عنوان: **اعتبارسنجی درخواست های $http با استفاده از یک Interceptor**
نویسنده: محمود راستین
تاریخ: ۱۳۹۴/۰۶/۱۱ ۱۰:۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, MVC, Security, AngularJS, ASP.NET Identity

پیش نیاز :

هدایت خودکار کاربر به صفحه لاگین در حین اعمال Ajax ایی
Angular Interceptors

ابتدا مشکل و هدف را بیان می‌کنیم:

مشکل : کاربر در صفحه‌ای حضور دارد که نیاز به اعتبارسنجی داشته و مدت اعتبار کاربر نیز تمام شده است، ولی هنوز در صفحه‌ای که نباید حضور داشته باشد، حضور دارد و بدتر از آن این است که می‌تواند درخواست‌های بی نتیجه‌ای را نیز ارسال کند.
هدف : کاربر را سریعا به صفحه‌ای که به آن تعلق دارد هدایت کنیم (یعنی صفحه‌ی ورود به سیستم).

و حالا از ابتدا پروسه را دنبال می‌کنیم. یک Controller سمت سرور داریم به این صورت :

```
[Authorize(Roles = AuthorizeRole.SuperAdministrator)]
    public partial class HomeController : Controller
    {
        [HttpPost]
        [AngularValidateAntiForgeryToken]
        public virtual JsonResult GetUserInfo()
        {
            var userInfoViewModel = _applicationUserManager.GetUserInfoById(User.Identity.GetUserId());
            return Json(userInfoViewModel);
        }
    }
```

همانطور که مشاهده می‌کنید از Authorize پیش فرض استفاده کرده‌ایم. حالا سمت کلاینت با استفاده از HTTP می‌خواهیم درخواستی را ارسال کنیم.

یعنی با کدی همانند کد زیر:

```
$scope.getUserInfo = function() {
            $http({
                    method: 'POST',
                    url: 'Home/GetUserInfo',
                    headers: $scope.getHeaders()
                }).
                success(function(data, status, headers, config) {
                    $scope.userInfo = data;
                }).
                error(function(data, status, headers, config) {

                }).then(function(res) {

                });
        }
```

و نتیجه‌ی کدهای بالا به صورت زیر درخواهد آمد :

```
success(function(data, status, headers, config) {  data = Object {UserName: "dotnettips.info"}, status = 200,
    $scope.userInfo = data;
}).
```

همانطور که می‌بینید داده‌های اولیه کاربر پس از ورود به سیستم، بدون هیچ مشکلی دریافت می‌شوند.

نکته : زمانیکه status برابر با 200 هست، یعنی درخواست OK می‌باشد. (در پیوست ، لیست تمامی کدها قرار داده شده است)

حالا فرض کنید کاربر در صفحه حضور دارد و به هر دلیلی اعتبار حضور کاربر منقضی شده است و حالا پس از مدتی کاربر درخواستی را به سرور ارسال می‌کند و می‌خواهد اطلاعات خودش را مشاهده کند.

درخواست کاربر با همان کدهای اولیه ارسال می‌شود و خروجی اینبار به صورت زیر در خواهد آمد :

```
success(function(data, status, headers, config) {  data = "", status = 200,
    $scope.userInfo = data;
}).
```

همانطور که می‌بینید وضعیت اینبار نیز OK می‌باشد، ولی هیچ داده‌ای از سرور دریافت نشده است. کاربر قطعا در اینجا دچار سردرگمی می‌شود. چون هیچ چیزی را مشاهده نمی‌کند و به هیچ مسیر دیگری نیز هدایت نمی‌شود و هرکار دیگری نیز انجام دهد، پاسخی مشاهده نمی‌کند.

راه حل چیست ؟

ابتدا باید برای درخواست‌های Ajax ایی اعتبارسنجی را اعمال کنیم. برای این کار باید یک Attribute جدید بسازیم. یعنی به این صورت :

```
[AttributeUsage(AttributeTargets.Class | AttributeTargets.Method, Inherited = true, AllowMultiple =
true)]
    public class MyCustomAuthorize : AuthorizeAttribute
    {
        protected override void HandleUnauthorizedRequest(AuthorizationContext filterContext)
        {
            if (filterContext.HttpContext.Request.IsAjaxRequest())
            {
                // یعنی اعتبارسنجی نشده است
                filterContext.HttpContext.Response.StatusCode = (int)HttpStatusCode.Unauthorized;
                filterContext.HttpContext.Response.End();
            }

            base.HandleUnauthorizedRequest(filterContext);
        }
    }
```

با استفاده از Attribute بالا ما درخواست‌های ای‌جکسی را نیز اعتبارسنجی می‌کنیم.

توجه : کد کامل‌تر همراه با توضیحات در بخش پیش نیازها آمده است.

حالا در سمت کلاینت نیز به این صورت عمل می‌کنیم :

```
$http({
                    method: 'POST',
                    url: 'Home/GetUserInfo',
                    headers: $scope.getHeaders()
```

```
                }).
                success(function(data, status, headers, config) {
                    $scope.userInfo = data;
                }).
                error(function(data, status, headers, config) {
                    if (status === 401) {
                        // you are not authorized
                    }
                }).then(function(res) {

                });
```

حالا دیگر متوجه خواهیم شد که کاربر اعتبارسنجی نشده است، یا اعتبار آن منقضی شده است و می‌توانیم کاربر را به مسیر **"ورود به سیستم"** هدایت کنیم.

در console مرورگر نیز خطای زیر رخ می‌دهد :

```
POST http://localhost:000000/Administrator/Home/GetUserInfo 401 (Unauthorized)
```

اکنون به هدف خودمان رسیده‌ایم. اما برای هر درخواست باید این دستوارت را تکرار کنیم ؟! قطعا خیر.

