عنوان: **مثال - نمایش درصد پیشرفت عملیات توسط SignalR**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۰۱ ۱۵:۱۲
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, jQuery, SignalR

برنامه‌های وب در سناریوهای بسیاری نیاز دارند تا درصد پیشرفت عملیاتی را به کاربران گزارش دهند. نمونه ساده آن، گزارش درصد پیشرفت میزان دریافت یک فایل است و یا اعلام درصد انجام یک عملیات طولانی از سمت سرور به کاربر. در ادامه قصد داریم این موضوع را توسط SignalR پیاده سازی کنیم.

نکته‌ای در مورد نگارش‌های مختلف SignalR

اگر برنامه شما قرار است دات نت 4 را پشتیبانی کند، آخرین نگارش SignalR که با آن سازگار است، نگارش 1.1.3 می‌باشد.
بنابراین اگر دستور ذیل را اجرا کنید:

```
PM> Install-Package Microsoft.AspNet.SignalR
```

SignalR 2 را نصب می‌کند که با دات نت 4 و نیم به بعد سازگار است.
اگر دستور ذیل را اجرا کنید، Siganlr 1.x را نصب می‌کند که با دات نت 4 به بعد سازگار است:

```
PM> Install-Package Microsoft.AspNet.SignalR -Version 1.1.3
```

پیش فرض این مطلب نیز استفاده از نگارش 1.1.3 می‌باشد تا بازه بیشتری از وب سرورها را شامل شود.
با اینکار Microsoft.AspNet.SignalR.JS نیز به صورت خودکار نصب می‌گردد و به این ترتیب کلاینت جاوا اسکریپتی Siganlr نیز در برنامه قابل استفاده خواهد بود.

تنظیمات فایل Global.asax.cs

سطر فراخوانی متد RouteTable.Routes.MapHubs باید در ابتدای متد Application_Start فایل Global.asax.cs قرار گیرد (پیش از هر تنظیم دیگری). تفاوتی هم نمی‌کند که برنامه وب فرم است یا MVC. به این ترتیب مسیریابی‌های SignalR تنظیم شده و مسیر http://localhost/signalr/hubs قابل استفاده خواهد بود.

تنظیمات اسکریپت‌های سمت کلاینت مورد نیاز

پس از نصب بسته SignalR، سه اسکریپت ذیل باید به ابتدای صفحه وب اضافه شوند تا کلاینت‌های جاوا اسکریپتی SignalR بتوانند با سرور ارتباط برقرار کنند:

```
<script src="Scripts/jquery-1.6.4.min.js" type="text/javascript"></script>
<script src="Scripts/jquery.signalR-1.1.3.min.js" type="text/javascript"></script>
<script src="signalr/hubs" type="text/javascript"></script>
```

این تنظیمات نیز برای هر دو نوع برنامه‌های وب فرم و MVC یکسان است.

تعریف کلاس Hub برنامه

```
using Microsoft.AspNet.SignalR;

namespace WebFormsSample03.Common
{
    public class ProgressHub : Hub
    {
        /// <summary>
```

```
        /// این متد استاتیک تعریف شده تا در برنامه به صورت مستقیم قابل استفاده باشد
        /// یا می‌شد اصلا این متد تعریف نشود و از همان دریافت زمینه هاب در کنترلر استفاده گردد
        /// </summary>
        public static void UpdateProgressBar(int value, string connectionId)
        {
            var ctx = GlobalHost.ConnectionManager.GetHubContext<ProgressHub>();
            ctx.Clients.Client(connectionId).updateProgressBar(value); //فراخوانی یک متد در سمت کلاینت
        }
    }
}
```

متدی که در کلاس هاب برنامه تعریف شده، از نوع استاتیک است. از این جهت که می‌خواهیم این متد را در خارج از این هاب و در یک کنترلر Web API فراخوانی کنیم. زمانیکه متدی به صورت استاتیک تعریف می‌شود، ارتباط آن با وهله جاری کلاس یا this قطع خواهد شد. به همین جهت نیاز است تا از طریق متد GlobalHost.ConnectionManager.GetHubContext مجددا به context کلاس هاب دسترسی پیدا کنیم.

