتخاب بیش از یک فایل را داشته باشد.
- showFileList نمایش لیست فایل‌ها را سبب می‌شود.

تعیین پسوند فایل‌های صفحه‌ی انتخاب فایل‌ها

هنگامیکه کاربر بر روی دکمه‌ی انتخاب فایل‌ها برای ارسال کلیک می‌کند، در صفحه‌ی دیالوگ باز شده می‌توان پسوندهای پیش فرض مجاز را نیز تعیین کرد.

برای این منظور تنها کافی است ویژگی accept را به input از نوع فایل اضافه کرد. چند مثال در این مورد:

```
<!-- Content Type with wildcard.  All Images -->
<input type="file" id="demoFile" title="Select file" accept="image/*" />

<!-- List of file extensions -->
<input type="file" id="demoFile" title="Select file" accept=".jpg,.png,.gif" />

<!-- Any combination of the above -->
<input type="file" id="demoFile" title="Select file" accept="audio/*,application/pdf,.png" />
```

نمایش متن کشیدن و رها کردن، بومی سازی برچسب‌ها و نمایش راست به چپ

همانطور که در تصاویر فوق ملاحظه می‌کنید، نمایش این ویجت راست به چپ و پیام‌های آن نیز ترجمه شده‌اند.

برای راست به چپ سازی آن مانند قبل تنها کافی است input مرتبط، در یک div با کلاس k-rtl محصور شود:

```
<div class="k-rtl k-header">
            <input name="files" id="files" type="file"  />
        </div>
```

برای بومی سازی پیام‌های آن می‌توان مانند مثال ذیل، خاصیت localization را مقدار دهی کرد:

```
<script type="text/javascript">
        $(function () {
            $("#files").kendoUpload({
                name: "files",
                async: {
                 //...
                },
                //...
                localization: {
                    select: 'انتخاب فایل‌ها برای ارسال',
                    remove: 'حذف فایل',
                    retry: 'سعی مجدد',
                    headerStatusUploading: 'در حال ارسال فایل‌ها',
                    headerStatusUploaded: 'پایان ارسال',
                    cancel: "لغو",
                    uploadSelectedFiles: "ارسال فایل‌ها",
                    dropFilesHere: "فایل‌ها را برای ارسال، کشیده و در اینجا رها کنید",
                    statusUploading: "در حال ارسال",
                    statusUploaded: "ارسال شد",
                    statusWarning: "اخطار",
                    statusFailed: "خطا در ارسال"
                }
            });
        });
    </script>
```

به علاوه متن dropFilesHere به صورت پیش فرض نامرئی است. برای نمایش آن نیاز است CSS موجود را بازنویسی کرد تا em مرتبط مرئی شود:

```
<style type="text/css">
div.k-dropzone {
    border: 1px solid #c5c5c5; /* For Default; Different for each theme */
}

div.k-dropzone em {
    visibility: visible;
}
</style>
```

تغییر قالب نمایش لیست فایل‌ها

لیست فایل‌ها در ویجت kendoUpload دارای یک قالب پیش فرض است که امکان بازنویسی کامل آن وجود دارد. ابتدا نیاز است یک kendo-template را بر این منظور تدارک دید:

```
<script id="fileListTemplate" type="text/x-kendo-template">
        <li class='k-file'>
            <span class='k-progress'></span>
            <span class='k-icon'></span>
            <span class='k-filename' title='#=name#'>#=name# (#=size# bytes)</span>
            <strong class='k-upload-status'></strong>
        </li>
    </script>
```

و سپس برای استفاده از آن خواهیم داشت:

```
<script type="text/javascript">
        $(function () {
            $("#files").kendoUpload({
```

```
                name: "files",
                async: {
                // ...
                },
                // ...
                template: kendo.template($('#fileListTemplate').html()),
                // ...
            });
        });
    </script>
```

در این قالب، مقدار size هر فایل نیز در کنار نام آن نمایش داده می‌شود.

رخدادهای ارسال فایل‌ها

افزونه‌ی kendoUpload در حالت ارسال Ajax ایی فایل‌ها، رخدادهایی مانند شروع به ارسال، موفقیت، پایان، درصد ارسال فایل‌ها و امثال آن‌را نیز به همراه دارد که لیست کامل آن‌ها را در ذیل مشاهده می‌کنید:

```
<script type="text/javascript">
        $(function () {
            $("#files").kendoUpload({
                name: "files",
                async: { // async configuration
                //...
                },
                //...
                localization: {
                },
                cancel: function () {
                    console.log('Cancel Event.');
                },
                complete: function () {
                    console.log('Complete Event.');
                },
                error: function () {
                    console.log('Error uploading file.');
                },
                progress: function (e) {
                    console.log('Uploading file ' + e.percentComplete);
                },
                remove: function () {
                    console.log('File removed.');
                },
                select: function () {
                    console.log('File selected.');
                },
                success: function () {
                    console.log('Upload successful.');
                },
                upload: function (e) {
                    console.log('Upload started.');
                }
            });
        });
    </script>
```

ارسال متادیتای اضافی به همراه فایل‌های ارسالی

فرض کنید می‌خواهید به همراه فایل‌های ارسالی به سرور، پارامتر codeId را نیز ارسال کنید. برای این منظور باید خاصیت e.data رویداد upload را به نحو ذیل مقدار دهی کرد:

```
<script type="text/javascript">
        $(function () {
            $("#files").kendoUpload({
                name: "files",
                async: {
                //...
```

```
                },
                //...
                localization: {
                },
                upload: function (e) {
                    console.log('Upload started.');
                    // Sending metadata to the save action
                    e.data = {
                        codeId: "1234567",
                        param2: 12
                        //, ...
                    };
                }
            });
        });
    </script>
```

سپس در سمت سرور، امضای متد Save بر اساس پارامترهای تعریف شده در سمت کاربر، به نحو ذیل تغییر می‌کند:

```
[HttpPost]
  public ActionResult Save(IEnumerable<HttpPostedFileBase> files, string codeId)
```

فعال سازی ارسال batch

اگر در متد Save سمت سرور یک break point قرار دهید، مشاهده خواهید کرد که به ازای هر فایل موجود در لیست در سمت کاربر، یکبار متد Save فراخوانی می‌شود و عملا متد Save، لیستی از فایل‌ها را در طی یک فراخوانی دریافت نمی‌کند. برای فعال سازی این قابلیت تنها کافی است خاصیت batch را به true تنظیم کنیم:

```
<script type="text/javascript">
        $(function () {
            $("#files").kendoUpload({
                name: "files",
                async: {
                    // ....
                    batch: true
                },
            });
        });
    </script>
```

به این ترتیب دیگر لیست فایل‌ها به صورت مجزا در سمت کاربر نمایش داده نمی‌شود و تمام آن‌ها با یک کاما از هم جدا خواهند شد. همچنین دیگر شاهد نمایش درصد پیشرفت تکی فایل‌ها نیز نخواهیم بود و اینبار درصد پیشرفت کل batch گزارش می‌شود.
در یک چنین حالتی باید دقت داشت که تنظیم maxRequestLength در web.config برنامه الزامی است؛ زیرا به صورت پیش فرض محدودیت 4 مگابایتی ارسال فایل‌ها توسط ASP.NET اعمال می‌شود:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.web>
    <!-- The request length is in kilobytes, execution timeout is in seconds  -->
    <httpRuntime maxRequestLength="10240" executionTimeout="120" />
  </system.web>

  <system.webServer>
    <security>
      <requestFiltering>
        <!-- The content length is in bytes  -->
        <requestLimits maxAllowedContentLength="10485760"/>
      </requestFiltering>
    </security>
  </system.webServer>
</configuration>
```

عنوان: شروع به کار با Ember.js
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۴ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, SPA, EmberJS

Ember.js کتابخانه‌ای است جهت ساده سازی تولید برنامه‌های تک صفحه‌ای وب. برنامه‌هایی که شبیه به برنامه‌های دسکتاپ در مرورگر کاربر عمل می‌کنند. دو برنامه نویس اصلی آن Yehuda Katz که عضو اصلی تیم‌های jQuery و Ruby on Rails است و Tom Dale که ابتدا SproutCore را به وجود آورد و بعدها به Ember.js تغییر نام یافت، هستند.

منابع اصلی Ember.js

پیش از شروع به بحث نیاز است با تعدادی از سایت‌های اصلی مرتبط با Ember.js آشنا شد:
سایت اصلی: http://emberjs.com
مخزن کدهای آن: https://github.com/emberjs
انجمن اختصاصی پرسش و پاسخ: http://discuss.emberjs.com
موتور قالب‌های آن: http://handlebarsjs.com
لیست منابع مطالعاتی مرتبط مانند ویدیوهای آموزشی و لیست مقالات موجود: http://emberwatch.com
و بسته‌ی نیوگت آن: https://www.nuget.org/packages/EmberJS

مفاهیم پایه‌ای Ember.js

شیء Application

```
App = Ember.Application.create();
```

یک برنامه‌ی Ember.js با تعریف وهله‌ای از شیء Application آن آغاز می‌شود. با اینکار به صورت خودکار رویدادگردان‌هایی به صفحه اضافه می‌شوند. کامپوننت‌های پیش فرض آن ایجاد شده و همچنین قالب اصلی برنامه رندر می‌شود.

مسیر یابی

با مرور قسمت‌های مختلف برنامه توسط کاربر، نیاز است حالات برنامه را مدیریت کرد؛ اینجا است که کار قسمت مسیریابی شروع می‌شود. مسیریابی، منابع مورد نیاز جهت آدرس‌های مشخصی را تامین می‌کند.

```
App.Router.map(function() {
    this.resource('accounts'); // takes us to /accounts
    this.resource('gallery'); // takes us to /gallery
});
```

در اینجا نحوه‌ی تعریف آغازین مسیریابی Ember.js را مشاهده می‌کنید که توسط متد resource آن مسیرهای قابل ارائه توسط برنامه مشخص می‌شوند.
به این ترتیب مسیرهای accounts/ و gallery/ قابل پردازش خواهند شد.

این مسیرها، تو در تو نیز می‌توانند باشند. برای مثال:

```
App.Router.map(function() {
    this.resource('news', function() {
        this.resource('images', function () { // takes us to /news/images
            this.route('add');// takes us to /news/images/add
        });
    });
});
```

به این ترتیب نحوه‌ی تعریف مسیریابی آدرس news/images/add را مشاهده می‌کنید. همچنین در این مثال از دو متد resource و route استفاده شده‌است. از متد resource برای حالت تعریف اسامی استفاده کنید و از متد route برای تعریف افعال و تغییر دهنده‌ها. برای نمونه در اینجا فعل افزودن تصاویر با متد route مشخص شده‌است.

مدل‌ها

مدل‌ها همان اشیایی هستند که برنامه مورد استفاده قرار می‌دهد و می‌توانند یک آرایه‌ی ساده و یا اشیاء JSON دریافتی از وب سرور باشند.

حداقل به دو روش می‌توان مدل‌ها را تعریف کرد:

الف) با استفاده از افزونه‌ی Ember Data

ب) با کمک شیء Ember.Object

```
App.SiteLink = Ember.Object.extend({});
App.SiteLink.reopenClass({
    findAll: function() {
        var links = [];
        //… $.getJSON …

        return links;
    }
});
```

ابتدا یک زیرکلاس از Ember.Object به کمک متد extend ایجاد خواهد شد. سپس از متد توکار reopenClass برای توسعه‌ی API کمک خواهیم گرفت.

در ادامه متد دلخواهی را ایجاد کرده و برای مثال آرایه‌ای از اشیاء دلخواه جاوا اسکریپتی را بازگشت خواهیم داد.

پس از تعریف مدل، نیاز است آن‌را به سیستم مسیریابی معرفی کرد:

```
App.GalleryRoute = Ember.Route.extend({
    model: function() {
        return App.SiteLink.findAll();
    }
});
```

به این ترتیب زمانیکه کاربر به آدرس gallery/ مراجعه می‌کند، دسترسی به model وجود خواهد داشت. در اینجا model یک واژه‌ی کلیدی است.

کنترلرها

کنترلرها جهت ارائه‌ی اطلاعات مدل‌ها به View و قالب برنامه تعریف می‌شوند. در اینجا همیشه باید بخاطر داشت که model تامین کننده‌ی اطلاعات است. کنترلر جهت در معرض دید قرار دادن این اطلاعات، به View برنامه کاربرد دارد و مدل‌ها هیچ اطلاعی از وجود کنترلرها ندارند.

کنترلرها علاوه بر اطلاعات model، می‌توانند حاوی یک سری خواص و اشیاء صرفا نمایشی که قرار نیست در بانک اطلاعاتی ذخیره شوند نیز باشند.

در Ember.js قالب‌ها (templates) اطلاعات خود را از کنترلر دریافت می‌کنند. کنترلرها اطلاعات مدل را به همراه سایر خواص نمایشی مورد نیاز در اختیار View و قالب‌های برنامه قرار می‌دهند.

برای تعریف یک کنترلر می‌توان درون شیء مسیریابی، با تعریف متد setupController شروع کرد:

```
App.GalleryRoute = Ember.Route.extend({
    setupController: function(controller) {
        controller.set('content', ['red', 'yellow', 'blue']);
    }
});
```

در این مثال یک خاصیت دلخواه به نام content تعریف و سپس آرایه‌ای به آن انتساب داده شده‌است.

روش دوم تعریف کنترلرها با ایجاد یک زیر کلاس از شیء Ember.Controller انجام می‌شود:

```
App.GalleryController = Ember.Controller.extend({
    search: '',
    content: ['red', 'yellow', 'blue'],
    query: function() {
        var data = this.get('search');
        this.transitionToRoute('search', { query: data });
```

```
    }
});
```

قالب‌ها یا templates

قالب‌ها قسمت‌های اصلی رابط کاربری را تشکیل خواهند داد. در اینجا از کتابخانه‌ای به نام handlebars برای تهیه قالب‌های سمت کاربر کمک گرفته می‌شود.

```
<script type="text/x-handlebars" data-template-name="sayhello">
    Hello,
    <strong>{{firstName}} {{lastName}}</strong>!
</script>
```

این قالب‌ها توسط تگ اسکریپت تعریف شده و نوع آن‌ها text/x-handlebars مشخص می‌شود. به این ترتیب Ember.js، این قسمت از صفحه را یافته و عبارات داخل {{}} را با مقادیر دریافتی از کنترلر جایگزین می‌کند.

```
<script type="text/x-handlebars" data-template-name="sayhello">
    Hello,
    <strong>{{firstName}} {{lastName}}</strong>!