حالا باید از یک Interceptor استفاده و درخواست‌های HTTP خودمان را مدیریت کنیم. برای این منظور ما یک Interceptor جدید را همانند کدهای زیر می‌نویسیم:

```
factory('AuthorizedInterceptor', function ($q) {
        return {
            response: function(response) {
                return response || $q.when(response);
            },
            responseError: function(rejection) {
                if (rejection.status === 401) {
                    // you are not authorized
                }
                return $q.reject(rejection);
            }
        };
    })
```

همانطور که مشاهده می‌کنید کدهای خطای درخواست $http را در Interceptor قرار دادیم. حالا کاربر درخواستی را ارسال می‌کند و ما با توجه به این Interceptor متوجه خواهیم شد که آیا اعتبار آن منقضی شده است یا خیر و در صورت منقضی شدن، کاربر را به صفحه‌ی ورود هدایت خواهیم کرد. یعنی بدین صورت :

```
}).factory('AuthorizedInterceptor', function ($q) {
    return {
        response: function(response) {
            return response || $q.when(response);
        },
        responseError: function(rejection) {  rejection = Object {data: "", status: 401, config: Object, statusText: "Unauthorized"}
            if (rejection.status === 401) {
                // you are not authorized
            }
            return $q.reject(rejection);
        }
    };
});
```

در خروجی بالا می‌بینید که کاربر، اعتبارسنجی نشده است و آن را به مسیر ورود هدایت خواهیم کرد.

با Interceptor بالا دیگر نیازی نیست برای هر درخواستی این وضعیت را چک کنیم و به صورت خودکار این چک کردن رخ خواهد داد.

پیوست : http_status_codes.rar

نظرات خوانندگان

نویسنده: یاسر مرادی
تاریخ: ۱۳۹۴/۰۶/۱۳ ۱۳:۱

با سلام و احترام

ممنون بابت مقاله خوبتون

ولی این شرط

```
filterContext.HttpContext.Request.IsAjaxRequest
```

معنی

یعنی اعتبارسنجی نشده است

را نمی‌دهد.

شما در متد HandleUnauthorizedRequest هستید و بالطبع اعتبار درخواست رد شده است، کاری که می‌خواهید انجام دهید، اعتبارسنجی نیست، بلکه اصلاح Response درخواست‌های Ajax ای است که Status Code آنها ۲۰۰ است که نباید باشد.

این نوع دقت داشتن و شفافیت لازمه ایجاد یک کد صحیح است، چون الآن و در مرحله ای بالاتر، می‌توانیم بگوییم که هر Response ای که وارد متد HandleUnauthorizedRequest شده است، اگر Status Code 401 ندارد، Status Code آن برابر ۴۰۱ قرار داده شود. فارغ از Ajax بودن یا نبودن.

و باز در مرحله ای بالاتر اگر در پروژه از Owin استفاده شده باشد، یک Owin Middleware و اگر نه یک ASP.NET Module جای مناسب‌تری برای نوشتن این چنین کدی است.

از این که IsAjaxRequest همیشه درست کار نمی‌کند و فقط وقتی درخواست Ajax ای بگوید که Ajax ای است، کار خواهد کرد نیز بگذریم.

موفق و پایدار باشید.

نویسنده: محسن خان
تاریخ: ۱۳۹۴/۰۶/۱۳ ۱۳:۲۷

نویسنده‌ی گرامی، خواهشمند است جهت رفاه حال خوانندگان گرانقدر، جمله‌ی مبهم «یعنی اعتبارسنجی نشده است» را به جمله‌ی بهتر «یعنی درخواست ای‌جکسی در این مرحله از کار اعتبارسنجی‌اش رد شده‌است و اکنون می‌خواهیم وضعیت خروجی آن‌را اصلاح کنیم» تغییر دهید. با تشکر فراوان!

نویسنده: محمود راستین
تاریخ: ۱۳۹۴/۰۶/۱۴ ۲:۵۶

با سلام
عبارت " یعنی اعتبارسنجی نشده است " برای تعریف خط زیرین خودش هستش که داره پاسخ رو اصلاح می‌کنه و برای شرط

استفاده نشده است که این پاسخ ما با توجه به کد وضعیت ،یعنی اعتبار سنجی نشده است . به بیان ساده‌تر معنی تغییر کد پاسخ می‌باشد.
من شخصا از مطلب جناب نصیری در اینجا استفاده می‌کنم برای این اعتبارسنجی. پیشنهاد میکنم این مطلب و نظرات اون رو ببینید. همچنین این مطلب در پیش نیازها درج شد تا در آینده خوانندگان دچار مشکل نشوند و ابتدا این مطلب را بخوانند.
و جناب مرادی در خط آخر مطلبی رو بیان کردید ، خوشحال میشیم جزئیات بیشتری رو بیان کنید. یعنی چی همیشه درست کار نمی‌کند ؟ چه شرایطی ممکن است رخ دهد که اشتباه کار کند ؟

با تشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۴/۰۶/۱۴ ۱۰:۱۹

چه شرایطی ممکن است رخ دهد که اشتباه کار کند ؟

اگر این درخواست دستی صادر بشه (یک ربات باشه) یا اگر فایروال خاصی در این میان هدرهای Ajax رو حذف کنه . مورد اول مهم نیست، چون نیازی نیست تا ربات را به صفحه‌ی لاگین هدایت کرد (مرورگر باز شده‌ای نداره). مورد دوم هم خیلی خیلی بعید هست. یعنی من ندیدم که در عمل فایروال‌های شرکت‌ها هدرهای HTTP رو تا این حد دستکاری کنند.