عنوان: **صفحه بندی و مرتب سازی خودکار اطلاعات به کمک jqGrid در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۲:۴۵
آدرس: www.dotnettips.info
گروه‌ها: MVC, jqGrid

jqGrid یکی از افزونه‌های بسیار محبوب jQuery جهت نمایش جدول مانند اطلاعات، در سمت کلاینت است. توانمندی‌های آن صرفا به نمایش ستون‌ها و ردیف‌ها خلاصه نمی‌شود. قابلیت‌هایی مانند صفحه بندی، مرتب سازی، جستجو، ویرایش توکار، تولید خودکار صفحات افزودن رکوردها، اعتبارسنجی داده‌ها، گروه بندی، نمایش درختی و غیره را نیز به همراه دارد. همچنین به صورت توکار پشتیبانی از راست به چپ را نیز لحاظ کرده‌است.
مجوز استفاده از فایل‌های جاوا اسکریپتی آن MIT است؛ به این معنا که در هر نوع پروژه‌ای قابل استفاده است. مجوز استفاده از کامپوننت‌های سمت سرور آن که برای نمونه جهت ASP.NET MVC یک سری HTML Helper را تدارک دیده‌اند، تجاری می‌باشد. در ادامه قصد داریم صرفا از فایل‌های JS عمومی آن استفاده کنیم.

دریافت jqGrid

برای دریافت jqGrid می‌توانید به مخزن کد آن، در آدرس https://github.com/tonytomov/jqGrid/releases و یا از طریق NuGet اقدام کنید:

```
PM> Install-Package Trirand.jqGrid
```

استفاده از NuGet بیشتر توصیه می‌شود، زیرا به صورت خودکار وابستگی‌های jQuery و همچنین jQuery UI آن‌را نیز به همراه داشته و نصب خواهد کرد.
از jQuery UI برای تولید صفحات جستجوی بر روی رکوردها و همچنین تولید خودکار صفحات ویرایش و یا افزودن رکوردها استفاده می‌کند. به علاوه آیکن‌ها، قالب و رنگ خود را نیز از jQuery UI دریافت می‌کند. بنابراین اگر قصد تغییر قالب آن‌را داشتید تنها کافی است یک قالب استاندارد دیگر jQuery UI را مورد استفاده قرار دهید.

تنظیمات اولیه فایل Layout سایت

پس از دریافت بسته‌ی نیوگت jqGrid، نیاز است فایل‌های مورد نیاز اصلی آن‌را به شکل زیر به فایل layout پروژه اضافه کرد:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - My ASP.NET Application</title>

    <link href="~/Content/themes/base/jquery.ui.all.css" rel="stylesheet" />
    <link href="~/Content/jquery.jqGrid/ui.jqgrid.css" rel="stylesheet" />
    <link href="~/Content/Site.css" rel="stylesheet" type="text/css" />
</head>
<body>
    <div>
        @RenderBody()
    </div>

    <script src="~/Scripts/jquery-1.7.2.min.js"></script>
    <script src="~/Scripts/jquery-ui-1.8.11.min.js"></script>
    <script src="~/Scripts/i18n/grid.locale-fa.js"></script>
    <script src="~/Scripts/jquery.jqGrid.min.js"></script>

    @RenderSection("Scripts", required: false)
</body>
</html>
```

فایل jquery.ui.all.css شامل تمامی فایل‌های CSS مرتبط با jQuery UI است و نیازی نیست تا سایر فایل‌های آن‌را لحاظ کرد.
این گرید به همراه فایل زبان فارسی grid.locale-fa.js نیز می‌باشد که در کدهای فوق پیوست شده‌است. البته اگر فرصت

کردید نیاز است کمی ترجمه‌های آن بهبود پیدا کنند.

تنظیمات ثانویه site.css

```
.ui-widget {
    font-family: Tahoma !important;
    font-size: 9pt !important;
}

/*how to move jQuery dialog close (X) button from right to left*/
.ui-jqgrid .ui-jqgrid-caption-rtl {
    text-align: center !important;
}

.ui-dialog .ui-dialog-titlebar-close {
    left: .3em !important;
}

.ui-dialog .ui-dialog-title {
    margin: .1em 0 .1em .8em !important;
    direction: rtl !important;
    float: right !important;
}
```

احتمالا تنظیمات قلم‌های jQuery UI و یا jqGrid مدنظر شما نیستند و نیاز به تعویض دارند. در اینجا نحوه‌ی بازنویسی آن‌ها را ملاحظه می‌کنید.
همچنین محل قرار گیری دکمه‌ی بسته شدن دیالوگ‌ها و راست به چپ کردن عناوین آن‌ها نیز در اینجا قید شده‌اند.

مدل برنامه

در ادامه قصد داریم لیستی از محصولات را با ساختار ذیل، توسط jqGrid نمایش دهیم:

```
namespace jqGrid01.Models
{
    public class Product
    {
        public int Id { set; get; }
        public string Name { set; get; }
        public decimal Price { set; get; }
        public bool IsAvailable { set; get; }
    }
}
```

ساختار داده‌ای مورد نیاز توسط jqGrid

jqGrid مستقل است از فناوری سمت سرور. بنابراین هر چند در عنوان بحث ASP.NET MVC ذکر شده‌است، اما از ASP.NET MVC صرفا جهت بازگرداندن خروجی JSON استفاده خواهیم کرد و این مورد در هر فناوری سمت سرور دیگری نیز می‌تواند انجام شود.

