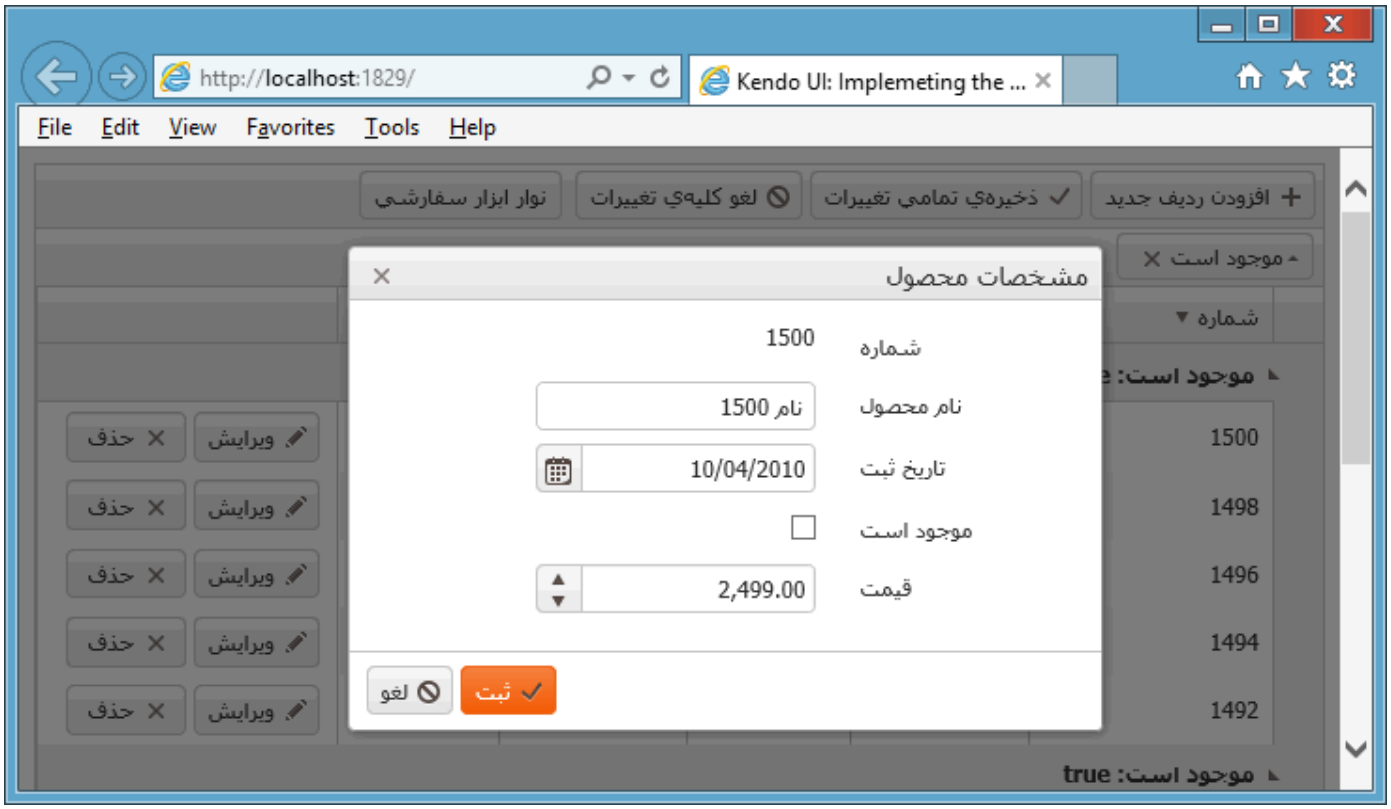
عنوان: **فعال سازی عملیات CRUD در Kendo UI Grid**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۱ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, ASP.NET Web API, jQuery, Kendo UI