البته تعریف این متد در اینجا ضروری نبود. حتی می‌شد بدنه کلاس هاب را خالی تعریف کرد و متد GetHubContext را مستقیما داخل یک کنترلر فراخوانی نمود.

متد UpdateProgressBar، مقدار value را به تنها یک کلاینت که Id آن مساوی connectionId دریافتی است، ارسال می‌کند. این کلاینت باید یک callback جاوا اسکریپتی را جهت تامین متد پویای updateProgressBar تدارک ببیند.

کلاس Web API کنترلر دریافت فایل‌ها

فرقی نمی‌کند که برنامه شما از نوع وب فرم است یا MVC. امکانات Web API در هر دو نوع پروژه، قابل دسترسی است (همان ایده یک ASP.NET واحد).

بنابراین نیاز است یک کنترلر وب API جدید را به پروژه اضافه کرده و محتوای آن را به شکل ذیل تغییر دهیم:

```
using System.Threading;
using System.Web.Http;
using WebFormsSample03.Common;

namespace WebFormsSample03
{
    public class DownloadRequest
    {
        public string Url { set; get; }
        public string ConnectionId { set; get; }
    }

    public class DownloaderController : ApiController
    {
        public void Post([FromBody]DownloadRequest data)
        {
            //todo: start downloading the data.Url ....

            ProgressHub.UpdateProgressBar(10, data.ConnectionId);
            Thread.Sleep(2000);

            ProgressHub.UpdateProgressBar(40, data.ConnectionId);
            Thread.Sleep(3000);

            ProgressHub.UpdateProgressBar(64, data.ConnectionId);
            Thread.Sleep(2000);

            ProgressHub.UpdateProgressBar(77, data.ConnectionId);
            Thread.Sleep(2000);

            ProgressHub.UpdateProgressBar(92, data.ConnectionId);
            Thread.Sleep(3000);

            ProgressHub.UpdateProgressBar(99, data.ConnectionId);
            Thread.Sleep(2000);

            ProgressHub.UpdateProgressBar(100, data.ConnectionId);
        }
    }
}
```

اگر برنامه شما وب فرم است، باید تنظیمات مسیریابی ذیل را نیز به آن افزود. در برنامه‌های MVC4 این تنظیم به صورت پیش فرض وجود دارد:

```
using System;
using System.Web.Http;
using System.Web.Routing;

namespace WebFormsSample03
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            // Register the default hubs route: ~/signalr
            RouteTable.Routes.MapHubs();

            RouteTable.Routes.MapHttpRoute(
                name: "DefaultApi",
                routeTemplate: "api/{controller}/{id}",
                defaults: new { id = RouteParameter.Optional }
            );
        }
    }
}
```

کاری که در این کنترلر انجام شده، شبیه سازی یک عملیات طولانی توسط متد Thread.Sleep است. همچنین این کنترلر، id کلاینت درخواست کننده یک url را نیز دریافت می‌کند. بنابراین می‌توان به نحو بهینه‌ای، تنها نتایج پیشرفت عملیات را به این کلاینت ارسال کرد و نه به سایر کلاینت‌ها.
همچنین در اینجا با توجه به مسیریابی تعریف شده، باید اطلاعات را به آدرس api/Downloader از نوع Post ارسال کرد.