```
using System.Collections.Generic;

namespace jqGrid01.Models
{
    public class JqGridData
    {
        public int Total { get; set; }

        public int Page { get; set; }

        public int Records { get; set; }

        public IList<JqGridRowData> Rows { get; set; }

        public object UserData { get; set; }
```

```
    }

    public class JqGridRowData
    {
        public int Id { set; get; }
        public IList<string> RowCells { set; get; }
    }
}
```

خروجی JSON مدنظر توسط jqGrid، یک چنین ساختاری را باید داشته باشد.
Total، نمایانگر تعداد صفحات اطلاعات است. عدد Page، شماره صفحه‌ی جاری است. عدد Records، تعداد کل رکوردهای گزارش را مشخص می‌کند. ساختار ردیف‌های آن نیز تشکیل شده‌است از یک Id به همراه سلول‌هایی که باید با فرمت string، بازگشت داده شوند.
UserData اختیاری است. برای مثال اگر خواستید جمع کل صفحه را در ذیل گرید نمایش دهید، می‌توانید یک anonymous object را در اینجا مقدار دهی کنید. خاصیت‌های آن دقیقا باید با نام خاصیت‌های ستون‌های متناظر، یکی باشند. برای مثال اگر می‌خواهید عددی را در ستون Id، در فوتر گرید نمایش دهید، باید نام خاصیت را Id ذکر کنید.

کدهای سمت کلاینت گرید

در اینجا کدهای کامل سمت کلاینت گرید را ملاحظه می‌کنید:

```
@{
    ViewBag.Title = "Index";
}

<div dir="rtl" align="center">
    <div id="rsperror"></div>
    <table id="list" cellpadding="0" cellspacing="0"></table>
    <div id="pager" style="text-align:center;"></div>
</div>

@section Scripts
{
    <script type="text/javascript">
        $(document).ready(function () {
            $('#list').jqGrid({
                caption: "آزمایش اول",
                //url from wich data should be requested
                url: '@Url.Action("GetProducts","Home")',
                //type of data
                datatype: 'json',
                jsonReader: {
                    root: "Rows",
                    page: "Page",
                    total: "Total",
                    records: "Records",
                    repeatitems: true,
                    userdata: "UserData",
                    id: "Id",
                    cell: "RowCells"
                },
                //url access method type
                mtype: 'GET',
                //columns names
                colNames: ['شماره', 'نام محصول', 'موجود است', 'قیمت'],
                //columns model
                colModel: [
                { name: 'Id', index: 'Id', align: 'right', width: 50, sorttype: "number" },
                { name: 'Name', index: 'Name', align: 'right', width: 300 },
                { name: 'IsAvailable', index: 'IsAvailable', align: 'center', width: 100, formatter:
'checkbox' },
                { name: 'Price', index: 'Price', align: 'center', width: 100, sorttype: "number" }
                ],
                //pager for grid
                pager: $('#pager'),
                //number of rows per page
                rowNum: 10,
                rowList: [10, 20, 50, 100],
                //initial sorting column
                sortname: 'Id',
```

```
                //initial sorting direction
                sortorder: 'asc',
                //we want to display total records count
                viewrecords: true,
                altRows: true,
                shrinkToFit: true,
                width: 'auto',
                height: 'auto',
                hidegrid: false,
                direction: "rtl",
                gridview: true,
                rownumbers: true,
                footerrow: true,
                userDataOnFooter: true,
                loadComplete: function() {
                    //change alternate rows color
                    $("tr.jqgrow:odd").css("background", "#E0E0E0");
                },
                loadError: function(xhr, st, err) {
                     jQuery("#rsperror").html("Type: " + st + "; Response: " + xhr.status + " " +
xhr.statusText);
                }
                //, loadonce: true
            })
            .jqGrid('navGrid', "#pager",
            {
                edit: false, add: false, del: false, search: false,
                refresh: true
            })
            .jqGrid('navButtonAdd', '#pager',
            {
                caption: "تنظیم نمایش ستون‌ها", title: "Reorder Columns",
                onClickButton: function() {
                     jQuery("#list").jqGrid('columnChooser');
                }
            });
        });
    </script>
}
```

- برای نمایش این گرید، به یک جدول و یک div نیاز است. از جدول با id مساوی list جهت نمایش رکوردهای برنامه استفاده می‌شود. از div با id مساوی pager برای نمایش اطلاعات صفحه بندی و نوار ابزار پایین گرید کمک گرفته خواهد شد.
Div سومی با id مساوی rsperror نیز تعریف شده‌است که از آن جهت نمایش خطاهای بازگشت داده شده از سرور استفاده کرده‌ایم.
- در ادامه نحوه‌ی فراخوانی افزونه‌ی jqGrid را بر روی جدول list ملاحظه می‌کنید.
- خاصیت caption، عنوان نمایش داده شده در بالای گرید را مقدار دهی می‌کند:

آزمایش اول

	شماره	نام محصول	موجود است	قیمت
1	1	نام 1	☑	1000
2	2	نام 2	☐	1001
3	3	نام 3	☑	1002
4	4	نام 4	☐	1003
5	5	نام 5	☑	1004
6	6	نام 6	☐	1005
7	7	نام 7	☑	1006
8	8	نام 8	☐	1007
9	9	نام 9	☑	1008
10	10	نام 10	☐	1009
		جمع صفحه		10045

نمایش 1 - 10 از 500 صفحه 1 از 50 10 تنظیم نمایش ستون‌ها

- خاصیت url، به آدرسی اشاره می‌کند که قرار است ساختار JqGridData ایی را که پیشتر در مورد آن بحث کردیم، با فرمت JSON بازگشت دهد. در اینجا برای مثال به یک اکشن متد کنترلری در یک پروژه‌ی ASP.NET MVC اشاره می‌کند.
- datatype را برابر json قرار داده‌ایم. از نوع xml نیز پشتیبانی می‌کند.
- شیء jsonReader را از این جهت مقدار دهی کرده‌ایم تا بتوانیم شیء JqGridData را با اصول نامگذاری دات نت، هماهنگ کنیم. برای درک بهتر این موضوع، فایل jquery.jqGrid.src.js را باز کنید و در آن به دنبال تعریف jsonReader بگردید. به یک چنین مقادیر پیش فرضی خواهید رسید:

```
ts.p.jsonReader = $.extend(true,{
root: "rows",
page: "page",
total: "total",
records: "records",
repeatitems: true,
cell: "cell",
id: "id",
userdata: "userdata",
subgrid: {root:"rows", repeatitems: true, cell:"cell"}
},ts.p.jsonReader);
```

برای مثال سلول‌ها را با نام cell دریافت می‌کند که در شیء JqGridData به RowCells تغییر نام یافته‌است. برای اینکه این تغییر نام‌ها توسط jqGrid پردازش شوند، تنها کافی است jsonReader را مطابق تعاریفی که ملاحظه می‌کنید، مقدار دهی کرد.
- در ادامه mtype به GET تنظیم شده‌است. در اینجا مشخص می‌کنیم که عملیات Ajax ایی دریافت اطلاعات از سرور توسط GET انجام شود یا برای مثال توسط POST.
- خاصیت colNames، معرف نام ستون‌های گرید است. برای اینکه این نام‌ها از راست به چپ نمایش داده شوند، باید خاصیت direction به rtl تنظیم شود.
- colModel آرایه‌ای است که تعاریف ستون‌ها را در بر دارد. مقدار name آن باید یک نام منحصربفرد باشد. از این نام در حین جستجو یا ویرایش اطلاعات استفاده می‌شود. مقدار index نامی است که جهت مرتب سازی اطلاعات، به سرور ارسال می‌شود. تنظیم sorttype در اینجا مشخص می‌کند که آیا به صورت پیش فرض، ستون جاری رشته‌ای مرتب شود یا اینکه برای مثال عددی پردازش گردد. مقادیر مجاز آن text (مقدار پیش فرض)، date، integer، int ، numeric، currency، number، float و datetime

هستند.

- در ستون IsAvailable، مقدار formatter نیز تنظیم شده‌است. در اینجا توسط formatter، نوع bool دریافتی با یک checkbox نمایش داده خواهد شد.
- خاصیت pager به id متناظری در صفحه اشاره می‌کند.
- توسط rowNum مشخص می‌کنیم که در هر صفحه چه تعداد رکورد باید نمایش داده شوند.
- تعداد رکوردهای نمایش داده شده را می‌توان توسط rowList پویا کرد. در اینجا آرایه‌ای را ملاحظه می‌کنید که توسط اعداد آن، کاربر امکان انتخاب صفحاتی مثلا 100 ردیفه را نیز پیدا می‌کند. rowList به صورت یک dropdown در کنار عناصر راهبری صفحه در فوتر گرید ظاهر می‌شود.
- خاصیت sortname، نحوه‌ی مرتب سازی اولیه گرید را مشخص می‌کند.
- خاصیت sortorder، جهت مرتب سازی اولیه‌ی گردید را تنظیم می‌کند.
- viewrecords: تعداد رکوردها را در نوار ابزار پایین گرید نمایش می‌دهد.
- altRows: سبب می‌شود رنگ متن ردیف‌ها یک در میان متفاوت باشد.
- shrinkToFit: به معنای تنظیم خودکار اندازه‌ی سلول‌ها بر اساس اندازه‌ی داده‌ای است که دریافت می‌کنند.
- width: عرض گرید، که در اینجا به auto تنظیم شده‌است.
- height: طول گرید، که در اینجا به auto جهت محاسبه‌ی خودکار، تنظیم شده‌است.
- gridview: برای بالا بردن سرعت نمایشی به true تنظیم شده‌است. در این حالت کل ردیف یکباره درج می‌شود. اگر از subgird یا حالت نمایش درختی استفاده شود، باید این خاصیت را false کرد.
- rownumbers: ستون سمت راست شماره ردیف‌های خودکار را نمایش می‌دهد.
- footerrow: سبب نمایش ردیف فوتر می‌شود.
- userDataOnFooter: سبب خواهد شد تا خاصیت UserData مقدار دهی شده، در ردیف فوتر ظاهر شود.
- loadComplete : یک callback است که زمان پایان بارگذاری صفحه‌ی جاری را مشخص می‌کند. در اینجا با استفاده از jQuery سبب شده‌ایم تا رنگ پس زمینه‌ی ردیف‌ها یک در میان تغییر کند.
- loadError: اگر از سمت سرور خطایی صادر شود، در این callback قابل دریافت خواهد بود.
- در ادامه توسط فراخوانی متد jqGrid با پارامتر navGrid، در ناحیه pager سبب نمایش دکمه refresh شده‌ایم. این دکمه سبب بارگذاری مجدد اطلاعات گردید از سرور می‌شود.
- همچنین به کمک متد jqGrid با پارامتر navButtonAdd در ناحیه pager، سبب نمایش دکمه‌ای که صفحه‌ی انتخاب ستون‌ها را ظاهر می‌کند، خواهیم شد.

پیشنیاز کدهای سمت سرور jqGrid

اگر به تنظیمات گرید دقت کرده باشید، خاصیت index ستون‌ها، نامی است که به سرور، جهت اطلاع رسانی در مورد فیلتر اطلاعات و مرتب سازی مجدد آن‌ها ارسال می‌گردد. این نام، بر اساس کلیک کاربر بر روی ستون‌های موجود، هر بار می‌توان متفاوت باشد. بنابراین بجای if و else نوشتن‌های طولانی جهت مرتب سازی اطلاعات، می‌توان از کتابخانه‌ی معروفی به نام dynamic LINQ استفاده کرد.