    {{#if person}}
    Welcome back,
    <strong>{{person.firstName}} {{person.lastName}}</strong>!
    {{/if}}

    <ul>
        {{#each friend in friends}}
        <li>
            {{friend.name}}
        </li>
        {{/each}}
    </ul>

    <img {{bindAttr src="link.url" }} />
    {{#linkTo ''about}}About{{/linkTo}}
</script>
```

در این مثال نحوه‌ی تعریف عبارات شرطی و یا یک حلقه را نیز مشاهده می‌کنید. همچنین امکان اتصال به ویژگی‌هایی مانند src یک تصویر و یا ایجاد لینک‌ها را نیز دارا است.

بهترین مرجع آشنایی با ریز جزئیات کتابخانه‌ی handlebars، مراجعه به سایت اصلی آن است.

قواعد پیش فرض نامگذاری در Ember.js

اگر به مثال‌های فوق دقت کرده باشید، خواصی مانند GalleryController و یا GalleryRoute به شیء App اضافه شده‌اند. این نوع نامگذاری‌ها در ember.js بر اساس روش convention over configuration کار می‌کنند. برای نمونه اگر مسیریابی خاصی را به نحو ذیل تعریف کردید:

```
this.resource('employees');
```

شیء مسیریابی آن App.EmployeesRoute

کنترلر آن App.EmployeesController

مدل آن App.Employee

View آن App.EmployeesView

و قالب آن employees

بهتر است تعریف شوند. به عبارتی اگر اینگونه تعریف شوند، به صورت خودکار توسط Ember.js یافت شده و هر کدام با مسئولیت‌های خاص مرتبط با آن‌ها پردازش می‌شوند و همچنین ارتباطات بین آن‌ها به صورت خودکار برقرار خواهد شد. به این ترتیب برنامه نظم بهتری خواهد یافت. با یک نگاه می‌توان قسمت‌های مختلف را تشخیص داد و همچنین کدنویسی پردازش و

اتصال قسمت‌های مختلف برنامه نیز به شدت کاهش می‌یابد.

تهیه‌ی اولین برنامه‌ی Ember.js

تا اینجا نگاهی مقدماتی داشتیم به اجزای تشکیل دهنده‌ی هسته‌ی Ember.js. در ادامه مثال ساده‌ای را جهت نمایش ساختار ابتدایی یک برنامه‌ی Ember.js، بررسی خواهیم کرد.

بسته‌ی Ember.js را همانطور که در قسمت منابع اصلی آن در ابتدای بحث عنوان شد، می‌توانید از سایت و یا مخزن کد آن دریافت کنید و یا اگر از VS.NET استفاده می‌کنید، تنها کافی است دستور ذیل را صادر نمائید:

```
PM> Install-Package EmberJS
```

پس از اضافه شدن فایل‌های js آن به پوشه‌ی Scripts برنامه، در همان پوشه، فایل جدید Scripts\app.js را نیز اضافه کنید. از آن برای افزودن تعاریف کدهای Ember.js استفاده خواهیم کرد.

در این حالت ترتیب تعریف اسکریپت‌های مورد نیاز صفحه به صورت ذیل خواهند بود:

```
<script src="Scripts/jquery-2.1.1.js" type="text/javascript"></script>
<script src="Scripts/handlebars.js" type="text/javascript"></script>
<script src="Scripts/ember.js" type="text/javascript"></script>
<script src="Scripts/app.js" type="text/javascript"></script>
```

کدهای ابتدایی فایل app.js جهت وهله سازی شیء Application و سپس تعریف مسیریابی صفحه‌ی index بر اساس روش convention over configuration به همراه تعریف یک کنترلر و افزودن متغیری به نام content به آن که با یک آرایه مقدار دهی شده‌است:

```
App = Ember.Application.create();
App.IndexRoute = Ember.Route.extend({
    setupController:function(controller) {
        controller.set('content', ['red', 'yellow', 'blue']);
    }
});
```

باید دقت داشت که تعریف مقدماتی Ember.Application.create به همراه یک سری تنظیمات پیش فرض نیز هست. برای مثال مسیریابی index به صورت خودکار به نحو ذیل توسط آن تعریف خواهد شد و نیازی به تعریف مجدد آن نیست:

```
App.Router.map(function() {
    this.resource('application');
    this.resource('index');
});
```

سپس برای اتصال این کنترلر به یک template خواهیم داشت:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <script src="Scripts/jquery-2.1.1.js" type="text/javascript"></script>
    <script src="Scripts/handlebars.js" type="text/javascript"></script>
    <script src="Scripts/ember.js" type="text/javascript"></script>
    <script src="Scripts/app.js" type="text/javascript"></script>
</head>
<body>
    <script type="text/x-handlebars" data-template-name="index">
        Hello,
        <strong>Welcome to Ember.js</strong>!
        <ul>
            {{#each item in content}}
            <li>
                {{item}}
            </li>
            {{/each}}
        </ul>
    </script>
```

```
</body>
</html>
```

توسط اسکریپتی از نوع text/x-handlebars، اطلاعات آرایه content دریافت و در طی یک حلقه در صفحه نمایش داده خواهد شد.

مقدار data-template-name در اینجا مهم است. اگر آن‌را به هر نام دیگری بجز index تنظیم کنید، منبع دریافت اطلاعات آن مشخص نخواهد بود. نام index در اینجا به معنای اتصال این قالب به اطلاعات ارائه شده توسط کنترلر index است.

تا همینجا اگر برنامه را اجرا کنید، به خوبی کار خواهد کرد. نکته‌ی دیگری که در مورد قالب‌های Ember.js قابل توجه هستند، قالب پیش فرض application است. با تعریف Ember.Application.create یک چنین قالبی نیز به ابتدای هر صفحه به صورت خودکار اضافه خواهد شد:

```
<body>
    <script type="text/x-handlebars" data-template-name="application">
        <h1>Header</h1>
        {{outlet}}
    </script>
```

outlet واژه‌ای است کلیدی که سبب رندر سایر قالب‌های تعریف شده در صفحه می‌گردد. مقدار data-template-name آن نیز به application تنظیم شده‌است (اگر این مقدار ذکر نگردد نیز به صورت خودکار از application استفاده می‌شود). برای مثال اگر بخواهید به تمام قالب‌های رندر شده در صفحات مختلف، مقدار ثابتی را اضافه کنید (مانند هدر یا منو)، می‌توان قالب application را به صورت دستی به نحو فوق اضافه کرد و آن‌را سفارشی سازی نمود.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS01.zip

نظرات خوانندگان

نویسنده: نصیری
تاریخ: ۱۳۹۳/۰۹/۱۸ ۱۷:۴۹

با سلام؛ با توجه به قدرت خیلی زیاد AngularJs و نیز اینکه پشتیبانی اون داره از طرف تیم قدرتمندی در google اداره میشه و دلیل بعدی اینکه تعداد خیلی زیادی از برنامه نویسهای دنیا به سمت این فریمورک رفتن بهتر نیست یادگرفتن و دنبال AngularJs رفتن را به EmberJs ترجیح داد
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۸ ۱۸:۱۸

- صرف نوشتن مطلبی در مورد موضوعی خاص، به معنای «بهترین بودن آن» و «برترین حالت ممکن» نیست. امروز راجع به ember.js مطلب نوشتم. شاید هفته‌های بعد در مورد meteor.js مطلب نوشتم.
- مقایسه‌ای در اینجا « AngularJS vs Ember »
- مقایسه‌ای کامل‌تر در اینجا « AngularJs vs EmberJs »
- « Rails JS frameworks: Ember.js vs. AngularJS »
- « AngularJS vs. Backbone.js vs. Ember.js »
- یک نمونه‌ی دیگر « The Top 10 Javascript MVC Frameworks Reviewed »
- این نظرسنجی را هم دنبال کنید: « آیا به یادگیری یا ادامه‌ی استفاده از AngularJS خواهید پرداخت؟ »

عنوان: **افزودن یک صفحه‌ی جدید و دریافت و نمایش اطلاعات از سرور به کمک Ember.js**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۵ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.NET Web API, jQuery, SPA, EmberJS

در قسمت قبل با مقدمات برپایی یک برنامه‌ی تک صفحه‌ای وب مبتنی بر Ember.js آشنا شدیم. مثال انتهای بحث آن نیز یک لیست ساده را نمایش می‌دهد. در ادامه همین برنامه را جهت نمایش لیستی از اشیاء JSON دریافتی از سرور تغییر خواهیم داد. همچنین یک صفحه‌ی about را نیز به آن اضافه خواهیم کرد.

پیشنیازهای سمت سرور

- ابتدا یک پروژه‌ی خالی ASP.NET را ایجاد کنید. نوع آن مهم نیست که Web Forms باشد یا MVC.
- سپس قصد داریم مدل کاربران سیستم را توسط یک ASP.NET Web API Controller در اختیار Ember.js قرار دهیم. مباحث پایه‌ای Web API نیز در وب فرم‌ها و MVC یکی است.

مدل سمت سرور برنامه چنین شکلی را دارد:

```
namespace EmberJS02.Models
{
    public class User
    {
        public int Id { set; get; }
        public string UserName { set; get; }
        public string Email { set; get; }
    }
}
```

کنترلر Web API ایی که این اطلاعات را در ختیار کلاینت‌ها قرار می‌دهد، به نحو ذیل تعریف می‌شود:

```
using System.Collections.Generic;
using System.Web.Http;
using EmberJS02.Models;

namespace EmberJS02.Controllers
{
    public class UsersController : ApiController
    {
        // GET api/<controller>
        public IEnumerable<User> Get()
        {
            return UsersDataSource.UsersList;
        }
    }
}
```

در اینجا UsersDataSource.UsersList صرفا یک لیست جنریک ساده از کلاس User است و کدهای کامل آن‌را می‌توانید از فایل پیوست انتهای بحث دریافت کنید.

همچنین فرض بر این است که مسیریابی سمت سرور ذیل را نیز به فایل global.asax.cs، جهت فعال سازی دسترسی به متدهای کنترلر UsersController تعریف کرده‌اید:

```
using System;
using System.Web.Http;
using System.Web.Routing;

namespace EmberJS02
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            RouteTable.Routes.MapHttpRoute(
               name: "DefaultApi",
```

```
                routeTemplate: "api/{controller}/{id}",
                defaults: new { id = RouteParameter.Optional }
                );
            }
        }
}
```

پیشنیازهای سمت کاربر

پیشنیازهای سمت کاربر این قسمت با قسمت « تهیه‌ی اولین برنامه‌ی Ember.js » دقیقا یکی است.
ابتدا فایل‌های مورد نیاز Ember.js به برنامه اضافه شده‌اند:

```
PM> Install-Package EmberJS
```

سپس یک فایل app.js با محتوای ذیل به پوشه‌ی Scripts اضافه شده‌است:

```
App = Ember.Application.create();
App.IndexRoute = Ember.Route.extend({
    setupController:function(controller) {
        controller.set('content', ['red', 'yellow', 'blue']);
    }
});
```

و در آخر یک فایل index.html با محتوای ذیل کار برپایی اولیه‌ی یک برنامه‌ی مبتنی بر Ember.js را انجام می‌دهد:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <script src="Scripts/jquery-2.1.1.js" type="text/javascript"></script>
    <script src="Scripts/handlebars.js" type="text/javascript"></script>
    <script src="Scripts/ember.js" type="text/javascript"></script>
    <script src="Scripts/app.js" type="text/javascript"></script>
</head>
<body>
    <script type="text/x-handlebars" data-template-name="application">
        <h1>Header</h1>
        {{outlet}}
    </script>
    <script type="text/x-handlebars" data-template-name="index">
        Hello,
        <strong>Welcome to Ember.js</strong>!
        <ul>
            {{#each item in content}}
            <li>
                {{item}}
            </li>
            {{/each}}
        </ul>
    </script>
</body>
</html>
```

تا اینجا را در قسمت قبل مطالعه کرده بودید.
در ادامه قصد داریم به هدر صفحه، دو لینک Home و About را اضافه کنیم؛ به نحوی که لینک Home به مسیریابی index و لینک About به مسیریابی about که صفحه‌ی جدید «درباره‌ی برنامه» را نمایش می‌دهد، اشاره کنند.

تعریف صفحه‌ی جدید About

برنامه‌های Ember.js، برنامه‌های تک صفحه‌ای وب هستند و صفحات جدید در آن‌ها به صورت یک template جدید تعریف می‌شوند که نهایتا متناظر با یک مسیریابی مشخص خواهند بود.
به همین جهت ابتدا در فایل app.js مسیریابی about را اضافه خواهیم کرد:

```
App.Router.map(function() {
    this.resource('about');
});
```

به این ترتیب با فراخوانی آدرس /about در مرورگر توسط کاربر، منابع مرتبط با این آدرس و قالب مخصوص آن، توسط Ember.js پردازش خواهند شد.
بنابراین به صفحه‌ی index.html برنامه مراجعه کرده و صفحه‌ی about را توسط یک قالب جدید تعریف می‌کنیم:

```
<script type="text/x-handlebars" data-template-name="about">
    <h2>Our about page</h2>
</script>
```

تنها نکته‌ی مهم در اینجا مقدار data-template-name است که سبب خواهد شد تا به مسیریابی about، به صورت خودکار متصل و مرتبط شود.

در این حالت اگر برنامه را در حالت معمولی اجرا کنید، خروجی خاصی را مشاهده نخواهید کرد. بنابراین نیاز است تا لینکی را جهت اشاره به این مسیر جدید به صفحه اضافه کنیم:

```
<script type="text/x-handlebars" data-template-name="application">
    <h1>Ember Demo App</h1>
    <ul class="nav">
        <li>{{#linkTo 'index'}}Home{{/linkTo}}</li>
        <li>{{#linkTo 'about'}}About{{/linkTo}}</li>
    </ul>
    {{outlet}}
</script>
```

اگر از قسمت قبل به خاطر داشته باشید، عنوان شد که قالب ویژه‌ی application به صورت خودکار با وهله سازی Ember.Application.create به صفحه اضافه می‌شود. اگر نیاز به سفارشی سازی آن وجود داشت، خصوصا جهت تعریف عناصری که باید در تمام صفحات حضور داشته باشند (مانند منوها)، می‌توان آن‌را به نحو فوق سفارشی سازی کرد.
در اینجا با استفاده از امکان یا directive ویژه‌ای به نام linkTo، لینک‌هایی به مسیریابی‌های index و about اضافه شده‌اند. به این ترتیب اگر کاربری برای مثال بر روی لینک About کلیک کند، کتابخانه‌ی Ember.js او را به صورت خودکار به مسیریابی about و سپس نمایش قالب مرتبط با آن (قالب about ایی که پیشتر تعریف کردیم) هدایت خواهد کرد؛ مانند تصویر ذیل:

همانطور که در آدرس صفحه نیز مشخص است، هرچند صفحه‌ی about نمایش داده شده‌است، اما هنوز نیز در همان صفحه‌ی اصلی برنامه قرار داریم. به علاوه در این قسمت جدید، همچنان منوی بالای صفحه نمایان است؛ از این جهت که تعاریف آن به قالب application اضافه شده‌اند.

دریافت و نمایش اطلاعات از سرور

اکنون که با نحوه‌ی تعریف یک صفحه‌ی جدید و برپایی سیم کشی‌های مرتبط با آن آشنا شدیم، می‌خواهیم صفحه‌ی دیگری را به نام Users به برنامه اضافه کنیم و در آن لیست کاربران ارائه شده توسط کنترلر Web API سمت سرور ابتدای بحث را نمایش دهیم.
بنابراین ابتدا مسیریابی جدید users را به صفحه اضافه می‌کنیم تا لیست کاربران، در آدرس users/ قابل دسترسی شود:

```
App.Router.map(function() {
    this.resource('about');
    this.resource('users');
});
```

سپس نیاز است مدلی را توسط فراخوانی Ember.Object.extend ایجاد کرده و به کمک متد reopenClass آن‌را توسعه دهیم:

```
App.UsersLink = Ember.Object.extend({});
App.UsersLink.reopenClass({
    findAll: function () {
        var users = [];
        $.getJSON('/api/users').then(function(response) {
            response.forEach(function(item) {
                users.pushObject(App.UsersLink.create(item));
            });
        });
        return users;
    }
});
```

در اینجا متد دلخواهی را به نام findAll اضافه کرده‌ایم که توسط متد getJSON جی‌کوئری، به مسیر /api/users سمت سرور متصل شده و لیست کاربران را از سرور به صورت JSON دریافت می‌کند. در اینجا خروجی دریافتی از سرور به کمک متد pushObject به آرایه کاربران اضافه خواهد شد. همچنین نحوه‌ی فراخوانی متد create مدل UsersLink را نیز در اینجا مشاهده می‌کنید (App.UsersLink.create).

پس از اینکه نحوه‌ی دریافت اطلاعات از سرور مشخص شد، باید اطلاعات این مدل را در اختیار مسیریابی Users قرار داد:

```
App.UsersRoute = Ember.Route.extend({
    model: function() {
        return App.UsersLink.findAll();
    }
});

