

بعد از آمدن نسخه‌ی سوم ASP.NET MVC مکانیسمی به نام Remote Validation به آن اضافه شد که کارش اعتبارسنجی از راه دور بود. فرض کنید نیاز است در یک فرم، قبل از اینکه کل فرم به سمت سرور ارسال شود، مقداری بررسی شده و اعتبارسنجی آن انجام گیرد و این اعتبارسنجی چیزی نیست که بتوان سمت کاربر و بدون فرستاده شدن مقداری به سمت سرور صورت گیرد. نمونه بارز این مسئله صفحه عضویت اکثر سایت‌هایی هست که روزانه داریم با آن‌ها کار می‌کنیم. فیلد نام کاربری توسط شما پر شده و بعد از بیرون آمدن از آن فیلد، سریعاً مشخص می‌شود که آیا این نام کاربری قابل استفاده برای شما هست یا خیر. به‌صورت معمول برای انجام این کار باید با جاوا اسکریپت، مدیریتی روی فیلد مربوطه انجام دهیم. مثلاً با بیرون آمدن فوکوس از روی فیلد، با Ajax نام کاربری وارد شده را به سمت سرور بفرستیم، چک کنیم و بعد از اینکه جواب برگشت بررسی کنیم که الان آیا این نام کاربری قبلاً گرفته شده یا نه.

انجام این کار به‌راحتی با مزین کردن خصوصیت (Property) مربوطه موجود در مدل برنامه به Attribute یا ویژگی Remote و داشتن یک Action در Controller مربوطه که کارش بررسی وجود یوزرنیم هست امکان پذیر است. ادامه بحث را با مثال همراه می‌کنم.

به عنوان مثال در سیستمی که قرار هست محصولات ما را ثبت کند، باید بیایم و قبل از اینکه محصول جدید به ثبت برسد این عملیات چک‌کردن را انجام دهیم تا کالای تکراری وارد سیستم نشود. شناسه اصلی که برای هر محصول وجود دارد بارکد هست و ما آن را می‌خواهیم مورد بررسی قرار دهیم.

### مدل برنامه

```
public class ProductModel
{
    public int Id { get; set; }

    [Display(Name = "نام کالا")]
    [Required(ErrorMessage = "{0} باید آن را وارد کنید {0}")]
    [StringLength(50, ErrorMessage = "کاراکتر باشد {1} از {0} بزرگتر باشد")]
    public string Name { get; set; }

    [Display(Name = "قیمت")]
    [Required(ErrorMessage = "{0} باید آن را وارد کنید {0}")]
    [DataType(DataType.Currency)]
    public double Price { get; set; }

    [Display(Name = "بارکد")]
    [Required(ErrorMessage = "{0} باید آن را وارد کنید {0}")]
    [StringLength(50, ErrorMessage = "کاراکتر باشد {1} از {0} بزرگتر باشد")]
    [Remote("IsProductExist", "Product", HttpMethod = "POST", ErrorMessage = "این بارکد از قبل در سیستم وجود دارد")]
    public string Barcode { get; set; }
}
```

همونطور که می‌بینید خصوصیت Barcode را مزین کردیم به ویژگی Remote. این ویژگی دارای ورودی‌های خاص خودش هست. وارد کردن نام اکشن و کنترلر مربوطه برای انجام این چک‌کردن از مهم‌ترین قسمت‌های اصلی هست. چیزهایی دیگه‌ای هم هست که می‌توانیم آن‌ها را مقداردهی کنیم. مثل HttpMethod, ErrorMessage و یا HttpMethod, AdditionFields که همان طریقه‌ی ارسال درخواست به سرور هست. ErrorMessage هم همان خطایی هست که در زمان رخداد قرار است نشان داده شود. AdditionFields هم خصوصیتی را مشخص می‌کند که ما می‌خواهیم به‌همراه فیلد مربوطه به سمت سرور بفرستیم. مثلاً می‌تونیم به‌همراه بارکد، نام کالا را هم برای بررسی‌های مورد نیازمان بفرستیم.

### کنترلر برنامه

```
[HttpPost]
[OutputCache(Location = OutputCacheLocation.None, NoStore = true)]
public ActionResult IsProductExist(string barcode)
```

```
{  
    if (barcode == "123456789") return Json(false); // اگر محصول وجود داشت  
    return Json(true);  
}
```

در اینجا به نمایش قسمتی از کنترلر برنامه می‌پردازیم. اکشنی که مربوط می‌شود به چک کردن مقدارهای لازم و در پایان آن یک خروجی Json را برمی‌گردانیم که مقدار true یا false دارد. در حقیقت مقدار را به این صورت برمی‌گردانیم که اگر مقدار ورودی در پایگاه داده وجود دارد، false را برمی‌گرداند و اگر وجود نداشت true. همین‌طور آمدم از کش شدن درخواست‌هایی که با Ajax آمده با ویژگی OutputCache جلوگیری کردیم.