```
PM> Install-Package DynamicQuery
```

به این ترتیب می‌توان قسمت orderby را به صورت پویا و با رشته‌ای دریافتی، مقدار دهی کرد.

کدهای سمت سرور بازگشت اطلاعات به فرمت JSON

در کدهای سمت کلاینت، به اکشن متد GetProducts اشاره شده بود. تعاریف کامل آن‌را در ذیل مشاهده می‌کنید:

```
using System;
using System.Collections.Generic;
using System.Globalization;
using System.Linq;
using System.Web.Mvc;
using jqGrid01.Models;
using jqGrid01.Extensions; // for dynamic OrderBy

namespace jqGrid01.Controllers
{
    public class HomeController : Controller
```

```
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult GetProducts(string sidx, string sord, int page, int rows)
        {
            var list = ProductDataSource.LatestProducts;

            var pageIndex = page - 1;
            var pageSize = rows;
            var totalRecords = list.Count;
            var totalPages = (int)Math.Ceiling(totalRecords / (float)pageSize);

            var products = list.AsQueryable()
                               .OrderBy(sidx + " " + sord)
                               .Skip(pageIndex * pageSize)
                               .Take(pageSize)
                               .ToList();

            var jqGridData = new JqGridData
            {
                UserData = new // نمایش در فوتر
                {
                    Name = "جمع صفحه",
                    Price = products.Sum(x => x.Price)
                },
                Total = totalPages,
                Page = page,
                Records = totalRecords,
                Rows = (products.Select(product => new JqGridRowData
                                                {
                                                    Id = product.Id,
                                                    RowCells = new List<string>
                                                    {

product.Id.ToString(CultureInfo.InvariantCulture),
                                                        product.Name,
                                                        product.IsAvailable.ToString(),