App.UsersController = Ember.ObjectController.extend({
    customHeader : 'Our Users List'
});
```

به این ترتیب زمانیکه کاربر به مسیر users/ مراجعه می‌کند، سیستم مسیریابی می‌داند که اطلاعات مدل خود را باید از کجا تهیه نماید.
همچنین در کنترلری که تعریف شده، صرفا یک خاصیت سفارشی و دلخواه جدید، به نام customHeader برای نمایش در ابتدای صفحه تعریف و مقدار دهی گردیده‌است.
اکنون قالبی که قرار است اطلاعات مدل را نمایش دهد، چنین شکلی را خواهد داشت:

```
<script type="text/x-handlebars" data-template-name="users">
    <h2>{{customHeader}}</h2>
    <ul>
        {{#each item in model}}
        <li>
            {{item.Id}}-{{item.UserName}} ({{item.Email}})
```

```
        </li>
        {{/each}}
    </ul>
</script>
```

با تنظیم data-template-name به users سبب خواهیم شد تا این قالب اطلاعات خودش را از مسیریابی users دریافت کند. سپس یک حلقه نوشته‌ایم تا کلیه عناصر موجود در مدل را خوانده و در صفحه نمایش دهد. همچنین در عنوان قالب نیز از خاصیت سفارشی customHeader استفاده شده‌است:

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS02.zip

عنوان: **ساخت یک بلاگ ساده با Ember.js، قسمت اول**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۶ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, SPA, EmberJS

پس از آشنایی مقدماتی با اجزای مهم تشکیل دهنده‌ی Ember.js (^ و ^)، بهتر است این دانسته‌ها را جهت تکمیل یک پروژه‌ی ساده‌ی تک صفحه‌ای بلاگ، بکار بگیریم.

در این بلاگ می‌توان:

- یک مطلب جدید را ارسال کرد.
- مطالب قابل ویرایش و یا حذف هستند.
- مطالب بلاگ قسمت ارسال نظرات دارند.
- امکان گزارشگیری از آخرین نظرات ارسالی وجود دارد.
- سایت صفحات درباره و تماس با ما را نیز دارا است.

ساختار پوشه‌های برنامه

در تصویر ذیل، ساختار پوشه‌های برنامه بلاگ را ملاحظه می‌کنید. چون قسمت سمت کلاینت این برنامه کاملا جاوا اسکریپتی است، پوشه‌های Routes، Models، Libs، Controllers، App و Templates آن در پوشه‌ی Scripts تعریف شده‌اند و به این ترتیب می‌توان تفکیک بهتری را بین اجزای تشکیل دهنده‌ی یک برنامه‌ی تک صفحه‌ای وب Emeber.js پدید آورد.

Solution 'EmberJS03' (1 project)
.nuget
EmberJS03
Properties
References
App_Readme
Content
bootstrap.css
styles.css
Controllers
Models
Scripts
App
Controllers
Libs
ember-data.min.js
ember-handlebars-loader-0.0.1.js
ember.min.js
handlebars.min.js
jquery-2.1.1.min.js
jquery-2.1.1.min.map
localstorage_adapter.js
Models
Routes
Templates
Global.asax
Index.html
packages.config
Web.config

فایل CSS بوت استرپ نیز به پوشه‌ی Content اضافه شده‌است.

دریافت پیشنیازهای سمت کاربر برنامه

در ساختار پوشه‌های فوق، از پوشه‌ی Libs برای قرار دادن کتابخانه‌های پایه برنامه مانند jQuery و Ember.js استفاده خواهیم کرد. به این ترتیب:
- نیاز به آخرین نگارش‌های Ember.js و همچنین افزونه‌ی Ember-Data آن برای کار ساده‌تر با داده‌ها و سرور وجود دارد. این فایل‌ها را از آدرس ذیل می‌توانید دریافت کنید (نسخه‌های نیوگت به دلیل قدیمی بودن و به روز نشدن مداوم آن‌ها توصیه نمی‌شوند):
http://emberjs.com/builds/#/beta
برای حالت آزمایش برنامه، استفاده از فایل‌های دیباگ آن توصیه می‌شوند (فایل‌هایی با نام اصلی و بدون پسوند prod یا min). زیرا این فایل‌ها خطاها و اطلاعات بسیار مفصلی را از اشکالات رخ داده، در کنسول وب مرورگرها، فایرباگ و یا Developer tools آن‌ها نمایش می‌دهند. نسخه‌ی min برای حالت ارائه‌ی نهایی برنامه است. نسخه‌ی prod همان نسخه‌ی دیباگ است با حذف اطلاعات دیباگ (نسخه‌ی production فشرده نشده). نسخه‌ی فشرده شده‌ی prod آن، فایل min نهایی را تشکیل می‌دهد.
- دریافت جی‌کوئری
- آخرین نگارش handlebars.js را از سایت رسمی آن دریافت کنید: http://handlebarsjs.com
- Ember Handlebars Loader: https://github.com/michaelrkn/ember-handlebars-loader
- Ember Data Local Storage Adapter: https://github.com/kurko/ember-localstorage-adapter

ترتیب تعریف و قرارگیری این فایل‌ها را پس از دریافت، در فایل index.html قرار گرفته در ریشه‌ی سایت، در کدهای ذیل مشاهده می‌کنید:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Ember Blog</title>

  <link href="Content/bootstrap.css" rel="stylesheet" />
  <link href="Content/bootstrap-theme.css" rel="stylesheet" />
  <link href="Content/styles.css" rel="stylesheet" />

  <script src="Scripts/Libs/jquery-2.1.1.min.js" type="text/javascript"></script>
  <script src="Scripts/Libs/bootstrap.min.js" type="text/javascript"></script>
  <script src="Scripts/Libs/handlebars-v2.0.0.js" type="text/javascript"></script>
  <script src="Scripts/Libs/ember.js" type="text/javascript"></script>
  <script src="Scripts/Libs/ember-handlebars-loader-0.0.1.js" type="text/javascript"></script>
  <script src="Scripts/Libs/ember-data.js" type="text/javascript"></script>
  <script src="Scripts/Libs/localstorage_adapter.js" type="text/javascript"></script>
</head>
<body>

</body>
</html>
```

اصلاح فایل ember-handlebars-loader-0.0.1.js

اگر به فایل ember-handlebars-loader-0.0.1.js مراجعه کنید، مسیر فایل‌های قالب handlebars قسمت‌های مختلف برنامه را از پوشه‌ی templates واقع در ریشه‌ی سایت می‌خواند. با توجه به تصویر ساختار پوشه‌ی پروژه‌ی جاری، پوشه‌ی template به داخل پوشه‌ی Scripts منتقل شده‌است و نیاز به یک چنین اصلاحی دارد:

```
url: "Scripts/Templates/" + name + ".hbs",
```

کار اسکریپت ember-handlebars-loader-0.0.1.js بارگذاری خودکار فایل‌های hbs یا handlebars از پوشه‌ی قالب‌های سفارشی

برنامه است. در این حالت اگر برنامه را اجرا کنید، خطای 404 را دریافت خواهید کرد. از این جهت که mime-type پسوند hbs در IIS تعریف نشده‌است. اضافه کردن آن نیز ساده‌است. به فایل web.config برنامه مراجعه کرده و تغییر ذیل را اعمال کنید:

```
<system.webServer>
  <staticContent>
        <mimeMap fileExtension=".hbs" mimeType="text/x-handlebars-template" />
  </staticContent>
 </system.webServer>
```

مزیت استفاده از نسخه‌ی دیباگ ember.js در حین توسعه‌ی برنامه

نسخه‌ی دیباگ ember.js علاوه بر به همراه داشتن خطاهای بسیار جامع و توضیح علل مشکلات، مواردی را که در آینده منسوخ خواهند شد نیز توضیح می‌دهد:

برای مثال در اینجا عنوان شده‌است که دیگر از linkTo استفاده نکنید و آن‌را به link-to تغییر دهید.
همچنین اگر از مرورگر کروم استفاده می‌کنید، افزونه‌ی Ember Inspector را نیز می‌توانید نصب کنید تا اطلاعات بیشتری را از جزئیات مسیریابی‌های تعریف شده و قالب‌های Ember.js بتوان مشاهده کرد. این افزونه به صورت یک برگه‌ی جدید در Developer tools آن ظاهر می‌شود.

ایجاد شیء Application

همانطور که در قسمت‌های پیشین نیز عنوان شد (^ و ^)، یک برنامه‌ی Ember.js با تعریف وهله‌ای از شیء Application آن آغاز می‌شود. برای این منظور به پوشه‌ی App مراجعه کرده و فایل جدید Scripts\App\blogger.js را اضافه کنید؛ با این محتوا:

```
Blogger = Ember.Application.create();
```

مدخل این فایل را نیز پس از تعاریف وابستگی‌های اصلی برنامه، به فایل index.html اضافه خواهیم کرد:

```
<script src="Scripts/App/blogger.js" type="text/javascript"></script>
```

تا اینجا برپایی اولیه‌ی برنامه‌ی تک صفحه‌ای وب مبتنی بر Ember.js ما به پایان می‌رسد.

تعاریف مسیریابی و قالب‌ها

اکنون در ادامه قصد داریم لیستی از عناوین مطالب ارسالی را نمایش دهیم. در ابتدا این عناوین را از یک آرایه‌ی ثابت جاوا اسکریپتی دریافت خواهیم کرد و پس از آن از یک Web API کنترلر، جهت دریافت اطلاعات از سرور کمک خواهیم گرفت.

کار router در Ember.js، نگاشت آدرس درخواستی توسط کاربر، به یک route یا مسیریابی تعریف شده‌است. به این ترتیب مدل، کنترلر و قالب آن route به صورت خودکار بارگذاری و پردازش خواهند.
با مراجعه به ریشه‌ی سایت، فایل index.html بارگذاری می‌شود.

در اینجا تصویری از صفحه‌ی آغازین بلاگ را مشاهده می‌کنید. در آن صفحه‌ی posts همان ریشه‌ی سایت نیز می‌باشد. بنابراین نیاز است ابتدا مسیریابی آن‌را تعریف کرد. برای این منظور، فایل جدید Scripts\App\router.js را به پوشه‌ی App اضافه کنید؛ با این محتوا:

```
Blogger.Router.map(function () {
   this.resource('posts', { path: '/' });
});
```

علت تعریف قسمت path این است که ریشه‌ی سایت (/) نیز به مسیریابی posts ختم شود. در غیر اینصورت کاربر با مراجعه به ریشه‌ی سایت، یک صفحه‌ی خالی را مشاهده خواهد کرد؛ زیرا به صورت پیش فرض، آدرس قابل ترجمه‌ی یک صفحه، با آدرس و نام مسیریابی آن یکی است.

همچنین مدخل آن‌را نیز در فایل index.html تعریف نمائید:

```
<script src="Scripts/App/blogger.js" type="text/javascript"></script>
<script src="Scripts/App/router.js" type="text/javascript"></script>
```

در اینجا Blogger همان شیء Application برنامه است که پیشتر در فایل Scripts\App\blogger.js تعریف کردیم. سپس به کمک متد Blogger.Router.map، اولین مسیریابی برنامه را افزوده‌ایم.

افزودن مسیریابی و قالب posts

در ادامه فایل جدید Scripts\Templates\posts.hbs را اضافه کنید. به این ترتیب قالب خالی مطالب به پوشه‌ی templates اضافه می‌شود. محتوای این فایل را به نحو ذیل تنظیم کنید:

```
<div class="container">
  <h1>Emeber.js blog</h1>
  <ul>
   <li>Item 1</li>
   <li>Item 2</li>
   <li>Item 3</li>
  </ul>
</div>
```

در اینجا دیگر نیازی به ذکر تگ script از نوع text/x-handlebars نیست.

برای بارگذاری این قالب نیاز است آن‌را به template loader توضیح داده شده در ابتدای بحث، در فایل index.html اضافه کنیم:

```
<script>
 EmberHandlebarsLoader.loadTemplates([
 'posts'
 ]);
</script>
```

اکنون برنامه را اجرا کنید. به تصویر ذیل خواهید رسید که در آن افزونه‌ی Ember Inspector نیز نمایش داده شده‌است:

افزودن مسیریابی و قالب about

در ادامه قصد داریم صفحه‌ی about را اضافه کنیم. مجددا با افزودن مسیریابی آن به فایل Scripts\App\router.js شروع می‌کنیم:

```
Blogger.Router.map(function () {
  this.resource('posts', { path: '/' });
  this.resource('about');
});
```

سپس فایل قالب آن‌را در مسیر Scripts\Templates\about.hbs ایجاد خواهیم کرد؛ برای مثال با این محتوای فرضی:

```
<h1>About Ember Blog</h1>
<p>Bla bla bla!</p>
```

اکنون نام این فایل را به template loader فایل index.html اضافه می‌کنیم.

```
<script>
 EmberHandlebarsLoader.loadTemplates([
 'posts', 'about'
 ]);
</script>
```

اگر از قسمت قبل به خاطر داشته باشید، ساختار کلی قالب‌های ember.js یک چنین شکلی را دارد:

```
<script type="text/x-handlebars" data-template-name="about">

</script>
```

اسکریپت template loader، این تعاریف را به صورت خودکار اضافه می‌کند. مقدار data-template-name را نیز به نام فایل متناظر با آن تنظیم خواهد کرد و چون ember.js بر اساس ایده‌ی convention over configuration کار می‌کند، مسیریابی about با کنترلری به همین نام و قالبی هم نام پردازش خواهد شد. بنابراین نام فایل‌های قالب را باید بر اساس مسیریابی‌های متناظر با آن‌ها تعیین کرد.
برای آزمایش این مسیر و قالب جدید آن، آدرس http://localhost/#/about را بررسی کنید.

اضافه کردن منوی ثابت بالای سایت

روش اول این است که به ابتدای هر دو قالب about.hbs و posts.hbs، تعاریف منو را اضافه کنیم. مشکل این‌کار، تکرار کدها و پایین آمدن قابلیت نگهداری برنامه است. روش بهتر، افزودن کدهای مشترک بین صفحات، در قالب application برنامه است. نمونه‌ی آن‌را در مثال قسمت قبل مشاهده کرده‌اید. در اینجا چون قصد نداریم به صورت دستی قالب‌ها را به صفحه اضافه کنیم و همچنین برای ساده شدن نگهداری برنامه، قالب‌ها را در فایل‌های مجزایی قرار داده‌ایم، تنها کافی است فایل جدید Scripts\Templates\application.hbs را به پوشه‌ی قالب‌های برنامه اضافه کنیم؛ با این محتوا:

```
<div class='container'>
 <nav class='navbar navbar-default' role='navigation'>
  <ul class='nav navbar-nav'>
  <li>{{#link-to 'posts'}}Posts{{/link-to}}</li>
  <li>{{#link-to 'about'}}About{{/link-to}}</li>
  </ul>
 </nav>

  {{outlet}}
</div>
```

و سپس همانند قبل، نام فایل قالب اضافه شده را به template loader فایل index.html اضافه می‌کنیم:

```
<script>
 EmberHandlebarsLoader.loadTemplates([
 'posts', 'about', 'application'
 ]);
</script>
```

افزودن مسیریابی و قالب contact

برای افزودن صفحه‌ی تماس با مای سایت، ابتدا مسیریابی آن‌را در فایل Scripts\App\router.js تعریف می‌کنیم:

```
Blogger.Router.map(function () {
  this.resource('posts', { path: '/' });
  this.resource('about');
  this.resource('contact');
});
```

سپس قالب متناظر با آن‌را به نام Scripts\Templates\contact.hbs اضافه خواهیم کرد؛ فعلا با این محتوای اولیه:

```
<h1>Contact</h1>
<ul>
  <li>Phone: ...</li>
  <li>Email: ...</li>
</ul>
```

و بعد template loader فایل index.html را از وجود آن مطلع خواهیم کرد:

```
<script>
 EmberHandlebarsLoader.loadTemplates([
 'posts', 'about', 'application', 'contact' ]);
</script>
```

همچنین لینکی به مسیریابی آن‌را به فایل Scripts\Templates\application.hbs که منوی سایت در آن تعریف شده‌است، اضافه می‌کنیم:

```
<div class='container'>
 <nav class='navbar navbar-default' role='navigation'>
  <ul class='nav navbar-nav'>
  <li>{{#link-to 'posts'}}Posts{{/link-to}}</li>
  <li>{{#link-to 'about'}}About{{/link-to}}</li>
  <li>{{#link-to 'contact'}}Contact{{/link-to}}</li>
  </ul>
 </nav>

  {{outlet}}
</div>
```

تعریف مسیریابی تو در تو در صفحه‌ی contact

در حال حاضر صفحه‌ی Contact، ایمیل و شماره تماس را در همان بار اول نمایش می‌دهد. می‌خواهیم این دو را تبدیل به دو لینک جدید کنیم که با کلیک بر روی هر کدام، محتوای مرتبط، در قسمتی از همان صفحه بارگذاری شده و نمایش داده شود.
برای اینکار نیاز است مسیریابی را تو در تو تعریف کنیم:

```
Blogger.Router.map(function () {
  this.resource('posts', { path: '/' });
  this.resource('about');
  this.resource('contact', function () {
   this.resource('email');
   this.resource('phone');
  });
});
```

اگر مسیریابی‌های email و یا phone را به صورت مستقل مانند about و یا posts تعریف کنیم، با کلیک کاربر بر روی لینک متناظر با هر کدام، یک صفحه‌ی کاملا جدید نمایش داده می‌شود. اما در اینجا قصد داریم تنها قسمت کوچکی از همان صفحه‌ی contact را با محتوای ایمیل و یا شماره تماس جایگزین نمائیم. به همین جهت مسیریابی‌های متناظر را در اینجا به صورت تو در تو و ذیل مسیریابی contact تعریف کرده‌ایم.