## کالای جدید

<input type="text"/>	نام کالا
<input type="text"/>	قیمت
<input type="text" value="۱۲۳۴۵۶۷۸۹"/>	بارکد

این بارکد از قبل در سیستم وجود دارد.

## نظرات خوانندگان

نویسنده: مجتبی

تاریخ: ۱۳۹۳/۰۵/۱۲ ۱۳:۳

مشکل این روش این است که بعد از یک بار اعتبار سنجی دفعه بعد با زدن هر کلید داخل تکست باکس می‌خواهد بره به بانک و اطلاعات را چک کنه.

نویسنده: علی اکبر کورش فر

تاریخ: ۱۳۹۳/۰۵/۱۴ ۸:۳۴

بله، البته. ولی به نظر شما چه نیازی هستش وقتی که فیلد مربوطه پر شد دوباره به فیلد برگرده. این فقط در حالتی هستش که کاربر بخواد مقدار رو تغییر بده. پس اولویت داره استفاده از این کار در برابر استفاده نکردن

نویسنده: جوکار

تاریخ: ۱۳۹۳/۰۶/۰۱ ۱۵:۴۸

روش بسیار جالبی بود. اما یک مشکل که در این روش با آن روبرو میشویم این است که هنگام ویرایش یک رکورد موجود در بانک، اگر قرار نباشد فیلد مورد نظر بروز رسانی شود، اکشن متد ذکر شده دست ما را می‌بندد و اجازه آپدیت را نمی‌دهد. شاید یک راه ایجاد یک viewModel جداگانه برای آپدیت باشد که در آن از remote attribute صرف نظر شود، اما این راه حل زیاد جالب به نظر نمی‌رسد! آیا راه حل مناسب‌تری وجود دارد؟

نویسنده: محسن خان

تاریخ: ۱۳۹۳/۰۶/۰۲ ۱۹:۰۶

برای حالت ویرایش AdditionFields آن کاربرد داره. مثلاً فیلد Id رو اینجا همیشه ارسال کرد تا مشخص باشه حالت ویرایش هست. در حالت ثبت معمولی، خوب هنوز Id رکورد مشخص نیست و نال هست.

نویسنده: علی خسروی

تاریخ: ۱۳۹۳/۰۷/۲۲ ۱۲:۵۷

با سلام و تشکر؛ در صورتی که قصد داشته باشیم هنگام remote یک loader هم نشون بدیم باید چکار کرد.

نویسنده: محسن خان

تاریخ: ۱۳۹۳/۰۷/۲۲ ۱۴:۱۲

از روال‌های رخدادگردان عمومی [ajaxStart](#) و [ajaxComplete](#) استفاده کنید.

نویسنده: م علی خسروی

تاریخ: ۱۳۹۳/۰۷/۲۳ ۹:۰۷

با تشکر

در واقع من سه input دارم که می‌خام remote بشند برای هر کدام هم یه loader در کنارش قرار دارم، چه طوری میشه فهمید الان کدوم input در صفحه ajax رو start کرده تا loader اون نمایش داده بشه

نویسنده: وحید نصیری

تاریخ: ۱۳۹۳/۰۷/۲۳ ۱۰:۴۷

اگر به فایل jquery.validate.js مراجعه کنید، در قسمت remote آن، متد startRequest پیش از شروع عملیات Ajax و متد

stopRequest پس از پایان کار فراخوانی می‌شوند.

```
prototype: {
  startRequest: function( element ) {
    //...
  },
  stopRequest: function( element, valid ) {
    //...
  },
}
```

این دو متد را باید برای نمایش loading بازنویسی کرد. برای مثال:

```
var originalStartRequest = $.validator.prototype.startRequest;
$.validator.prototype.startRequest = function (element) {
  // یافتن عنصر در حال بررسی
  var container = $('form').find("[data-valmsg-for='" + element.name + "']");
  // افزودن کلاس نمایش منتظر بمانید
  container.addClass('loading');

  // فراخوانی متد اصلی برای انجام کارهای درونی افزونه
  originalStartRequest.apply(this, arguments);
};

var originalStopRequest = $.validator.prototype.stopRequest;
$.validator.prototype.stopRequest = function (element) {
  // یافتن عنصر در حال بررسی
  var container = $('form').find("[data-valmsg-for='" + element.name + "']");
  // حذف کلاس نمایش منتظر بمانید
  container.removeClass('loading');

  // فراخوانی متد اصلی برای انجام کارهای درونی افزونه
  originalStopRequest.apply(this, arguments);
};
```

در اینجا loading به span مخفی data-valmsg-for اضافه می‌شود.

```
<span class="field-validation-valid" data-valmsg-replace="true" data-valmsg-for="Url"></span>
```

نمونه‌ی این بازنویسی در مطلب « [اعتبارسنجی سمت کاربر wysiwyg-editorها در ASP.NET MVC](#) » هم انجام شده‌است.