product.Price.ToString(CultureInfo.InvariantCulture)
                                                    }
                                                })).ToList()
            };
            return Json(jqGridData, JsonRequestBehavior.AllowGet);
        }
    }
}
```

- سطر ProductDataSource.LatestProducts چیزی نیست بجز لیست جنریکی از محصولات.
- امضای متد GetProducts نیز مهم است. دقیقا همین پارامترها با همین نام‌ها از طرف jqGrid به سرور ارسال می‌شوند که توسط آن‌ها ستون مرتب سازی، جهت مرتب سازی، صفحه‌ی جاری و تعداد ردیفی که باید بازگشت داده شوند، قابل دریافت است.
- در این کدها دو قسمت مهم وجود دارند:

الف) متد OrderBy نوشته شده، به صورت پویا عمل می‌کند و از کتابخانه‌ی Dynamic LINQ مایکروسافت بهره می‌برد.

به علاوه توسط Take و Skip کار صفحه بندی و بازگشت تنها بازه‌ای از اطلاعات مورد نیاز، انجام می‌شود.

ب) لیست جنریک محصولات، در نهایت باید با فرمت JqGridData به صورت JSON بازگشت داده شود. نحوه‌ی این Projection را در اینجا می‌توانید ملاحظه کنید.

هر ردیف این لیست، باید تبدیل شود به ردیفی از جنس JqGridRowData، تا توسط jqGrid قابل پردازش گردد.

- توسط مقدار دهی UserData، برچسبی را در ذیل ستون Name و مقداری را در ذیل ستون Price نمایش خواهیم داد.

برای مطالعه‌ی بیشتر

بهترین راهنمای جزئیات این Grid، مستندات آنلاین آن هستند:

http://www.trirand.com/jqgridwiki/doku.php?id=wiki:jqgriddocs

همچنین این مستندات را با فرمت PDF نیز می‌توانید مطالعه کنید:

http://www.trirand.com/blog/jqgrid/downloads/jqgriddocs.pdf

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید

jqGrid01.zip

نظرات خوانندگان

نویسنده: امانی فرد
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۳:۳۳

امکان دارد شکل گرید را بصورت گرید بوت استرپ نمایش داد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۴:۱۰

همانطور که در بحث عنوان شد، قالب این گرید از jQuery UI تامین می‌شود. بنابراین هر نوع قالب نوشته شده‌ای برای jQuery UI با آن سازگار است؛ مانند:

- jQuery UI Bootstrap
- jqGrid.bootstrap

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۳/۰۴/۱۱ ۱۹:۲۱

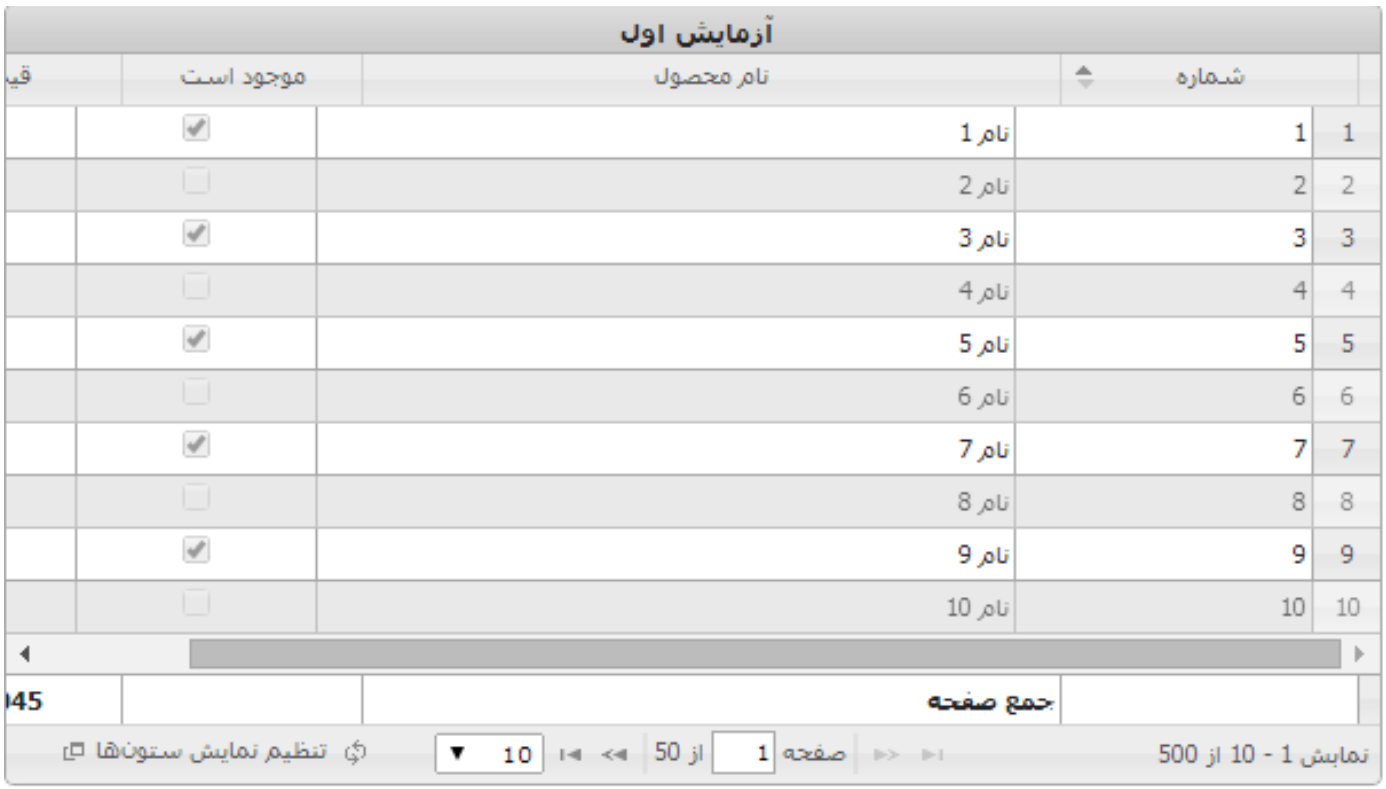