پس از آن دو فایل قالب جدید Scripts\Templates\email.hbs را با محتوای:

```
<h2>Email</h2>
<p>
<span></span> Email name@site.com.
</p>
```

و فایل قالب Scripts\Templates\phone.hbs را با محتوای:

```
<h2>Phone</h2>
<p>
<span></span> Call 12345678.
</p>
```

به پوشه‌ی قالب‌ها اضافه نمائید به همراه معرفی نام آن‌ها به template loader برنامه در صفحه‌ی index.html :

```
<script>
 EmberHandlebarsLoader.loadTemplates([
 'posts', 'about', 'application', 'contact', 'email', 'phone' ]);
</script>
```

اکنون به قالب contact.hbs مجددا مراجعه کرده و تعاریف آن را به نحو ذیل تغییر دهید:

```
<h1>Contact</h1>
<div class="row">
  <div class="col-md-6">
   <p>
    Want to get in touch?
    <ul>
      <li>{{#link-to 'phone'}}Phone{{/link-to}}</li>
      <li>{{#link-to 'email'}}Email{{/link-to}}</li>
    </ul>
   </p>
  </div>
  <div class="col-md-6">
   {{outlet}}
  </div>
</div>
```

در اینجا دو لینک به مسیریابی‌های Phone و Email تعریف شده‌اند. همچنین {{outlet}} نیز قابل مشاهده است. با کلیک بر روی لینک Phone، مسیریابی آن فعال شده و سپس قالب متناظر با آن در قسمت {{outlet}} رندر می‌شود. در مورد لینک Email نیز به همین نحو رفتار خواهد شد.

در اینجا نحوه‌ی پردازش مسیریابی contact را ملاحظه می‌کنید. ابتدا درخواستی جهت مشاهده‌ی آدرس http://localhost/#/contact دریافت می‌شود. سپس router این درخواست را به مسیریابی همنامی منتقل می‌کند. این مسیریابی ابتدا قالب عمومی application را رندر کرده و سپس قالب اصلی و همنام مسیریابی جاری یا همان contact.hbs را رندر می‌کند. در این صفحه چون مسیریابی تو در تویی تعریف شده‌است، اگر درخواستی برای مشاهده‌ی http://localhost/#/contact/phone دریافت شود، محتوای آن‌را در {{outlet}} قالب contact.hbs جاری رندر می‌کند.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS03_01.zip

عنوان: ساخت یک بلاگ ساده با Ember.js، قسمت دوم
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۷ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, SPA, EmberJS

پس از تهیه ساختار اولیه‌ی بلاگی مبتنی بر ember.js در قسمت قبل ، در ادامه قصد داریم امکانات تعاملی را به آن اضافه کنیم. بنابراین کار را با تعریف کنترلرها که تعیین کننده‌ی رفتار برنامه هستند، ادامه می‌دهیم.

اضافه کردن دکمه‌ی More info به صفحه‌ی About و مدیریت کلیک بر روی آن

فایل Scripts\Templates\about.hbs را گشوده و سپس محتوای فعلی آن را به نحو ذیل تکمیل کنید:

```
<h2>About Ember Blog</h2>

<p>Bla bla bla!</p>

<button class="btn btn-primary" {{action 'showRealName' }}>more info</button>
```

در ember.js اگر قصد مدیریت عملی را که قرار است توسط کلیک بر روی المانی رخ دهد، داشته باشیم، می‌توان از handlebar helper ایی به نام action استفاده کرد. سپس برای تهیه کدهای مرتبط با آن، این اکشن را باید در کنترلر متناظر با route جاری (مسیریابی about) اضافه کنیم.
به همین جهت فایل جدید Scripts\Controllers\about.js را در پوشه‌ی کنترلرهای سمت کاربر اضافه کنید (نام آن با نام مسیریابی یکی است)؛ با این محتوا:

```
Blogger.AboutController = Ember.Controller.extend({
  actions: {
   showRealName: function () {
    alert("You clicked at showRealName of AboutController.");
   }
  }
});
```

کنترلرها به صورت یک خاصیت جدید به شیء Application برنامه اضافه می‌شوند. مطابق اصول نامگذاری ember.js، نام خاصیت کنترلر با حروف بزرگ متناظر با route آن شروع می‌شود و به نام Controller ختم خواهد شد. به این ترتیب ember.js هرگاه قصد پردازش مسیریابی about را داشته باشد، می‌داند که باید از کدام شیء جهت پردازش اعمال کاربر استفاده کند.
در ادامه این خاصیت را با تهیه یک زیرکلاس از کلاس پایه Controller تهیه شده توسط ember.js مقدار دهی می‌کنیم. به این ترتیب به کلیه امکانات این کلاس پایه دسترسی خواهیم داشت؛ به علاوه می‌توان ویژگی‌های سفارشی را نیز به آن افزود. برای مثال در اینجا در قسمت actions آن، دقیقا مطابق نام اکشنی که در فایل about.hbs تعریف کرده‌ایم، یک متد جدید اضافه شده‌است.

پس از تعریف کنترلر about.js نیاز است مدخل متناظر با آن‌را به فایل index.html برنامه نیز در انتهای تعاریف موجود، اضافه کرد:

```
<script src="Scripts/Controllers/about.js" type="text/javascript"></script>
```

اکنون یکبار برنامه را اجرا کرده و در صفحه‌ی about بر روی دکمه‌ی more info کلیک کنید.

اضافه کردن دکمه‌ی ارسال پیام خصوصی به صفحه‌ی Contact و مدیریت کلیک بر روی آن

در ادامه به قالب فعلی Scripts\Templates\contact.hbs یک دکمه را جهت ارسال پیام خصوصی اضافه می‌کنیم.

```
<h1>Contact</h1>
<div class="row">
  <div class="col-md-6">
   <p>
    Want to get in touch?
    <ul>
      <li>{{#link-to 'phone'}}Phone{{/link-to}}</li>
      <li>{{#link-to 'email'}}Email{{/link-to}}</li>
    </ul>
   </p>

   <p>
    Or, click here to send a secret message:
   </p>
   <button class="btn btn-primary" {{action 'sendMessage' }}>Send message</button>
  </div>
  <div class="col-md-6">
   {{outlet}}
  </div>
</div>
```

سپس برای مدیریت اکشن جدید sendMessage نیاز است کنترلر آن‌را نیز تعریف کنیم. با توجه به نام مسیریابی جاری، نام این کنترلر نیز contact خواهد بود. برای این منظور ابتدا فایل جدید Scripts\Controllers\contact.js را اضافه نمائید؛ با این محتوا:

```
Blogger.ContactController = Ember.Controller.extend({
  actions: {
   sendMessage: function () {
    var message = prompt('Type your message here:');
   }
  }
});
```

همچنین مدخل متناظر با فایل contact.js نیز باید به صفحه‌ی index.html اضافه شود:

```
<script src="Scripts/Controllers/contact.js" type="text/javascript"></script>
```

نمایش تصویری تعاملی در صفحه‌ی about

تا اینجا با نحوه‌ی تعریف اکشن‌ها در قالب‌ها و مدیریت آن‌ها توسط کنترلرهای متناظر آشنا شدیم. در ادامه قصد داریم با اصول binding اطلاعات در ember.js آشنا شویم. برای مثال فرض کنید می‌خواهیم دکمه‌ای را در صفحه‌ی about قرار داده و با کلیک بر روی آن، لوگوی ember.js را که به صورت یک تصویر مخفی در صفحه قرار دارد، نمایان کنیم. برای اینکار نیاز است خاصیتی را در کنترلر متناظر، تعریف کرده و سپس آن‌را به template جاری bind کرد.

برای این منظور فایل Scripts\Templates\about.hbs را گشوده و تعاریف موجود آن‌را به نحو ذیل تکمیل کنید:

```
<h2>About Ember Blog</h2>
<p>Bla bla bla!</p>
<button class="btn btn-primary" {{action 'showRealName' }}>more info</button>

{{#if isAuthorShowing}}
<button class="btn btn-warning" {{action 'hideAuthor' }}>Hide Image</button>
<p><img src="Content/images/ember-productivity-sm.png"></p>
{{else}}
<button class="btn btn-info" {{action 'showAuthor' }}>Show Image</button>
{{/if}}
```

در اینجا بر اساس مقدار خاصیت isAuthorShowing تصمیم گیری خواهد شد که آیا تصویر لوگوی ember.js نمایش داده شود یا خیر. همچنین دو اکشن نمایش و مخفی کردن تصویر نیز اضافه شده‌اند که با کلیک بر روی هر کدام، سبب تغییر وضعیت خاصیت isAuthorShowing خواهیم شد.

کنترلر about (فایل Scripts\Controllers\about.js) جهت مدیریت این خاصیت جدید، به همراه دو اکشن تعریف شده، اینبار به نحو ذیل تغییر خواهد یافت:

```
Blogger.AboutController = Ember.Controller.extend({
  isAuthorShowing: false,
  actions: {
   showRealName: function () {
    alert("You clicked at showRealName of AboutController.");
   },
   showAuthor: function () {
    this.set('isAuthorShowing', true);
   },
   hideAuthor: function () {
    this.set('isAuthorShowing', false);
   }
  }
});
```

ابتدا خاصیت isAuthorShowing به کنترلر اضافه شده‌است. از این خاصیت بار اولی که مسیر http://localhost:25918/#/about توسط کاربر درخواست می‌شود، استفاده خواهد شد.

سپس در دو متد showAuthor و hideAuthor که به اکشن‌های دو دکمه‌ی جدید تعریف شده در قالب about متصل خواهند شد، نحوه‌ی تغییر مقدار خاصیت isAuthorShowing را توسط متد set ملاحظه می‌کنید.

این قسمت مهم‌ترین تفاوت ember.js با jQuery است. در jQuery مستقیما المان‌های صفحه در همانجا تغییر داده می‌شوند. در ember.js منطق مدیریت کننده‌ی رابط کاربری و کدهای قالب متناظر با آن از هم جدا شده‌اند تا بتوان یک برنامه‌ی بزرگ را بهتر مدیریت کرد. همچنین در اینجا مشخص است که هر قسمت و هر فایل، چه ارتباطی با سایر اجزای تعریف شده دارد و چگونه به هم متصل شده‌اند و اینبار شاهد انبوهی از کدهای جاوا اسکریپتی مخلوط بین المان‌های HTML صفحه نیستیم.

نمایش پیامی به کاربر پس از ارسال پیام خصوصی در صفحه‌ی تماس با ما

قصد داریم ویژگی مشابهی را به صفحه‌ی contact نیز اضافه کنیم. اگر کاربر بر روی دکمه‌ی ارسال پیام کلیک کرد، پیام تشکری به همراه عددی ویژه به او نمایش خواهیم داد.

برای این‌کار قالب Scripts\Templates\contact.hbs را به نحو ذیل تکمیل کنید:

```
<h1>Contact</h1>
<div class="row">
  <div class="col-md-6">
   <p>
    Want to get in touch?
    <ul>
      <li>{{#link-to 'phone'}}Phone{{/link-to}}</li>
      <li>{{#link-to 'email'}}Email{{/link-to}}</li>
    </ul>
   </p>
   {{#if messageSent}}
   <p>
    Thank you. Your message has been sent.
    Your confirmation number is {{confirmationNumber}}.
   </p>
   {{else}}
   <p>
    Or, click here to send a secret message:
   </p>
   <button class="btn btn-primary" {{action 'sendMessage' }}>Send message</button>
   {{/if}}
  </div>
  <div class="col-md-6">
   {{outlet}}
  </div>
</div>
```

در آن شرط بررسی messageSent if اضافه شده‌است؛ به همراه نمایش confirmationNumber در انتهای پیام تشکر.

برای تعریف منطق مرتبط با این خواص، به کنترلر contact واقع در فایل Scripts\Controllers\contact.js مراجعه کرده و آن‌را به نحو ذیل تغییر می‌دهیم:

```
Blogger.ContactController = Ember.Controller.extend({
  messageSent: false,
  actions: {
   sendMessage: function () {
    var message = prompt('Type your message here:');
    if (message) {
      this.set('confirmationNumber', Math.round(Math.random() * 100000));
      this.set('messageSent', true);
    }
   }
  }
});
```

همانطور که مشاهده می‌کنید، مقدار اولیه خاصیت messageSent مساوی false است. بنابراین در قالب contact.hbs قسمت else شرط نمایش داده می‌شود. اگر کاربر پیامی را وارد کند، خاصیت confirmationNumber به یک عدد اتفاقی و خاصیت messageSent به true تنظیم خواهد شد. به این ترتیب اینبار به صورت خودکار پیام تشکر به همراه عددی اتفاقی، به کاربر نمایش داده می‌شود.

بنابراین به صورت خلاصه، کار کنترلر، مدیریت منطق نمایشی برنامه است و برای اینکار حداقل دو مکانیزم را ارائه می‌دهد: اکشن‌ها و خواص. اکشن‌ها بیانگر نوعی رفتار هستند؛ برای مثال نمایش یک popup و یا تغییر مقدار یک خاصیت. مقدار خواص را می‌توان مستقیما در صفحه نمایش داد و یا از آن‌ها جهت پردازش عبارات شرطی و نمایش قسمت خاصی از قالب جاری نیز می‌توان کمک گرفت.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS03_02.zip

عنوان: ساخت یک بلاگ ساده با Ember.js، قسمت سوم
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۸ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, SPA, EmberJS

پس از ایجاد کنترلرها ، در این قسمت سعی خواهیم کرد تا آرایه‌ای ثابت از مطالب و نظرات را در سایت نمایش دهیم. همچنین امکان ویرایش اطلاعات را نیز به این آرایه‌های جاوا اسکریپتی مدل، اضافه خواهیم کرد.