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

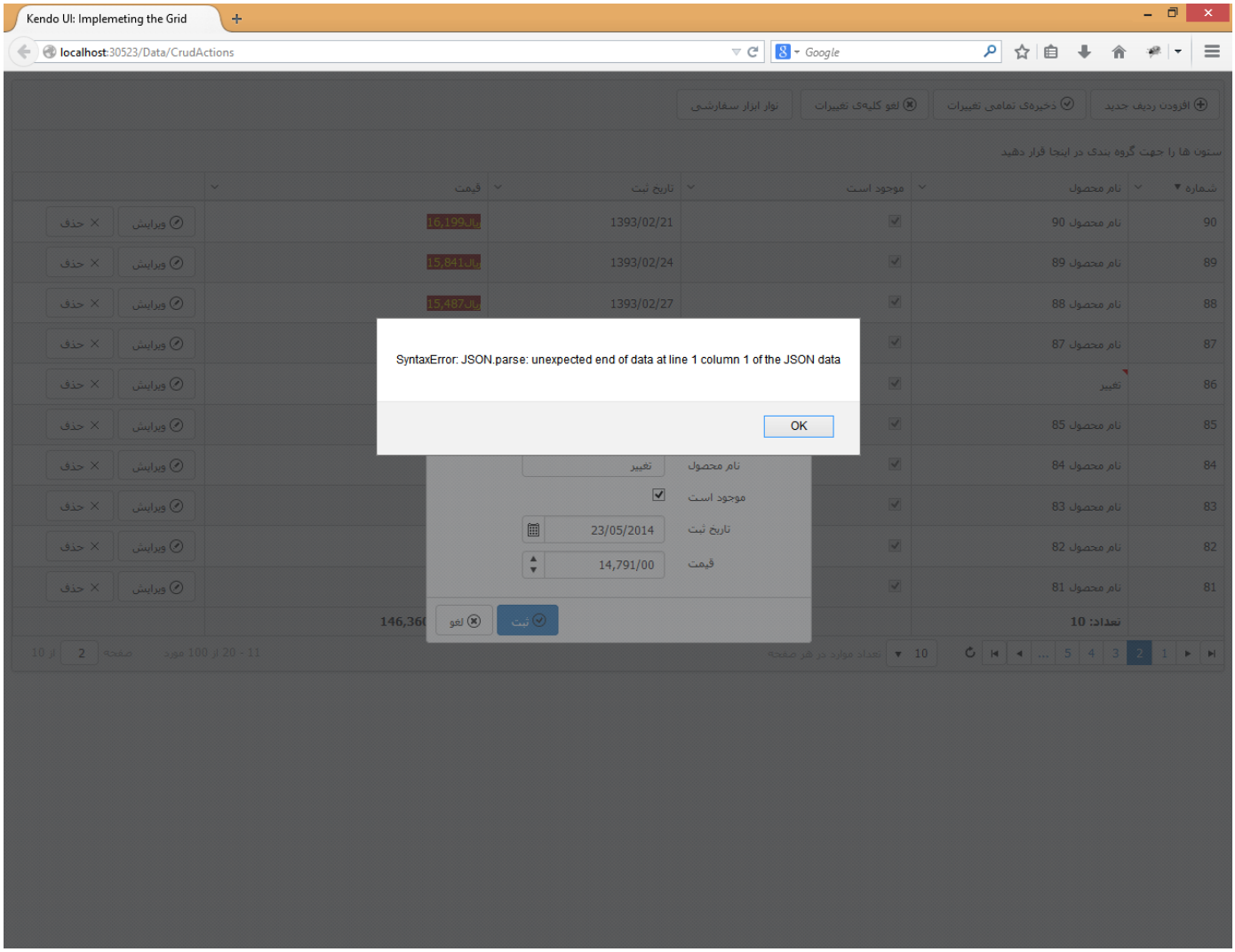
در بحث Grid در پلتفرم KendoUI آیا راهی هست که بتونیم بوسیله کوکی براحتی آخرین وضعیت گرید را ذخیره کنیم تا در مراجعه بعدی بصورت اتوماتیک وضعیت قبلی را لود کنه؟ متشکرم

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.