متاسفانه در حالت rtl تغییر اندازه‌ی ستون‌ها با باگ همراه هست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۲۰:۶

چه مشکلی دقیقا؟

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۳/۰۴/۱۱ ۲۰:۳۰

مثلا در اینجا سایز ستون Id را بیشتر کردم، ردیف‌های متناظرش باهاش هماهنگ نیستند.(با موس اندازه‌ی ستون را تغییر دهیم)



نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۲۰:۵۱

درسته. البته فقط با کروم 35 اینطوری هست. با IE 11 و فایرفاکس 30 تست کردم مشکلی نبود.

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۳/۰۴/۱۱ ۲۳:۴۳

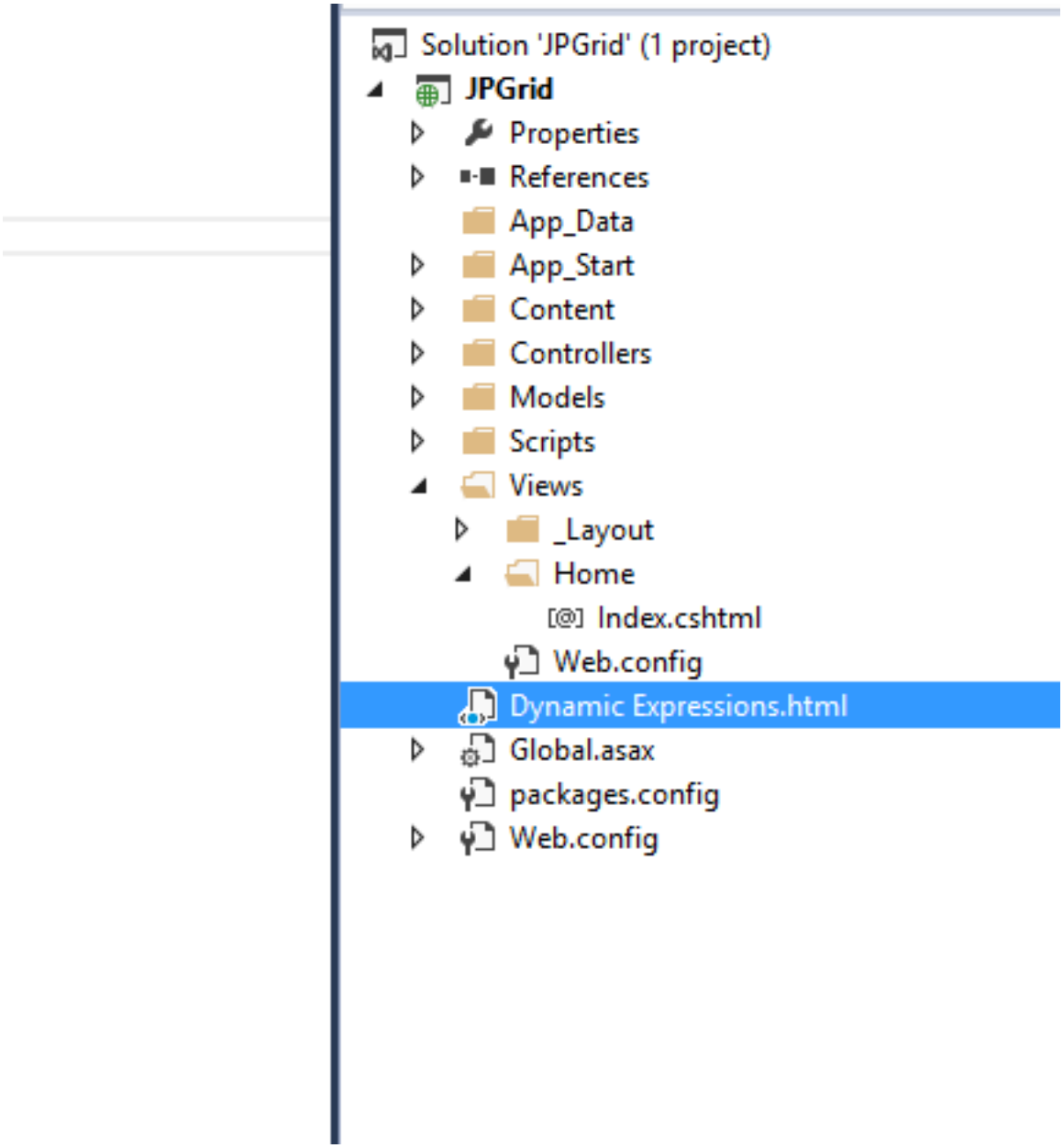
خیلی عالی. خیلی وقته که دنبال یک دیتا گرید می‌گردم تا مشکلی نداشته باشه. برای AngularJS یک Datagrid هست به نام ng-grid که خیلی خوب و کامل هست ولی rtl را خوب ساپورت نمی‌کنه. نسخه‌ی rtl هم که یه نفر براش نوشته باگهای زیادی داره مثل همین تغییر سایز ستون ها.
در کل سوالم این هست که باگ تغییر سایز ستون jqGrid از کروم هست یا خود jqGrid؟ چون می‌خوام از ng-grid کوچ کنم به jq-grid و ببینم ارزش این مهاجرت را داره یا نه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۲ ۰:۱۹

به نظر کروم در هر نگارشی نحوه‌ی تشخیص آن فرق می‌کند و این مساله مشکلاتی را به همراه داشته. بحثی در این مورد .

نویسنده: محمد دلیری
تاریخ: ۱۳۹۳/۰۴/۱۶ ۲۲:۱۳

وقتی این کتاب خانه (Install-Package DynamicQuery) رو اضافه کردم پوشه‌ی به نام Extensions و کلاسی به نام LinqExtensions.cs نساخت فقط چیزی که در تصویر زیر میبینید ساخته ؟ چکار باید بکنم ؟



نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۶ ۲۲:۴۵

این بسته، کامپایل شده‌ی آن فایل را تحت عنوان Dynamic.dll به لیست ارجاعات پروژه اضافه می‌کند. فایل Dynamic Expressions.html هم راهنمای آن است. به عبارتی همه چیز آماده است؛ از فضای نام جدید System.Linq.Dynamic آن استفاده کنید.

نویسنده: daniyal
تاریخ: ۱۳۹۳/۰۴/۲۱ ۱۵:۲۷

سلام
آیا می‌شه ستون هایی که عدم نمایش آنها انتخاب شده اند را از طریق یک پارامتر در getproducts گرفت
اگر می‌شه چه طوری ؟

نویسنده: وحید نصیری

تاریخ: ۱۳۹۳/۰۴/۲۱ ۱۸:۵۶

columnChooser رویداد done هم دارد که از آن می‌شود برای آنالیز ستون‌ها استفاده کرد:

```
.jqGrid('navButtonAdd', '#pager',
            {
                caption: "تنظیم نمایش ستون‌ها", title: "Reorder Columns",
                onClickButton: function () {
                    jQuery("#list").jqGrid('columnChooser', {
                        done: function (perm) {
                            if (perm) {
                                $("#list").jqGrid("remapColumns", perm, true, false);
                            }

                            var colModel = $("#list").jqGrid('getGridParam', 'colModel');
                            var hiddenColumns = new Array();
                            for (var i = 0; i < colModel.length; i++) {
                                if (colModel[i].hidden) {
                                    hiddenColumns.push(colModel[i].name);
                                }
                            }

                            $.ajax({
                                type: "POST",
                                url: "@Url.Action("HiddenColumns","Home")",
                                dataType: "json",
                                traditional: true,
                                data: { hiddenColumns: hiddenColumns }
                            });
                        }
                    });
                }
            })
```

متد getGridParam با پارامتر colModel، آرایه‌ای از خواص ستون‌ها را بر می‌گرداند.

```
var colModel = $("#list").jqGrid('getGridParam', 'colModel');
```

در این خواص اگر hidden مساوی true بود، یعنی مخفی شده‌است. در نهایت از این‌ها می‌شود یک آرایه را تشکیل داد و به سرور ارسال کرد:

```
public ActionResult HiddenColumns(string[] hiddenColumns)
        {
            //todo: save it in the DB or cookies or session ....

            return Content("OK");
        }
```

امضای اکشن متدی است که لیست نام ستون‌های مخفی را دریافت می‌کند.

نویسنده: daniyal
تاریخ: ۱۳۹۳/۰۴/۲۲ ۹:۳۰

خیلی مچکرم؛ من می‌خواستم بدونم آیا می‌تونم بدون فرستادن لیست ستون‌های پنهان شده به یک اکشن دیگر و در نتیجه ذخیره کردن آن بر روی کوکی و یا دیتابیس و یا روش‌های دیگر مستقیم در اکشن GetProducts بگیرتش، اگر بخوام شفاف‌تر بگم ، من می‌خوام وقتی کاربر دکمه خروجی پی دی اف رو زد ستون‌های گرید رو کاربر قبلش تنظیم کرده باشه و در پی دی اف مورد نظر ستون‌های حذف شده رو دیگر نبینه همچنین با در نظر گرفتن موارد فیلتر شده در گرید .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۲۲ ۱۲:۲

برای ارسال پارامترهای دلخواه به سرور از خاصیت postData استفاده کنید:

```
function getHiddenColumnsList() {
            var colModel = $("#list").jqGrid('getGridParam', 'colModel');
            var hiddenColumns = new Array();

            if (!colModel)
                return hiddenColumns;

            for (var i = 0; i < colModel.length; i++) {
                if (colModel[i].hidden) {
                    hiddenColumns.push(colModel[i].name) ;
                }
            }
            return hiddenColumns;
        }

      $(document).ready(function () {
            $('#list').jqGrid({
                    // ...
                    postData: { 'hiddenColumns': function() { return getHiddenColumnsList(); } }
                    // ...
                });
        });
```

سمت سرور در اکشن متد GetProducts، خاصیت جدید hiddenColumns به صورت یک رشته که عناصر آن با کاما از هم جدا شده‌اند، قابل دریافت و آنالیز است.

و برای گزارش‌گیری با Pdf Report در تعریف ستون‌ها (مثلا ستون Id):

```
column.IsVisible(hiddenColumns.Split(',').All(col => col != "Id"));
```

یک نکته: ذکر function در postData ضروری است؛ وگرنه فقط یکبار محاسبه می‌شود.

نویسنده: alireza
تاریخ: ۱۳۹۳/۰۵/۱۰ ۱۵:۳۲

سلام خسته نباشید. فایل‌های پروژه‌هاتون را که دانلود میکنم، وقتی اجرا میکنم ارور زیر را میدهد:

```
Could not load file or assembly 'System.Web.Mvc, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35' or one of its dependencies. The system cannot find the file specified.
```

دلیلش چیه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۱۰ ۱۷:۳۳

- ابتدا به اینترنت وصل شوید.
- سپس در خط فرمان پاورشل نیوگت (^ و ^) دستور زیر را اجرا کنید:

```
PM> update-package -reinstall
```

به این صورت بسته‌های MVC 5 آن به صورت صحیح به پروژه اضافه می‌شوند.

نویسنده: alireza
تاریخ: ۱۳۹۳/۰۵/۱۱ ۱۱:۳

با mvc5 باید باشه ؟ من mvc 4 دارم

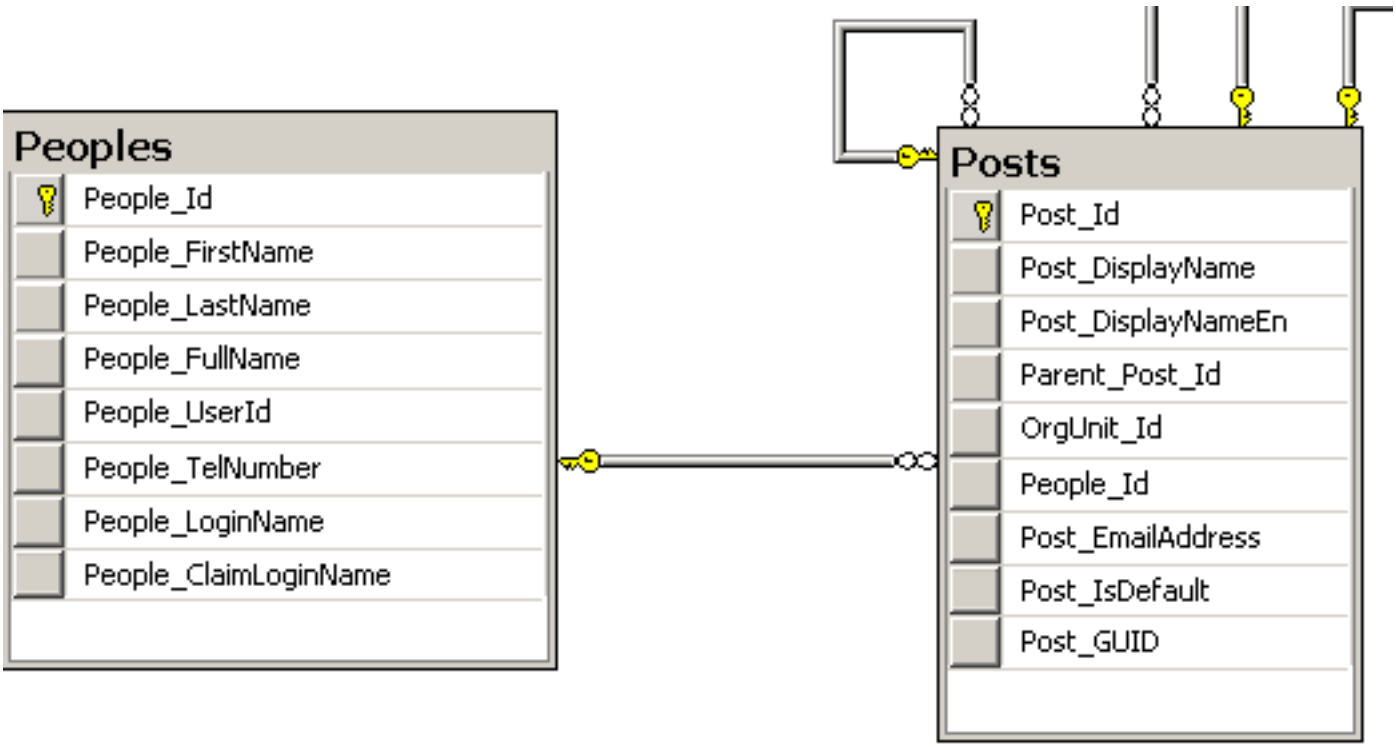
نویسنده: وحید نصیری

تاریخ: ۱۳۹۳/۰۵/۱۱ ۱۱:۲۱

خیر. یک پروژه خالی MVC4 ایجاد کنید. بعد پوشه‌ها و فایل‌های این پروژه را در آن اضافه کنید (منهای فایل‌های کانفیگ)؛ کار می‌کند. jqGrid هیچگونه وابستگی به فناوری‌های سمت سرور ندارد. با وب فرم‌ها قابل استفاده است. با PHP هم قابل استفاده است. در اینجا از MVC فقط برای بازگشت اطلاعات مورد نیاز آن به فرمت JSON استفاده شده‌است. در این مثال خاص، کاربرد ویژه‌ی دیگری ندارد. می‌شد بجای آن از Web API هم استفاده کرد.

نویسنده: ناصر پورعلی
تاریخ: ۱۳۹۳/۰۵/۱۴ ۱۴:۲۲

سلام؛ از مثالی که زدید دارم استفاده میکنم، در کلاس ReflectionHelper متد FindFieldType موقعی که از جداول خود ارجاع دهنده استفاده میکنیم(در اینجا جدول پست)، خطا stackoverflow می‌گیرم، کجاش رو باید چک کنیم که فراخوانی بی نهایت اتفاق نیفته؟ مگه فیلد dumplevel همچین کاری انجام نمیده؟
data model م به این صورته :



نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۱۴ ۱۵:۶

«مگه فیلد dumplevel همچین کاری انجام نمیده؟»
بله. dump level از stack overflow جلوگیری می‌کند. اما جای دیگری است که stack overflow واقعی رخ می‌دهد:
« بررسی خطای Circular References در ASP.NET MVC Json Serialization »

نویسنده: بهاره
تاریخ: ۱۳۹۳/۰۵/۲۰ ۱۶:۰

سلام
اگه بخوام نام ستون‌ها رو از روی مدل یا ویو بگ بخونم و دستی ننویسم باید چیکار کنم. من از مدل Database First استفاده

می‌کنم. چطور میتونم اینکار رو انجام بدم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۰ ۱۷:۱۴

« استفاده ازExpressionها جهت ایجاد Strongly typed view در ASP.NET MVC »

نویسنده: احسان شیروان
تاریخ: ۱۳۹۳/۰۶/۱۲ ۲۱:۷

با سلام و خسته نباشید من یه مشکلی داشتم می‌خواستم در صورت امکان راهنمایی نمایید :
توی یه صفحه می‌خوام لود شدن داده‌های این گرید با رویداد کلیک یه دکمه اتفاق بیفته یه مقدار سرچ هم زدم که تقریبا همشون می‌گفتن توی رویداد کلیک بنویسم :

```
jQuery("#list").trigger("reloadGrid");
```

اما با این هم جواب نداد و اصلا ActionMethod فراخوانی نمیشه ممنون میشم راهنمایی نمایید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۲ ۲۱:۲۴

reloadGrid برای حالتی است که Grid در صفحه نمایش داده شده و موجود است. برای نمایش یک گرید با کلیک بر روی یک دکمه، کل کدهای داخل document.ready مثال فوق را داخل یک متد جداگانه قرار دهید و سپس آن‌را مستقلا فراخوانی کنید.
document.ready یعنی به محض آماده شدن DOM این اطلاعات را اجرا کن.

نویسنده: احسان شیروان
تاریخ: ۱۳۹۳/۰۶/۱۲ ۲۱:۳۹

راستش اولش هم همین کارو کردم جواب نگرفتم تستش هم کردم مطمئنم که رویداد کلیک و تابع مذکور فراخوانی میشه اما متد سرور فراخوانی نمیشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۲ ۲۲:۳۸

پس از آزمایش مشکلی مشاهده نشد.
همانطور که عنوان شد، کل اطلاعات داخل document ready به داخل یک متد مجزا منتقل شد:

```
function loadGrid() {
            $('#list').jqGrid({
                caption: "آزمایش اول",
                // .....
            });
        }
```

و سپس فراخوانی آن توسط یک دکمه:

```
<button onclick="loadGrid()">Load Grid</button>
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۲۰ ۱۳:۹

یک نکته‌ی تکمیلی

نسخه‌ی جدید و زنده‌ی dynamic LINQ در اینجا نگهداری می‌شود.

```
PM> Install-Package System.Linq.Dynamic.Library
```