تعریف مدل سمت کاربر برنامه

فایل جدید Scripts\App\store.js را اضافه کرده و محتوای آن‌را به نحو ذیل تغییر دهید:

```
var posts = [
  {
      id: '1',
      title: "Getting Started with Ember.js",
      body: "Bla bla bla 1."
  },
  {
      id: '2',
      title: "Routes and Templates",
      body: "Bla bla bla 2."
  },
  {
      id: '3',
      title: "Controllers",
      body: "Bla bla bla 3."
  }
];

var comments = [
    {
        id: '1',
        postId: '3',
        text: 'Thanks!'
    },
    {
        id: '2',
        postId: '3',
        text: 'Good to know that!'
    },
    {
        id: '3',
        postId: '1',
        text: 'Great!'
    }
];
```

در اینجا دو آرایه ثابت از اشیاء مطالب و نظرات را مشاهده می‌کنید.
سپس جهت استفاده از آن، تعریف مدخل آن‌را به فایل index.html، پیش از تعاریف کنترلرها اضافه خواهیم کرد:

```
<script src="Scripts/App/store.js" type="text/javascript"></script>
```

ویرایش قالب مطالب برای نمایش لیستی از عناوین ارسالی

قالب فعلی Scripts\Templates\posts.hbs صرفا دارای یک سری عنوان درج شده به صورت مستقیم در صفحه است. اکنون قصد داریم آن‌را جهت نمایش لیستی از آرایه مطالب تغییر دهیم.

همانطور که در تصویر ملاحظه می‌کنید، با درخواست آدرس صفحه‌ی مطالب، router آن مسیریابی متناظری را یافته و سپس بر این اساس، template، کنترلر و مدلی را انتخاب می‌کند. به صورت پیش فرض، قالب و کنترلر انتخاب شده، مواردی هستند همنام با مسیریابی جاری. اما مقدار پیش فرضی برای model وجود ندارد و باید آن‌را به صورت دستی مشخص کرد.
برای این منظور فایل Scripts\Routes\posts.js را به پوشه‌ی routes با محتوای ذیل اضافه کنید:

```
Blogger.PostsRoute = Ember.Route.extend({
    controllerName: 'posts',
    renderTemplare: function () {
        this.render('posts');
    },
    model: function () {
        return posts;
    }
});
```

در اینجا صرفا جهت نمایش پیش فرض‌ها و نحوه‌ی کار یک route، دو خاصیت controllerName و renderTemplare آن نیز مقدار دهی شده‌اند. این دو خاصیت به صورت پیش فرض، همنام مسیریابی جاری مقدار دهی می‌شوند و نیازی به ذکر صریح آن‌ها نیست. اما خاصیت model یک مسیریابی است که باید دقیقا مشخص شود. در اینجا مقدار آن‌را به آرایه posts تعریف شده در فایل Scripts\App\store.js تنظیم کرده‌ایم. به این ترتیب مدل تعریف شده در اینجا، به صورت خودکار در کنترلر posts و قالب متناظر با آن، قابل استفاده خواهد بود.
همچنین اگر به خاطر داشته باشید، در پوشه‌ی کنترلرها فایل posts.js تعریف نشده‌است. اگر اینکار صورت نگیرد، ember.js به صورت خودکار کنترلر پیش فرضی را ایجاد خواهد کرد. در کل، یک قالب هیچگاه به صورت مستقیم با مدل کار نمی‌کند. این کنترلر است که مدل را در اختیار یک قالب قرار می‌دهد.
سپس مدخل تعریف این فایل را به فایل index.html، پس از تعاریف کنترلرها اضافه نمائید:

```
<script src="Scripts/Routes/posts.js" type="text/javascript"></script>
```

اکنون فایل Scripts\Templates\posts.hbs را گشوده و به نحو ذیل، جهت نمایش عناوین مطالب، ویرایش کنید:

```
<h2>Emeber.js blog</h2>
<ul>
    {{#each post in model}}
    <li>{{post.title}}</li>
    {{/each}}
</ul>
```

در این قالب، حلقه‌ای بر روی عناصر model تشکیل شده و سپس خاصیت title هر عضو نمایش داده می‌شود.

نمایش لیست آخرین نظرات ارسالی

در ادامه قصد داریم تا آرایه comments ابتدای بحث را در صفحه‌ای جدید نمایش دهیم. بنابراین نیاز است تا ابتدا مسیریابی آن تعریف شود. بنابراین فایل Scripts\App\router.js را گشوده و مسیریابی جدید recent-comments را به آن اضافه کنید:

```
Blogger.Router.map(function () {
    this.resource('posts', { path: '/' });
    this.resource('about');
    this.resource('contact', function () {
        this.resource('email');
        this.resource('phone');
    });
    this.resource('recent-comments');
});
```

سپس جهت تعیین مدل این مسیریابی جدید نیاز است تا فایل Scripts\Routes\recent-comments.js را در پوشه‌ی routes با محتوای ذیل اضافه کرد:

```
Blogger.RecentCommentsRoute = Ember.Route.extend({
    model: function () {
        return comments;
    }
});
```

در اینجا آرایه comments بازگشتی، همان آرایه‌ای است که در ابتدای بحث در فایل Scripts\App\store.js تعریف کردیم.
همچنین نیاز است تا تعریف مدخل این فایل جدید را نیز به انتهای تعاریف مداخل فایل index.html اضافه کنیم:

```
<script src="Scripts/Routes/recent-comments.js" type="text/javascript"></script>
```

اکنون قالب application واقع در فایل Scripts\Templates\application.hbs را جهت افزودن منوی مرتبط با این مسیریابی جدید، به نحو ذیل ویرایش خواهیم کرد:

```
<div class='container'>
    <nav class='navbar navbar-default' role='navigation'>
        <ul class='nav navbar-nav'>
            <li>{{#link-to 'posts'}}Posts{{/link-to}}</li>
            <li>{{#link-to 'recent-comments'}}Recent comments{{/link-to}}</li>
            <li>{{#link-to 'about'}}About{{/link-to}}</li>
            <li>{{#link-to 'contact'}}Contact{{/link-to}}</li>
        </ul>
    </nav>
    {{outlet}}
</div>
```

و در آخر قالب جدید Scripts\Templates\recent-comments.hbs را برای نمایش لیست آخرین نظرات، با محتوای زیر اضافه می‌کنیم:

```
<h1>Recent comments</h1>
<ul>
  {{#each comment in model}}
    <li>{{comment.text}}</li>
  {{/each}}
</ul>
```

برای فعال شدن آن نیاز است تا تعریف این قالب جدید را به template loader برنامه، در فایل index.html اضافه کنیم:

```
<script type="text/javascript">
    EmberHandlebarsLoader.loadTemplates([
       'posts', 'about', 'application', 'contact', 'email', 'phone',
       'recent-comments'
    ]);
</script>
```

نمایش مجزای هر مطلب در یک صفحه‌ی جدید

تا اینجا در صفحه‌ی اول سایت، لیست عناوین مطالب را نمایش دادیم. در ادامه نیاز است تا بتوان هر عنوان را به صفحه‌ی متناظر و اختصاصی آن لینک کرد؛ برای مثال لینکی مانند http://localhost:25918/#/posts/3 به سومین مطلب ارسالی اشاره می‌کند. Ember.js به عدد 3 در اینجا، یک dynamic segment می‌گوید. از این جهت که مقدار آن بر اساس شماره مطلب درخواستی، متفاوت خواهد بود. برای پردازش این نوع آدرس‌ها نیاز است مسیریابی ویژه‌ای را تعریف کرد. فایل Scripts\App\router.js را گشوده و سپس مسیریابی post را به نحو ذیل به آن اضافه نمائید:

```
Blogger.Router.map(function () {
    this.resource('posts', { path: '/' });
    this.resource('about');
    this.resource('contact', function () {
        this.resource('email');
        this.resource('phone');
    });
    this.resource('recent-comments');
    this.resource('post', { path: 'posts/:post_id' });
});
```

قسمت پویای مسیریابی با یک : مشخص می‌شود.

با توجه به اینکه این مسیریابی جدید post نام گرفت (جهت نمایش یک مطلب)، به صورت خودکار، کنترلر و قالبی به همین نام را بارگذاری می‌کند. همچنین مدل خود را نیز باید از مسیریابی خاص خود دریافت کند. بنابراین فایل جدید Scripts\Routes\post.js را در پوشه‌ی routes با محتوای ذیل اضافه کنید:

```
Blogger.PostRoute = Ember.Route.extend({
    model: function (params) {
        return posts.findBy('id', params.post_id);
    }
});
```

در اینجا مدل مسیریابی post بر اساس پارامتری به نام params تعیین می‌شود. این پارامتر حاوی مقدار متغیر پویای post_id که در مسیریابی جدید post مشخص کردیم می‌باشد. در ادامه از آرایه posts تعریف شده در ابتدای بحث، توسط متد findBy که توسط Ember.js اضافه شده‌است، عنصری را که خاصیت id آن مساوی post_id دریافتی است، انتخاب کرده و به عنوان مقدار مدل بازگشت می‌دهیم.

برای مثال، جهت آدرس http://localhost:25918/#/posts/3، مقدار post_id به صورت خودکار به عدد 3 تنظیم می‌شود.

پس از آن نیاز است مدخل این فایل جدید را در صفحه‌ی index.html نیز اضافه کنیم:

```
<script src="Scripts/Routes/post.js" type="text/javascript"></script>
```

در ادامه برای نمایش اطلاعات مدل نیاز است قالب جدید Scripts\Templates\post.hbs را با محتوای زیر اضافه کنیم:

```
<h1>{{title}}</h1>
<p>{{body}}</p>
```

و template loader صفحه‌ی index.html را نیز باید از وجود آن باخبر کرد:

```
<script type="text/javascript">
    EmberHandlebarsLoader.loadTemplates([
       'posts', 'about', 'application', 'contact', 'email', 'phone',
       'recent-comments', 'post'
    ]);
</script>
```

اکنون به قالب Scripts\Templates\posts.hbs مراجعه کرده و هر عنوان را به مطلب متناظر با آن لینک می‌کنیم:

```
<h2>Emeber.js blog</h2>
<ul>
```

```
    {{#each post in model}}
    <li>{{#link-to 'post' post.id}}{{post.title}}{{/link-to}}</li>
    {{/each}}
</ul>
```

همانطور که ملاحظه می‌کنید، link-to امکان پذیرش id یک مطلب را به صورت متغیر نیز دارا است که سبب خواهد شد تا عناوین، به مطالب متناظر لینک شوند:

همچنین با کلیک بر روی هر عنوان نیز مطلب مرتبط نمایش داده خواهد شد:

افزودن امکان ویرایش مطالب

می‌خواهیم در صفحه‌ی نمایش جزئیات یک مطلب، امکان ویرایش آن‌را نیز فراهم کنیم. بنابراین فایل Scripts\Templates\post.hbs را گشوده و محتوای آن‌را به نحو ذیل ویرایش کنید:

```
<h2>{{title}}</h2>
{{#if isEditing}}
<form>
    <div class="form-group">
        <label for="title">Title</label>
        {{input value=title id="title" class="form-control"}}
    </div>
    <div class="form-group">
        <label for="body">Body</label>
        {{textarea value=body id="body" class="form-control" rows="5"}}
    </div>
    <button class="btn btn-primary" {{action 'save' }}>Save</button>
</form>
{{else}}
<p>{{body}}</p>
<button class="btn btn-primary" {{action 'edit' }}>Edit</button>
{{/if}}
```

شبیه به این if و else را در قسمت قبل حین ایجاد صفحات about و یا contact نیز مشاهده کرده‌اید. در اینجا اگر خاصیت isEditing مساوی true باشد، فرم ویرایش اطلاعات ظاهر می‌شود و اگر خیر، محتوای مطلب جاری نمایش داده خواهد شد.
در فرم تعریف شده، المان‌های ورودی اطلاعات از handlebar helperهای ویژه‌ی input و textarea استفاده می‌کنند؛ بجای المان‌های متداول HTML. همچنین value یکی به title و دیگری به body تنظیم شده‌است (خواص مدل ارائه شده توسط کنترلر متصل به قالب). این مقادیر نیز داخل '' قرار ندارند؛ به عبارتی در یک handlebar helper به عنوان متغیر در نظر گرفته می‌شوند.
به این ترتیب اطلاعات کنترلر جاری، به این المان‌های ورودی اطلاعات به صورت خودکار bind می‌شوند و برعکس. اگر کاربر مقادیر آن‌ها را تغییر دهد، تغییرات نهایی به صورت خودکار به خواص متناظری در کنترلر جاری منعکس خواهند شد (two-way data binding).
دو دکمه نیز تعریف شده‌اند که به اکشن‌های save و edit متصل هستند.
بنابراین نیاز به یک کنترلر جدید، به نام post داریم تا بتوان رفتار قالب post را کنترل کرد. برای این منظور فایل جدید Scripts\Controllers\post.js را با محتوای ذیل ایجاد کنید:

```
Blogger.PostController = Ember.ObjectController.extend({
    isEditing: false,
    actions: {
        edit: function () {
            this.set('isEditing', true);
        },
        save: function () {
            this.set('isEditing', false);
        }
    }
});
```

همچنین مدخل تعریف آن‌را نیز به فایل index.html اضافه نمائید (پس از تعاریف کنترلرهای موجود):

```
<script src="Scripts/Controllers/post.js" type="text/javascript"></script>
```

اگر به کدهای این کنترلر دقت کرده باشید، اینبار زیرکلاسی از ObjectController ایجاد شده‌است و نه Controller، مانند مثال‌های قبل . ObjectController تغییرات رخ داده بر روی خواص مدل را که توسط کنترلر در معرض دید قالب قرار داده‌است، به صورت خودکار به مدل مرتبط نیز منعکس می‌کند (Ember.ObjectController.extend)؛ اما Controller خیر (Ember.Controller.extend). در اینجا مدل کنترلر، تنها «یک» شیء است که بر اساس id آن انتخاب شده‌است. به همین جهت از ObjectController برای ارائه two-way data binding کمک گرفته شد.
در ember.js، یک قالب تنها با کنترلر خودش دارای تبادل اطلاعات است. اگر این کنترلر از نوع ObjectController باشد، تغییرات خاصیتی در یک قالب، ابتدا به کنترلر آن منعکس می‌شود و سپس این کنترلر، در صورت یافتن معادلی از این خاصیت در مدل، آن‌را به روز خواهد کرد. در حالت استفاده از Controller معمولی، صرفا تبادل اطلاعات بین قالب و کنترلر را شاهد خواهیم بود و نه بیشتر.

در ابتدای کار مقدار خاصیت isEditing مساوی false است. این مورد سبب می‌شود تا در بار اول بارگذاری اطلاعات یک مطلب

انتخابی، صرفا عنوان و محتوای مطلب نمایش داده شوند؛ به همراه یک دکمه‌ی edit. با کلیک بر روی دکمه‌ی edit، مطابق کدهای کنترلر فوق، تنها خاصیت isEditing به true تنظیم می‌شود و در این حالت، بدنه‌ی اصلی شرط if isEditing در قالب post، رندر خواهد شد.

برای مثال در ابتدا مطلب شماره یک را انتخاب می‌کنیم:

با کلیک بر روی دکمه‌ی edit، فرم ویرایش ظاهر خواهد شد:

نکته‌ی جالب آن، مقدار دهی خودکار المان‌های ویرایش اطلاعات است. در این حالت سعی کنید، عنوان مطلب جاری را اندکی ویرایش کنید:

با ویرایش عنوان، می‌توان بلافاصله مقدار تغییر یافته را در برچسب عنوان مطلب نیز مشاهده کرد. این مورد دقیقا مفهوم two-way data binding و اتصال مقادیر value هر کدام از handlebar helperهای ویژه‌ی input و textarea را به عناصر مدل ارائه شده توسط کنترلر post، بیان می‌کند.
در این حالت در کدهای متد save، تنها کافی است که خاصیت isEditing را به false تنظیم کنیم. زیرا کلیه مقادیر ویرایش شده توسط کاربر، در همان لحظه در برنامه منتشر شده‌اند و نیاز به کار بیشتری برای اعمال تغییرات نیست.