نویسنده: جوادنب
تاریخ: ۱۳۹۳/۱۲/۲۲ ۱۱:۶

سلام؛ در مثال فوق وقتی میخوام یک رکور جدید درج کنم خطای Internal Server Error میده!
راستی این تکه کد منظور از مقدار name چیه؟

```
toolbar: [
                    { name: "create", text: "افزودن ردیف جدید" },
                    { name: "save", text: "ذخیره‌ی تمامی تغییرات" },
                    { name: "cancel", text: "لغو کلیه‌ی تغییرات" },
                    { template: kendo.template($("#toolbarTemplate").html()) }
            ],
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۲/۲۲ ۱۱:۲۱

- جزئیات خطای عمومی Internal Server Error را در برگه‌ی response فایرباگ می‌توانید مشاهده کنید. اطلاعات بیشتر
- همچنین ELMAH را هم می‌توانید نصب کنید تا خطاها را بهتر بتوانید بررسی کنید.
- کدهای مثال جاری را بازنویسی شده جهت ASP.NET MVC و بدون استفاده از Web API در اینجا می‌توانید مشاهده کنید. با این View و این Controller . کدهای سمت کلاینت و سمت سرور خودتان را با این دو فایل انطباق دهید.
- name، نام یک سری command و دستور از پیش تعریف شده‌ی Kendo UI Grid است.

نویسنده: جوادنب
تاریخ: ۱۳۹۳/۱۲/۲۲ ۱۳:۳۵

همه چیز را چک کردم ولی بازم هم خطا میده!
این هم جزییات response

```
<h2> <i>Invalid JSON primitive: Id.</i> </h2></span>
[ArgumentException: Invalid JSON primitive: Id.]
System.Web.Script.Serialization.JavaScriptObjectDeserializer.DeserializePrimitiveObject() +585
```

من وقتی این صفحه داره Get میشه با استفاده از تکه کد زیر خروجی بر میگردونم. مشکل نداره؟

```
string json = new JavaScriptSerializer().Serialize(list);
return json;
```

یعنی خروجی string بر میگردونم. مشکل داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۲/۲۲ ۱۳:۵۴

- بله. مشکل دارد. خروجی دریافتی این گرید باید با فرمت DataSourceResult به صورت JSON ارسال شود (به صورت JSON ارسال شدن، شامل فرمت اطلاعات و همچنین تنظیم Content-Type آن است). جزئیات این فرمت و علت وجودی آن در مطلب « صفحه بندی، مرتب سازی و جستجوی پویای اطلاعات به کمک Kendo UI Grid » بررسی شده‌اند. به همین جهت در این مثال‌ها از متد ToDataSourceResult کمک گرفته شده‌است (هم در مثال وب API و هم در مثال MVC آن)
- زمانیکه که خطای Invalid JSON primitive را در سمت سرور دریافت می‌کنید، یعنی سمت کلاینت، اطلاعات را با فرمت نادرستی به سمت سرور ارسال می‌کند و این فرمت JSON نیست. یعنی در قسمت DataSource اطلاعات اضافی و یا نادرستی وجود دارند که با View ذکر شده هماهنگ نیستند.

همانطور که عنوان شد، قسمت‌های سمت سرور و سمت کلاینت را با مثال ارسالی هماهنگ کنید و *انطباق* دهید. اگر وب API است این مثال و اگر MVC است، این مثال .

نویسنده: رضا نادری
تاریخ: ۱۳۹۴/۰۱/۰۴ ۰:۳۶

سلام؛ من میخوام در حین آپدیت یکی از رکورد هام یکی از فیلدهای من از نوع text area باشه تا کاربر بتونه متن بیشتری را وارد نماید. منظورم در حین popup هستش.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۱/۰۴ ۱:۵۰

```
$("#grid").kendoGrid({
// ...
    columns:
    [
        {
            field: "Your Field",
            title: "Your Field Name",
            width: "20%",
            editor: function (container, options) {
                $('<textarea cols="20" rows="4" data-bind="value: ' + options.field +
'"></textarea>').appendTo(container);
            }
        },
        // ...
   ]
// ...
});
```