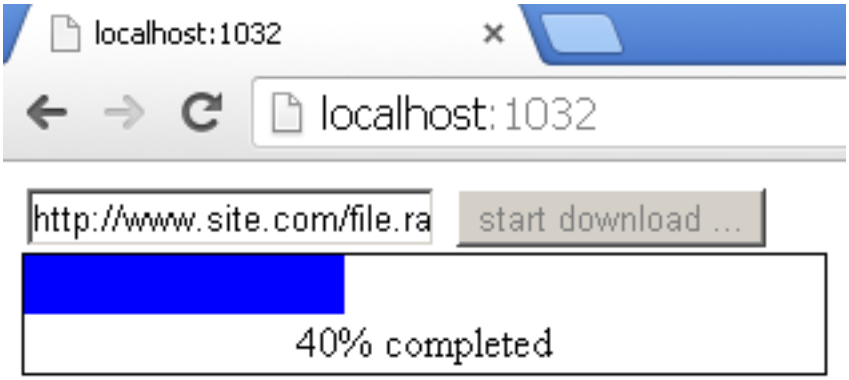
تعریف کلاینت متصل به Hub

در سمت سرور، متد پویای updateProgressBar فراخوانی شده است. اکنون باید این متد را در سمت کلاینت پیاده سازی کنیم:

```
<form id="form1" runat="server">
    <div>
    <input id="txtUrl" value="http://www.site.com/file.rar" type="text" />
        <input id="send" type="button" value="start download ..." />
        <br />
        <div id="bar" style="border: #000 1px solid; width:300px;"></div>
    </div>
    </form>
    <script type="text/javascript">
        $(function () {
            $.connection.hub.logging = true; //اطلاعات بیشتری را در جاوا اسکریپت کنسول مرورگر لاگ می‌کند
            var progressHub = $.connection.progressHub; //این نام مستعار پیشتر توسط ویژگی نام هاب تنظیم شده است
            progressHub.client.updateProgressBar = function (value) {
                //متدی که در اینجا تعریف شده دقیقا مطابق نام متد پویایی است که در هاب تعریف شده است
                //به این ترتیب سرور می‌تواند کلاینت را فراخوانی کند
                $("#bar").html(GaugeBar.generate(value));
            };
            $.connection.hub.start() // فاز اولیه ارتباط را آغاز می‌کند
            .done(function () {
                $("#send").click(function () {
                    $("#send").attr('disabled', 'disabled');
                    var myClientId = $.connection.hub.id;
                    // اکنون اتصال برقرار است به سرور
                    $.ajax({
                        type: "POST",
                        contentType: "application/json",
                        url: "/api/Downloader",
                        data: JSON.stringify({ Url: $("#txtUrl").val(), ConnectionId: myClientId })
                    }).success(function () {
                        $("#send").removeAttr('disabled');
                    }).fail(function () {
                        //
                    });
                });
            });
```

```
    });
</script>
```

بر روی این فرم، یک جعبه متنی که Url را دریافت می‌کند و یک دکمه‌ی آغاز کار دریافت این Url، وجود دارد.
در ابتدای کار صفحه، اتصال به progressHub برقرار می‌شود. اگر دقت کنید، نام این هاب با حروف کوچک در اینجا (در سمت کلاینت) آغاز می‌گردد.
سپس با تعریف یک callback به نام progressHub.client.updateProgressBar، پیام‌های دریافتی از طرف سرور را به یک افزونه progress bar جی‌کوئری، برای نمایش ارسال می‌کند.
کار اتصال به رویداد کلیک دکمه‌ی آغاز دریافت فایل، در متد done باید انجام شود. این callback زمانی فراخوانی می‌گردد که کار اتصال به سرور با موفقیت صورت گرفته باشد.
سپس در ادامه توسط jQuery Ajax، اطلاعات Url و همچنین Id کلاینت را به مسیر api/Downloader یا همان web api controller ارسال می‌کنیم.



کدهای کامل این مثال را از اینجا نیز می‌توانید دریافت نمائید:

WebFormsSample03.zip

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۰ ۱۱:۱

جهت اطلاع؛ همیشه آخرین نسخه x.1 مخصوص دات نت 4 را در صفحه ذیل بررسی کنید:

http://www.nuget.org/packages/Microsoft.AspNet.SignalR

برای مثال در این تاریخ Microsoft ASP.NET SignalR 1.1.4 نسخه آخر x.1 است و از لحاظ امنیتی نیاز است این به روز رسانی صورت گیرد.