اضافه کردن دکمه‌ی مرتب سازی بر اساس عناوین، در صفحه‌ی اول سایت

Ember.ObjectController.extend برای data bindg یک شیء کاربرد دارد. اگر قصد داشته باشیم با آرایه‌ای از اشیاء کار کنیم می‌توان از ArrayController استفاده کرد. فرض کنید در صفحه‌ی اول سایت می‌خواهیم امکان مرتب سازی مطالب را بر اساس عنوان آن‌ها اضافه کنیم. فایل Scripts\Templates\posts.hbs را گشوده و لینک Sort by title را به انتهای آن اضافه کنید:

```
<h2>Emeber.js blog</h2>
<ul>
    {{#each post in model}}
    <li>{{#link-to 'post' post.id}}{{post.title}}{{/link-to}}</li>
    {{/each}}
</ul>

<a href="#" class="btn btn-primary" {{action 'sortByTitle'}}>Sort by title</a>
```

در اینجا چون قصد تغییر رفتار قالب posts را توسط اکشن جدید sortByTitle داریم، نیاز است کنترلر متناظر با آن‌را نیز اضافه کنیم. برای این منظور فایل جدید Scripts\Controllers\posts.js را به پوشه‌ی کنترلرها اضافه کنید؛ با محتوای ذیل:

```
Blogger.PostsController = Ember.ArrayController.extend({
    sortProperties: ['id'],// مقادیر پیش فرض مرتب سازی
    sortAscending: false,
    actions: {
        sortByTitle: function () {
            this.set('sortProperties', ['title']);
            this.set('sortAscending', !this.get('sortAscending'));
        }
    }
});
```

sortProperties جزو خواص کلاس پایه ArrayController است. اگر مانند سطر اول به صورت مستقیم مقدار دهی شود، خاصیت یا خواص پیش فرض مرتب سازی را مشخص می‌کند. اگر مانند اکشن sortByTitle توسط متد set مقدار دهی شود، امکان مرتب سازی تعاملی و با فرمان کاربر را فراهم خواهد کرد.

در ادامه، تعریف مدخل این کنترلر جدید را نیز باید به فایل index.html، اضافه کرد:

```
<script src="Scripts/Controllers/posts.js" type="text/javascript"></script>
```

اگر برنامه را در این حالت اجرا کرده و بر روی دکمه‌ی Sort by title کلیک کنید، اتفاقی رخ نمی‌دهد. علت اینجا است که ArrayController خروجی تغییر یافته خودش را توسط خاصیتی به نام arrangedContent در اختیار قالب خود قرار می‌دهد. بنابراین نیاز است فایل قالب Scripts\Templates\posts.hbs را به نحو ذیل ویرایش کرد:

```
<h2>Emeber.js blog</h2>
<ul>
    {{#each post in arrangedContent}}
    <li>{{#link-to 'post' post.id}}{{post.title}}{{/link-to}}</li>
    {{/each}}
</ul>

<a href="#" class="btn btn-primary" {{action 'sortByTitle'}}>Sort by title</a>
```

اینبار کلیک بر روی دکمه‌ی مرتب سازی بر اساس عناوین، هربار لیست موجود را به صورت صعودی و یا نزولی مرتب می‌کند.

یک نکته: حلقه‌ی ویژه‌ای به نام each

اگر قالب Scripts\Templates\posts.hbs را به نحو ذیل، با یک حلقه‌ی each ساده بازنویسی کنید:

```
<h2>Ember.js blog</h2>
<ul>
    {{#each}}
    <li>{{#link-to 'post' id}}{{title}}{{/link-to}}</li>
    {{/each}}
</ul>

<a href="#" class="btn btn-primary" {{action 'sortByTitle'}}>Sort by title</a>
```

هم در حالت نمایش معمولی و هم در حالت استفاده از ArrayController برای نمایش اطلاعات مرتب شده، بدون مشکل کار می‌کند و نیازی به تغییر نخواهد داشت.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS03_03.zip

عنوان: ساخت یک بلاگ ساده با Ember.js، قسمت چهارم
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۹ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, SPA, EmberJS

در قسمت قبل ، اطلاعات نمایش داده شده، از یک سری آرایه ثابت جاوا اسکریپتی تامین شدند. در یک برنامه‌ی واقعی نیاز است داده‌ها را یا از HTML 5 local storage تامین کرد و یا از سرور به کمک Ajax. برای اینگونه اعمال، ember.js به همراه افزونه‌ای است به نام Ember Data که جزئیات کار با آن‌را در این قسمت بررسی خواهیم کرد.

استفاده از Ember Data با Local Storage

برای کار با HTML 5 local storage نیاز به Ember Data Local Storage Adapter نیز هست که در قسمت اول این سری، آدرس دریافت آن معرفی شد. این فایل‌ها نیز در پوشه‌ی Scripts\Libs برنامه کپی خواهند شد.
در ادامه به فایل Scripts\App\store.js که در قسمت قبل جهت تعریف دو آرایه ثابت مطالب و نظرات اضافه شد، مراجعه کرده و محتوای فعلی آن‌را با کدهای زیر جایگزین کنید:

```
Blogger.ApplicationSerializer = DS.LSSerializer.extend();
Blogger.ApplicationAdapter = DS.LSAdapter.extend();
```

این تعاریف سبب خواهند شد تا Ember Data از Local Storage Adapter استفاده کند.
در ادامه با توجه به حذف دو آرایه‌ی posts و comments که پیشتر در فایل store.js تعریف شده بودند، نیاز است مدل‌های متناظری را جهت تعریف خواص آن‌ها، به برنامه اضافه کنیم. این‌کار را با افزودن دو فایل جدید comment.js و post.js به پوشه‌ی Scripts\Models انجام خواهیم داد.
محتوای فایل Scripts\Models\post.js :

```
Blogger.Post = DS.Model.extend({
  title: DS.attr(),
  body: DS.attr()
});
```

محتوای فایل Scripts\Models\comment.js :

```
Blogger.Comment = DS.Model.extend({
  text: DS.attr()
});
```

سپس مداخل تعریف آن‌ها را به فایل index.html نیز اضافه خواهیم کرد:

```
<script src="Scripts/Models/post.js" type="text/javascript"></script>
<script src="Scripts/Models/comment.js" type="text/javascript"></script>
```

برای تعاریف مدل‌ها در Ember data مرسوم است که نام مدل‌ها، اسامی جمع نباشند. سپس با ایجاد وهله‌ای از DS.Model.extend یک مدل ember data را تعریف خواهیم کرد. در این مدل، خواص هر شیء را مشخص کرده و مقدار آن‌ها همیشه DS.attr() خواهد بود. این نکته را در دو مدل Post و Comment مشاهده می‌کنید. اگر دقت کنید به هر دو مدل، خاصیت id اضافه نشده‌است. این خاصیت به صورت خودکار توسط Ember data تنظیم می‌شود.

اکنون نیاز است برنامه را جهت استفاده از این مدل‌های جدید به روز کرد. برای این منظور فایل Scripts\Routes\posts.js را گشوده و مدل آن‌را به نحو ذیل ویرایش کنید:

```
Blogger.PostsRoute = Ember.Route.extend({
    //controllerName: 'posts', // مقدار پیش فرض است و نیازی به ذکر آن نیست
```

```
    //renderTemplare: function () {
    //    this.render('posts'); // مقدار پیش فرض است و نیازی به ذکر آن نیست
    //},
    model: function () {
        return this.store.find('post');
    }
});
```

در اینجا this.store معادل data store برنامه است که مطابق تنظیمات برنامه، همان ember data می‌باشد. سپس متد find را به همراه نام مدل، به صورت رشته‌ای در اینجا مشخص می‌کنیم.

به همین ترتیب فایل Scripts\Routes\recent-comments.js را نیز جهت استفاده از data store ویرایش خواهیم کرد:

```
Blogger.RecentCommentsRoute = Ember.Route.extend({
    model: function () {
        return this.store.find('comment');
    }
});
```

و فایل Scripts\Routes\post.js که در آن منطق یافتن یک مطلب بر اساس آدرس مختص به آن قرار دارد، به صورت ذیل بازنویسی می‌شود:

```
Blogger.PostRoute = Ember.Route.extend({
    model: function (params) {
        return this.store.find('post', params.post_id);
    }
});
```

اگر متد find بدون پارامتر ذکر شود، به معنای بازگشت تمامی عناصر موجود در آن مدل خواهد بود و اگر پارامتر دوم آن مانند این مثال تنظیم شود، تنها همان وهله‌ی درخواستی را بازگشت می‌دهد.

افزودن امکان ثبت یک مطلب جدید

تا اینجا اگر برنامه را اجرا کنید، برنامه بدون خطا بارگذاری خواهد شد اما فعلا رکوردی را برای نمایش ندارد. در ادامه، برنامه را جهت افزودن مطالب جدید توسعه خواهیم داد. برای اینکار ابتدا به فایل Scripts\App\router.js مراجعه کرده و سپس مسیریابی جدید new-post را تعریف خواهیم کرد:

```
Blogger.Router.map(function () {
    this.resource('posts', { path: '/' });
    this.resource('about');
    this.resource('contact', function () {
        this.resource('email');
        this.resource('phone');
    });
    this.resource('recent-comments');
    this.resource('post', { path: 'posts/:post_id' });
    this.resource('new-post');
});
```

اکنون در صفحه‌ی اول سایت، توسط قالب Scripts\Templates\posts.hbs، دکمه‌ای را جهت ایجاد یک مطلب جدید اضافه خواهیم کرد:

```
<h2>Ember.js blog</h2>
<ul>
    {{#each post in arrangedContent}}
    <li>{{#link-to 'post' post.id}}{{post.title}}{{/link-to}}</li>
    {{/each}}
</ul>

<a href="#" class="btn btn-primary" {{action 'sortByTitle' }}>Sort by title</a>
{{#link-to 'new-post' classNames="btn btn-success"}}New Post{{/link-to}}
```

در اینجا دکمه‌ی New Post به مسیریابی جدید new-post اشاره می‌کند.
برای تعریف عناصر نمایشی این مسیریابی، فایل جدید قالب Scripts\Templates\new-post.hbs را با محتوای زیر اضافه کنید:

```
<h1>New post</h1>
<form>
  <div class="form-group">
    <label for="title">Title</label>
    {{input value=title id="title" class="form-control"}}
  </div>

  <div class="form-group">
    <label for="body">Body</label>
    {{textarea value=body id="body" class="form-control" rows="5"}}
  </div>


  <button class="btn btn-primary" {{action 'save'}}>Save</button>
</form>
```

با نمونه‌ی این فرم در قسمت قبل در حین ویرایش یک مطلب، آشنا شدیم. دو المان دریافت اطلاعات در آن قرار دارند که هر کدام به خواص مدل برنامه bind شده‌اند. همچنین یک دکمه‌ی save، با اکشنی به همین نام در اینجا تعریف شده‌است.
پس از آن نیاز است نام فایل قالب new-post را به template loader برنامه در فایل index.html اضافه کرد:

```
<script type="text/javascript">
    EmberHandlebarsLoader.loadTemplates([
       'posts', 'about', 'application', 'contact', 'email', 'phone',
       'recent-comments', 'post', 'new-post'
    ]);
</script>
```

برای مدیریت دکمه‌ی save این قالب جدید نیاز است کنترلر جدیدی را در فایل جدید Scripts\Controllers\new-post.js تعریف کنیم؛ با این محتوا:

```
Blogger.NewPostController = Ember.Controller.extend({
    actions: {
        save: function () {
            var newPost = this.store.createRecord('post', {
                title: this.get('title'),
                body: this.get('body')
            });
            newPost.save();
            this.transitionToRoute('posts');
        }
    }
});
```

به همراه افزودن مدخلی از آن به فایل index.html برنامه:

```
<script src="Scripts/Controllers/new-post.js" type="text/javascript"></script>
```

در اینجا کنترلر جدید NewPostController را مشاهده می‌کنید. از این جهت که برای دسترسی به خواص مدل تغییر کرده، از متد this.get استفاده شده‌است، نیازی نیست حتما از یک ObjectController مانند قسمت قبل استفاده کرد و Controller معمولی نیز برای اینکار کافی است.
آرگومان اول this.store.createRecord نام مدل است و آرگومان دوم آن، وهله‌ای که قرار است به آن اضافه شود. همچنین باید دقت داشت که برای تنظیم یک خاصیت، از متد this.set و برای دریافت مقدار یک خاصیت تغییر کرده از this.get به همراه نام خاصیت مورد نظر استفاده می‌شود و نباید مستقیما برای مثال از this.title استفاده کرد.
this.store.createRecord صرفا یک شیء جدید (ember data object) را ایجاد می‌کند. برای ذخیره سازی نهایی آن باید متد save آن‌را فراخوانی کرد (پیاده سازی الگوی active record است). به این ترتیب این شیء در local storage ذخیره خواهد شد.
پس از ذخیره‌ی مطلب جدید، از متد this.transitionToRoute استفاده شده‌است. این متد، برنامه را به صورت خودکار به

صفحه‌ی متناظر با مسیریابی posts هدایت می‌کند.

اکنون برنامه را اجرا کنید. بر روی دکمه‌ی سبز رنگ new post در صفحه‌ی اول کلیک کرده و یک مطلب جدید را تعریف کنید. بلافاصله عنوان و لینک متناظر با این مطلب را در صفحه‌ی اول سایت مشاهده خواهید کرد.
همچنین اگر برنامه را مجددا بارگذاری کنید، این مطالب هنوز قابل مشاهده هستند؛ زیرا در local storage مرورگر ذخیره شده‌اند.

در اینجا اگر به لینک‌های تولید شده دقت کنید، id آن‌ها عددی نیست. این روشی است که local storage با آن کار می‌کند.

افزودن امکان حذف یک مطلب به سایت

برای حذف یک مطلب، دکمه‌ی حذف را به انتهای قالب Scripts\Templates\post.hbs اضافه خواهیم کرد:

```
<h2>{{title}}</h2>
{{#if isEditing}}
<form>
    <div class="form-group">
        <label for="title">Title</label>
        {{input value=title id="title" class="form-control"}}
    </div>
    <div class="form-group">
        <label for="body">Body</label>
        {{textarea value=body id="body" class="form-control" rows="5"}}
    </div>
    <button class="btn btn-primary" {{action 'save' }}>Save</button>
</form>
{{else}}
<p>{{body}}</p>
<button class="btn btn-primary" {{action 'edit' }}>Edit</button>
<button class="btn btn-danger" {{action 'delete' }}>Delete</button>
{{/if}}
```

سپس کنترلر Scripts\Controllers\post.js را جهت مدیریت اکشن جدید delete به نحو ذیل تکمیل می‌کنیم:

```
Blogger.PostController = Ember.ObjectController.extend({
    isEditing: false,
    actions: {
        edit: function () {
            this.set('isEditing', true);
        },
        save: function () {
            var post = this.get('model');
            post.save();

            this.set('isEditing', false);
        },
        delete: function () {
            if (confirm('Do you want to delete this post?')) {
                this.get('model').destroyRecord();
                this.transitionToRoute('posts');
            }
        }
    }
});
```

متد destroyRecord، مدل انتخابی را هم از حافظه و هم از data store حذف می‌کند. سپس کاربر را به صفحه‌ی اصلی سایت هدایت خواهیم کرد.

متد save نیز در اینجا بهبود یافته‌است. ابتدا مدل جاری دریافت شده و سپس متد save بر روی آن فراخوانی می‌شود. به این ترتیب اطلاعات از حافظه به local storage نیز منتقل خواهند شد.

