

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 `Hash (#) syntax` 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 `HTML` است که در محل `container1` درج خواهد شد. همانطور که ملاحظه می‌کنید، متد `kendo.template`، نهایتاً 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 شده است:

```
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 نهایتاً 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 شده است. 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، قطعه 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، صرفاً 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؛ دقیقاً در کنار فایل‌های اصلی vs.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](#)