ثبت و نمایش نظرات به همراه تنظیمات روابط اشیاء در Ember Data

در ادامه قصد داریم امکان افزودن نظرات را به مطالب، به همراه نمایش آن‌ها در ذیل هر مطلب، پیاده سازی کنیم. برای اینکار نیاز است رابطه‌ی بین یک مطلب و نظرات مرتبط با آن‌را در مدل ember data مشخص کنیم. به همین جهت فایل Scripts\Models\post.js را گشوده و تغییرات ذیل را به آن اعمال کنید:

```
Blogger.Post = DS.Model.extend({
  title: DS.attr(),
  body: DS.attr(),
  comments: DS.hasMany('comment', { async: true })
});
```

در اینجا خاصیت جدیدی به نام comments به مدل مطلب اضافه شده‌است و توسط آن می‌توان به تمامی نظرات یک مطلب دسترسی یافت؛ تعریف رابطه‌ی یک به چند، به کمک متد DS.hasMany که پارامتر اول آن نام مدل مرتبط است. تعریف async: true

برای کار با local storage اجباری است و در نگارش‌های آتی ember data حالت پیش فرض خواهد بود.
همچنین نیاز است یک سر دیگر رابطه را نیز مشخص کرد. برای این منظور فایل Scripts\Models\comment.js را گشوده و به نحو ذیل تکمیل کنید:

```
Blogger.Comment = DS.Model.extend({
    text: DS.attr(),
    post: DS.belongsTo('post', { async: true })
});
```

در اینجا خاصیت جدید post به مدل نظر اضافه شده‌است و مقدار آن از طریق متد DS.belongsTo که مدل post را به یک نظر، مرتبط می‌کند، تامین خواهد شد. بنابراین در این حالت اگر به شیء comment مراجعه کنیم، خاصیت جدید post.id آن، به id مطلب متناظر اشاره می‌کند.

در ادامه نیاز است بتوان تعدادی نظر را ثبت کرد. به همین جهت با تعریف مسیریابی آن شروع می‌کنیم. این مسیریابی تعریف شده در فایل Scripts\App\router.js نیز باید تو در تو باشد؛ زیرا قسمت ثبت نظر (new-comment) دقیقا داخل همان صفحه‌ی نمایش یک مطلب ظاهر می‌شود:

```
Blogger.Router.map(function () {
    this.resource('posts', { path: '/' });
    this.resource('about');
    this.resource('contact', function () {
        this.resource('email');
        this.resource('phone');
    });
    this.resource('recent-comments');
    this.resource('post', { path: 'posts/:post_id' }, function () {
        this.resource('new-comment');
    });
    this.resource('new-post');
});
```

لینک آن‌را نیز به انتهای فایل Scripts\Templates\post.hbs اضافه می‌کنیم. از این جهت که این لینک به مدل جاری اشاره می‌کند، با استفاده از متغیر this، مدل جاری را به عنوان مدل مورد استفاده مشخص خواهیم کرد:

```
<h2>{{title}}</h2>
{{#if isEditing}}
<form>
    <div class="form-group">
        <label for="title">Title</label>
        {{input value=title id="title" class="form-control"}}
    </div>
    <div class="form-group">
        <label for="body">Body</label>
        {{textarea value=body id="body" class="form-control" rows="5"}}
    </div>
    <button class="btn btn-primary" {{action 'save' }}>Save</button>
</form>
{{else}}
<p>{{body}}</p>
<button class="btn btn-primary" {{action 'edit' }}>Edit</button>
<button class="btn btn-danger" {{action 'delete' }}>Delete</button>
{{/if}}

<h2>Comments</h2>
{{#each comment in comments}}
<p>
    {{comment.text}}
</p>
{{/each}}

<p>{{#link-to 'new-comment' this class="btn btn-success"}}New comment{{/link-to}}</p>
{{outlet}}
```

پس از تکمیل روابط مدل‌ها، قالب Scripts\Templates\post.hbs را جهت استفاده از این خواص به روز خواهیم کرد. در تغییرات جدید، قسمت <h2>Comments</h2> به انتهای صفحه اضافه شده‌است. سپس حلقه‌ای بر روی خاصیت جدید comments تشکیل

شده و مقدار خاصیت text هر آیتم نمایش داده می‌شود.

در انتهای قالب نیز یک {{outlet}} اضافه شده‌است. کار آن نمایش قالب ارسال یک نظر جدید، پس از کلیک بر روی لینک New Comment می‌باشد. این قالب را با افزودن فایل Scripts\Templates\new-comment.hbs با محتوای ذیل ایجاد خواهیم کرد:

```
<h2>New comment</h2>

<form>
  <div class="form-group">
    <label for="text">Your thoughts:</label>
    {{textarea value=text id="text" class="form-control" rows="5"}}
  </div>

  <button class="btn btn-primary" {{action "save"}}>Add your comment</button>
</form>
```

سپس نام این قالب را به template loader فایل index.html نیز اضافه می‌کنیم؛ تا در ابتدای بارگذاری برنامه شناسایی شده و استفاده شود:

```
<script type="text/javascript">
    EmberHandlebarsLoader.loadTemplates([
       'posts', 'about', 'application', 'contact', 'email', 'phone',
       'recent-comments', 'post', 'new-post', 'new-comment'
    ]);
</script>
```

این قالب به خاصیت text یک comment متصل بوده و همچنین اکشن جدیدی به نام save دارد. بنابراین برای مدیریت اکشن save، نیاز به کنترلری متناظر خواهد بود. به همین جهت فایل جدید Scripts\Controllers\new-comment.js را با محتوای ذیل ایجاد کنید:

```
Blogger.NewCommentController = Ember.ObjectController.extend({
    needs: ['post'],
    actions: {
        save: function () {
            var comment = this.store.createRecord('comment', {
                text: this.get('text')
            });
            comment.save();

            var post = this.get('controllers.post.model');
            post.get('comments').pushObject(comment);
            post.save();

            this.transitionToRoute('post', post.id);
        }
    }
});
```

و مدخل تعریف آن‌را نیز به صفحه‌ی index.html اضافه می‌کنیم:

```
<script src="Scripts/Controllers/new-comment.js" type="text/javascript"></script>
```

قسمت ذخیره سازی comment جدید با ذخیره سازی یک post جدید که پیشتر بررسی کردیم، تفاوتی ندارد. از متد this.store.createRecord جهت معرفی وهله‌ای جدید از comment استفاده و سپس متد save آن، برای ثبت نهایی فراخوانی شده‌است.

در ادامه باید این نظر جدید را به post متناظر با آن مرتبط کنیم. برای اینکار نیاز است تا به مدل کنترلر post دسترسی داشته باشیم. به همین جهت خاصیت needs را به تعاریف کنترلر جاری به همراه نام کنترلر مورد نیاز، اضافه کرده‌ایم. به این ترتیب می‌توان توسط متد this.get و پارامتر controllers.post.model در کنترلر NewComment به اطلاعات کنترلر post دسترسی یافت. سپس خاصیت comments شیء post جاری را یافته و مقدار آن‌را به comment جدیدی که ثبت کردیم، تنظیم می‌کنیم. در ادامه با فراخوانی متد save، کار تنظیم ارتباطات یک مطلب و نظرهای جدید آن به پایان می‌رسد.

در آخر با فراخوانی متد transitionToRoute به مطلبی که نظر جدیدی برای آن ارسال شده‌است باز می‌گردیم.

همانطور که در این تصویر نیز مشاهده می‌کنید، اطلاعات ذخیره شده در local storage را توسط افزونه‌ی Ember Inspector نیز می‌توان مشاهده کرد.

افزودن دکمه‌ی حذف به لیست نظرات ارسالی

برای افزودن دکمه‌ی حذف، به قالب Scripts\Templates\post.hbs مراجعه کرده و قسمتی را که لیست نظرات را نمایش می‌دهد، به نحو ذیل تکمیل می‌کنیم:

```
{{#each comment in comments}}
```

```
<p>
    {{comment.text}}
    <button class="btn btn-xs btn-danger" {{action 'delete' }}>delete</button>
</p>
{{/each}}
```

همچنین برای مدیریت اکشن جدید delete، کنترلر جدید comment را در فایل Scripts\Controllers\comment.js اضافه خواهیم کرد.

```
Blogger.CommentController = Ember.ObjectController.extend({
    needs: ['post'],
    actions: {
        delete: function () {
            if (confirm('Do you want to delete this comment?')) {
                var comment = this.get('model');
                comment.deleteRecord();
                comment.save();

                var post = this.get('controllers.post.model');
                post.get('comments').removeObject(comment);
                post.save();
            }
        }
    }
});
```

به همراه تعریف مدخل آن در فایل index.html :

```
<script src="Scripts/Controllers/comment.js" type="text/javascript"></script>
```

در این حالت اگر برنامه را اجرا کنید، پیام «Do you want to delete this **post**» را مشاهده خواهید کرد بجای پیام «Do you want to delete this **comment**». علت اینجا است که قالب post به صورت پیش فرض به کنترلر post متصل است و نه کنترلر comment. برای رفع این مشکل تنها کافی است از itemController به نحو ذیل استفاده کنیم:

```
{{#each comment in comments  itemController="comment"}}
<p>
    {{comment.text}}
    <button class="btn btn-xs btn-danger" {{action 'delete' }}>delete</button>
</p>
{{/each}}
```

به این ترتیب اکشن delete به کنترلر comment ارسال خواهد شد و نه کنترلر پیش فرض post جاری.
در کنترلر Comment روش دیگری را برای حذف یک رکورد مشاهده می‌کنید. می‌توان ابتدا متد deleteRecord را بر روی مدل فراخوانی کرد و سپس آن‌را save نمود تا نهایی شود. همچنین در اینجا نیاز است نظر حذف شده را از سر دیگر رابطه نیز حذف کرد. روش دسترسی به post جاری در این حالت، همانند توضیحات NewCommentController است که پیشتر بحث شد.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS03_04.zip

عنوان: **ساخت یک بلاگ ساده با Ember.js، قسمت پنجم**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۰ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.NET Web API, jQuery, SPA, EmberJS

مقدمات ساخت بلاگ مبتنی بر ember.js در قسمت قبل به پایان رسید. در این قسمت صرفا قصد داریم بجای استفاده از HTML 5 local storage از یک REST web service مانند یک ASP.NET Web API Controller و یا یک ASP.NET MVC Controller استفاده کنیم و اطلاعات نهایی را به سرور ارسال و یا از آن دریافت کنیم.

تنظیم Ember data برای کار با سرور

Ember data به صورت پیش فرض و در پشت صحنه با استفاده از Ajax برای کار با یک REST Web Service طراحی شده‌است و کلیه تبادلات آن نیز با فرمت JSON انجام می‌شود. بنابراین تمام کدهای سمت کاربر قسمت قبل نیز در این حالت کار خواهند کرد. تنها کاری که باید انجام شود، حذف تنظیمات ابتدایی آن برای کار با HTML 5 local storage است.
برای این منظور ابتدا فایل index.html را گشوده و سپس مدخل localstorage_adapter.js را از آن حذف کنید:

```
<!--<script src="Scripts/Libs/localstorage_adapter.js" type="text/javascript"></script>-->
```

همچنین دیگر نیازی به store.js نیز نمی‌باشد:

```
<!--<script src="Scripts/App/store.js" type="text/javascript"></script>-->
```

اکنون برنامه را اجرا کنید، چنین پیام خطایی را مشاهده خواهید کرد:

همانطور که عنوان شد، ember data به صورت پیش فرض با سرور کار می‌کند و در اینجا به صورت خودکار، یک درخواست Get را به آدرس http://localhost:25918/posts جهت دریافت آخرین مطالب ثبت شده، ارسال کرده‌است و چون هنوز وب سرویسی در برنامه تعریف نشده، با خطای 404 و یا یافت نشد، مواجه شده‌است.
این درخواست نیز بر اساس تعاریف موجود در فایل Scripts\Routes\posts.js، به سرور ارسال شده‌است:

```
Blogger.PostsRoute = Ember.Route.extend({
    model: function () {
        return this.store.find('post');
    }
```

```
});
```

Ember data شبیه به یک ORM عمل می‌کند. تنظیمات ابتدایی آن‌را تغییر دهید، بدون نیازی به تغییر در کدهای اصلی برنامه، می‌تواند با یک منبع داده جدید کار کند.

تغییر تنظیمات پیش فرض آغازین Ember data

آدرس درخواستی http://localhost:25918/posts به این معنا است که کلیه درخواست‌ها، به همان آدرس و پورت ریشه‌ی اصلی سایت ارسال می‌شوند. اما اگر یک ASP.NET Web API Controller را تعریف کنیم، نیاز است این درخواست‌ها، برای مثال به آدرس api/posts ارسال شوند؛ بجای /posts.
برای این منظور پوشه‌ی جدید Scripts\Adapters را ایجاد کرده و فایل web_api_adapter.js را با این محتوا به آن اضافه کنید:

```
DS.RESTAdapter.reopen({
      namespace: 'api'
});
```

سپس تعریف مدخل آن‌را نیز به فایل index.html اضافه نمائید:

```
<script src="Scripts/Adapters/web_api_adapter.js" type="text/javascript"></script>
```

تعریف فضای نام در اینجا سبب خواهد شد تا درخواست‌های جدید به آدرس api/posts ارسال شوند.

تغییر تنظیمات پیش فرض ASP.NET Web API

در سمت سرور، بنابر اصول نامگذاری خواص، نام‌ها با حروف بزرگ شروع می‌شوند:

```
namespace EmberJS03.Models
{
    public class Post
    {
        public int Id { set; get; }
        public string Title { set; get; }
        public string Body { set; get; }
    }
}
```

اما در سمت کاربر و کدهای اسکریپتی، عکس آن صادق است. به همین جهت نیاز است که
CamelCasePropertyNamesContractResolver را در JSON.NET تنظیم کرد تا به صورت خودکار اطلاعات ارسالی به کلاینت‌ها را به صورت camel case تولید کند:

```
using System;
using System.Web.Http;
using System.Web.Routing;
using Newtonsoft.Json.Serialization;

namespace EmberJS03
{
    public class Global : System.Web.HttpApplication
    {

        protected void Application_Start(object sender, EventArgs e)
        {
            RouteTable.Routes.MapHttpRoute(
               name: "DefaultApi",
               routeTemplate: "api/{controller}/{id}",
               defaults: new { id = RouteParameter.Optional }
               );

            var settings =
GlobalConfiguration.Configuration.Formatters.JsonFormatter.SerializerSettings;
            settings.ContractResolver = new CamelCasePropertyNamesContractResolver();
```

```
        }
    }
}
```

نحوه‌ی صحیح بازگشت اطلاعات از یک ASP.NET Web API جهت استفاده در Ember data

با تنظیمات فوق، اگر کنترلر جدیدی را به صورت ذیل جهت بازگشت لیست مطالب تهیه کنیم:

```
namespace EmberJS03.Controllers
{
    public class PostsController : ApiController
    {
        public IEnumerable<Post> Get()
        {
            return DataSource.PostsList;
        }
    }
}
```

با یک چنین خطایی در سمت کاربر مواجه خواهیم شد:

```
WARNING: Encountered "0" in payload, but no model was found for model name "0" (resolved model name
using DS.RESTSerializer.typeForRoot("0"))
```

این خطا از آنجا ناشی می‌شود که Ember data، اطلاعات دریافتی از سرور را بر اساس قرارداد JSON API دریافت می‌کند. برای حل این مشکل راه‌حل‌های زیادی مطرح شده‌اند که تعدادی از آن‌ها را در لینک‌های زیر می‌توانید مطالعه کنید:

http://jsonapi.codeplex.com

https://github.com/xqiu/MVCSPAWithEmberjs

https://github.com/rmichela/EmberDataAdapter

https://github.com/MilkyWayJoe/Ember-WebAPI-Adapter

http://blog.yodersolutions.com/using-ember-data-with-asp-net-web-api

http://emadibrahim.com/2014/04/09/emberjs-and-asp-net-web-api-and-json-serialization

و خلاصه‌ی آن‌ها به این صورت است:

خروجی JSON تولیدی توسط ASP.NET Web API چنین شکلی را دارد:

```
[
  {
    Id: 1,
    Title: 'First Post'
  }, {
    Id: 2,
    Title: 'Second Post'
  }
]
```

اما Ember data نیاز به یک چنین خروجی دارد:

```
{
  posts: [{
    id: 1,
    title: 'First Post'
  }, {
    id: 2,
    title: 'Second Post'
  }]
}
```

به عبارتی آرایه‌ی مطالب را از ریشه‌ی posts باید دریافت کند (مطابق فرمت JSON API). برای انجام اینکار یا از لینک‌های معرفی شده استفاده کنید و یا راه حل ساده‌ی ذیل هم پاسخگو است:

```
using System.Web.Http;
using EmberJS03.Models;

namespace EmberJS03.Controllers
{
    public class PostsController : ApiController
    {
        public object Get()
        {
            return new { posts = DataSource.PostsList };
        }
    }
}
```

در اینجا ریشه‌ی posts را توسط یک anonymous object ایجاد کرده‌ایم.
اکنون اگر برنامه را اجرا کنید، در صفحه‌ی اول آن، لیست عناوین مطالب را مشاهده خواهید کرد.

تاثیر قرارداد JSON API در حین ارسال اطلاعات به سرور توسط Ember data

در تکمیل کنترلرهای Web API مورد نیاز (کنترلرهای مطالب و نظرات)، نیاز به متدهای Post، Update و Delete هم خواهد بود.
دقیقا فرامین ارسالی توسط Ember data توسط همین HTTP Verbs به سمت سرور ارسال می‌شوند. در این حالت اگر متد Post کنترلر نظرات را به این شکل طراحی کنیم:

```
public HttpResponseMessage Post(Comment comment)
```

کار نخواهد کرد؛ چون مطابق فرمت JSON API ارسالی توسط Ember data، یک چنین شیء JSON ایی را دریافت خواهیم کرد:

```
{"comment":{"text":"data...","post":"3"}}
```

بنابراین Ember data چه در حین دریافت اطلاعات از سرور و چه در زمان ارسال اطلاعات به آن، اشیاء جاوا اسکریپتی را در یک ریشه‌ی هم نام آن شیء قرار می‌دهد.
برای پردازش آن، یا باید از راه حل‌های ثالث مطرح شده در ابتدای بحث استفاده کنید و یا می‌توان مطابق کدهای ذیل، کل اطلاعات JSON ارسالی را توسط کتابخانه‌ی JSON.NET نیز پردازش کرد:

```
namespace EmberJS03.Controllers
{
    public class CommentsController : ApiController
    {
        public HttpResponseMessage Post(HttpRequestMessage requestMessage)
        {
            var jsonContent = requestMessage.Content.ReadAsStringAsync().Result;
            // {"comment":{"text":"data...","post":"3"}}
            var jObj = JObject.Parse(jsonContent);
            var comment = jObj.SelectToken("comment", false).ToObject<Comment>();


            var id = 1;
            var lastItem = DataSource.CommentsList.LastOrDefault();
            if (lastItem != null)
            {
                id = lastItem.Id + 1;
            }
            comment.Id = id;
            DataSource.CommentsList.Add(comment);

            // ارسال آی دی با فرمت خاص مهم است
            return Request.CreateResponse(HttpStatusCode.Created, new { comment = comment });
        }
    }
```

```
}
```

در اینجا توسط requestMessage به محتوای ارسال شده‌ی به سرور که همان شیء JSON ارسالی است، دسترسی خواهیم داشت. سپس متد JObject.Parse، آن‌را به صورت عمومی تبدیل به یک شیء JSON می‌کند و نهایتا با استفاده از متد SelectToken آن می‌توان ریشه‌ی comment و یا در کنترلر مطالب، ریشه‌ی post را انتخاب و سپس تبدیل به شیء Comment و یا Post کرد.
همچنین فرمت return نهایی هم مهم است. در این حالت خروجی ارسالی به سمت کاربر، باید مجددا با فرمت JSON API باشد؛ یعنی باید comment اصلاح شده را به همراه ریشه‌ی comment ارسال کرد. در اینجا نیز anonymous object تهیه شده، چنین کاری را انجام می‌دهد.

Lazy loading در Ember data

تا اینجا اگر برنامه را اجرا کنید، لیست مطالب صفحه‌ی اول را مشاهده خواهید کرد، اما لیست نظرات آن‌ها را خیر؛ از این جهت که ضرورتی نداشت تا در بار اول ارسال لیست مطالب به سمت کاربر، تمام نظرات متناظر با آن‌ها را هم ارسال کرد. بهتر است زمانیکه کاربر یک مطلب خاص را مشاهده می‌کند، نظرات خاص آن‌را به سمت کاربر ارسال کنیم.
در تعاریف سمت کاربر Ember data، پارامتر دوم رابطه‌ی hasMany که با async:true مشخص شده‌است، دقیقا معنای lazy loading را دارد.

```
Blogger.Post = DS.Model.extend({
   title: DS.attr(),
   body: DS.attr(),
   comments: DS.hasMany('comment', { async: true } /* lazy loading */)
});
```

در سمت سرور، دو راه برای فعال سازی این lazy loading تعریف شده در سمت کاربر وجود دارد:
الف) Idهای نظرات هر مطلب را به صورت یک آرایه، در بار اول ارسال لیست نظرات به سمت کاربر، تهیه و ارسال کنیم:

```
namespace EmberJS03.Models
{
    public class Post
    {
        public int Id { set; get; }
        public string Title { set; get; }
        public string Body { set; get; }

        // lazy loading via an array of IDs
        public int[] Comments { set; get; }
    }
}
```

در اینجا خاصیت Comments، تنها کافی است لیستی از Idهای نظرات مرتبط با مطلب جاری باشد. در این حالت در سمت کاربر اگر مطلب خاصی جهت مشاهده‌ی جزئیات آن انتخاب شود، به ازای هر Id ذکر شده، یکبار دستور Get صادر خواهد شد.
ب) این روش به علت تعداد رفت و برگشت بیش از حد به سرور، کارآیی آنچنانی ندارد. بهتر است جهت مشاهده‌ی جزئیات یک مطلب، تنها یکبار درخواست Get کلیه نظرات آن صادر شود.
برای اینکار باید مدل برنامه را به شکل زیر تغییر دهیم:

```
namespace EmberJS03.Models
{
    public class Post
    {
        public int Id { set; get; }
        public string Title { set; get; }
        public string Body { set; get; }

        // load related models via URLs instead of an array of IDs
        // ref. https://github.com/emberjs/data/pull/1371
        public object Links { set; get; }

        public Post()
        {
```

```
            Links = new { comments = "comments" }; // api/posts/id/comments
        }
    }
}
```

در اینجا یک خاصیت جدید به نام Links ارائه شده‌است. نام Links در Ember data استاندارد است و از آن برای دریافت کلیه اطلاعات لینک شده‌ی به یک مطلب استفاده می‌شود. با تعریف این خاصیت به نحوی که ملاحظه می‌کنید، اینبار Ember data تنها یکبار درخواست ویژه‌ای را با فرمت api/posts/id/comments، به سمت سرور ارسال می‌کند. برای مدیریت آن، قالب مسیریابی پیش فرض api/{controller}/{id} را می‌توان به صورت api/{controller}/{id}/{name} اصلاح کرد:

```
namespace EmberJS03
{
    public class Global : System.Web.HttpApplication
    {

        protected void Application_Start(object sender, EventArgs e)
        {
            RouteTable.Routes.MapHttpRoute(
               name: "DefaultApi",
               routeTemplate: "api/{controller}/{id}/{name}",
               defaults: new { id = RouteParameter.Optional, name = RouteParameter.Optional }
               );

            var settings =
GlobalConfiguration.Configuration.Formatters.JsonFormatter.SerializerSettings;
            settings.ContractResolver = new CamelCasePropertyNamesContractResolver();
        }
    }
}
```

اکنون دیگر درخواست جدید api/posts/3/comments با پیام 404 یا یافت نشد مواجه نمی‌شود.
در این حالت در طی یک درخواست می‌توان کلیه نظرات را به سمت کاربر ارسال کرد. در اینجا نیز ذکر ریشه‌ی comments همانند ریشه posts، الزامی است:

```
namespace EmberJS03.Controllers
{
    public class PostsController : ApiController
    {
        //GET api/posts/id
        public object Get(int id)
        {
            return
                new
                {
                    posts = DataSource.PostsList.FirstOrDefault(post => post.Id == id),
                    comments = DataSource.CommentsList.Where(comment => comment.Post == id).ToList()
                };
        }
    }
}
```

پردازش‌های async و متد transitionToRoute در Ember.js

اگر متد حذف مطالب را نیز به کنترلر Posts اضافه کنیم:

```
namespace EmberJS03.Controllers
{
    public class PostsController : ApiController
    {
        public HttpResponseMessage Delete(int id)
        {
            var item = DataSource.PostsList.FirstOrDefault(x => x.Id == id);
            if (item == null)
                return Request.CreateResponse(HttpStatusCode.NotFound);
```

```
            DataSource.PostsList.Remove(item);

            //حذف کامنت‌های مرتبط
            var relatedComments = DataSource.CommentsList.Where(comment => comment.Post ==
id).ToList();
            relatedComments.ForEach(comment => DataSource.CommentsList.Remove(comment));

            return Request.CreateResponse(HttpStatusCode.OK, new { post = item });
        }
    }
}
```

قسمت سمت سرور کار تکمیل شده‌است. اما در سمت کاربر، چنین خطایی را دریافت خواهیم کرد:

```
Attempted to handle event `pushedData` on  while in state root.deleted.inFlight.
```

منظور از حالت inFlight در اینجا این است که هنوز کار حذف سمت سرور تمام نشده‌است که متد transitionToRoute را صادر کرده‌اید. برای اصلاح آن، فایل Scripts\Controllers\post.js را باز کرده و پس از متد destroyRecord، متد then را قرار دهید:

```
Blogger.PostController = Ember.ObjectController.extend({
    isEditing: false,
    actions: {
        edit: function () {
            this.set('isEditing', true);
        },
        save: function () {
            var post = this.get('model');
            post.save();

            this.set('isEditing', false);
        },
        delete: function () {
            if (confirm('Do you want to delete this post?')) {
                var thisController = this;
                var post = this.get('model');
                post.destroyRecord().then(function () {
                    thisController.transitionToRoute('posts');
                });
            }
        }
    }
});
```

به این ترتیب پس از پایان عملیات حذف سمت سرور، قسمت then اجرا خواهد شد . همچنین باید دقت داشت که this اشاره کننده به کنترلر جاری را باید پیش از فراخوانی then ذخیره و استفاده کرد.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS03_05.zip

عنوان: **استفاده از ویجت آپلود KendoUI بصورت پاپ آپ**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۳/۱۰/۲۴ ۱۱:۳۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, Kendo UI

پیشنیاز:

- بررسی ویجت Kendo UI File Upload

در مطلب قبل جزئیات استفاده از ویجت آپلود فریمورک قدرتمند Kendo UI عنوان شدند. در این مطلب قصد داریم طریقه‌ی استفاده از آن را به صورت پاپ آپ، در ویجت گرید Kendo بررسی کنیم.

مدل زیر را در نظر بگیرید:

```
var product = {
    ProductId: 1001,
    ProductName: "Product 1001",
    Available: true,
    Filename: "Image02.png"
};
```

و برای ایجاد یک دیتاسورس سمت کاربر برای گرید، یک منبع داده را با استفاده از مدل بالا تولید میکنیم:

```
var productCount = 100;
        var products = [];

        function datasourceFilling() {
            for (var i = 0; i < productCount; i++) {
                var product = {
                    ProductId: i,
                    ProductName: "Product " + i + " Name",
                    Available: i % 2 == 0 ? true: false,
                    Filename: i % 2 == 0 ? "Image01.png" : "Image02.png"
                };
                products.push(product);
            }
        }
```

سپس برای نمایش در گرید، سیم پیچی‌های لازم را انجام میدهیم:

```
function makekendoGrid() {

            $("#grid").kendoGrid({
                dataSource: {
                    data: products,
                    schema: {
                        model:
                        {
                            //id: "ProductId",
                            fields:
                            {
                                ProductId: { editable: false, nullable: true },
                                ProductName: { validation: { required: true } },
                                Available: { type: "boolean" },
                                ImageName: { type: "string", editable: false },
                                Filename: { type: "string", validation: { required: true } }
                            }
                        }
                    },
                    pageSize: 20
                },
                height: 550,
                scrollable: true,
                sortable: true,
                filterable: true,
                pageable: {
                    input: true,
                    numeric: false
                },
                editable: {
```

```
                    mode: "popup",
                },
                columns: [
                    { field: "ProductName", title: "Product Name" },
                    { field: "Available", width: "100px", template: '<input type="checkbox" #=
Available ? checked="checked" : "" # disabled="disabled" ></input>' },
                    { field: "ImageName", width: "150px", template: "<img src='/img/#=Filename#'
alt='#=Filename #' Title='#=Filename #' height='80' width='80'/>" },
                    { field: "Filename", width: "100px", editor: fileEditor },
                    { command: ["edit"], title: " ", width: "200px" }
                ]
            });

            var grid = $('#grid').data('kendoGrid');
            grid.hideColumn(3);
        }
```

همانطور که میبینید در ستون‌های گرید، یک ستون ImageName نیز اضافه شده‌است که به‌صورت تمپلیت تصویر محصول را نمایش میدهد. برای خود فیلد نیز از یک تمپلیت ادیتور که در ادامه تعریف خواهیم کرد استفاده کرده‌ایم:

```
function fileEditor(container, options) {

            $('<input type="file" name="file"/>')
                .appendTo(container)
                .kendoUpload({
                    multiple: false,
                    async: {
                        saveUrl: "@Url.Action("Save", "Home")",
                        removeUrl: "@Url.Action("Remove", "Home")",
                        autoUpload: true,
                    },
                    upload: function (e) {
                        alert("upload");
                        e.data = { Id: options.model.Id };
                    },
                    success: function (e) {
                        alert("success");
                        options.model.set("ImageName", e.response.ImageUrl);
                    },
                    error: function (e) {
                        alert("error");
                        alert("Failed to upload " + e.files.length + " files " +
e.XMLHttpRequest.status + " " + e.XMLHttpRequest.responseText);
                    }
                });
        }
```

برای آپلود فایل و حذف آن نیز اکشن‌های لازم را در سمت سرور مینویسم:

```
[System.Web.Mvc.HttpPost]
        public virtual ActionResult Save(HttpPostedFileBase file)
        {
            var exName = Path.GetExtension(file.FileName);
            var totalFileName = System.Guid.NewGuid().ToString().ToLower().Replace("-", "") + exName;
            var physicalPath = Path.Combine(Server.MapPath("/img"), totalFileName);
            file.SaveAs(physicalPath);
            return Json(new { ImageUrl = totalFileName }, "text/plain");
        }

        [System.Web.Mvc.HttpPost]
        public virtual ContentResult Remove(string fileName)
        {
            if (fileName != null)
            {
                var physicalPath = Path.Combine(Server.MapPath("/img"), fileName);
                System.IO.File.Delete(physicalPath);
            }

            // Return an empty string to signify success
            return Content("");
        }
```

حاصل کار بصورت تصویر زیر نمایش داده شده است:

Edit
Product Name
Product 4 Name
Available
01
ImageName
Select...
Filename
Update
Cancel