عنوان: **نگاهی به SignalR Clients**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۱۲ ۱۲:۱۱
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, jQuery, SignalR

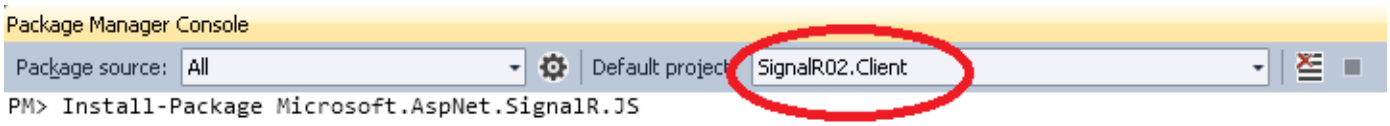
در قسمت قبل موفق به ایجاد اولین Hub خود شدیم. در ادامه، برای تکمیل برنامه نیاز است تا کلاینتی را نیز برای آن تهیه کنیم. مصرف کنندگان یک Hub می‌توانند انواع و اقسام برنامه‌های کلاینت مانند jQuery Clients و یا حتی یک برنامه کنسول ساده باشند و همچنین Hubهای دیگر نیز قابلیت استفاده از این امکانات Hubهای موجود را دارند. تیم SignalR امکان استفاده از Hubهای آن‌را در برنامه‌های دات نت 4 به بعد، برنامه‌های WinRT، ویندوز فون 8، سیلورلایت 5، jQuery و همچنین برنامه‌های CPP نیز مهیا کرده‌اند. به علاوه گروه‌های مختلف نیز با توجه به سورس باز بودن این مجموعه، کلاینت‌های iOS via Mono، iOS Native و Android via Mono را نیز به این لیست اضافه کرده‌اند.

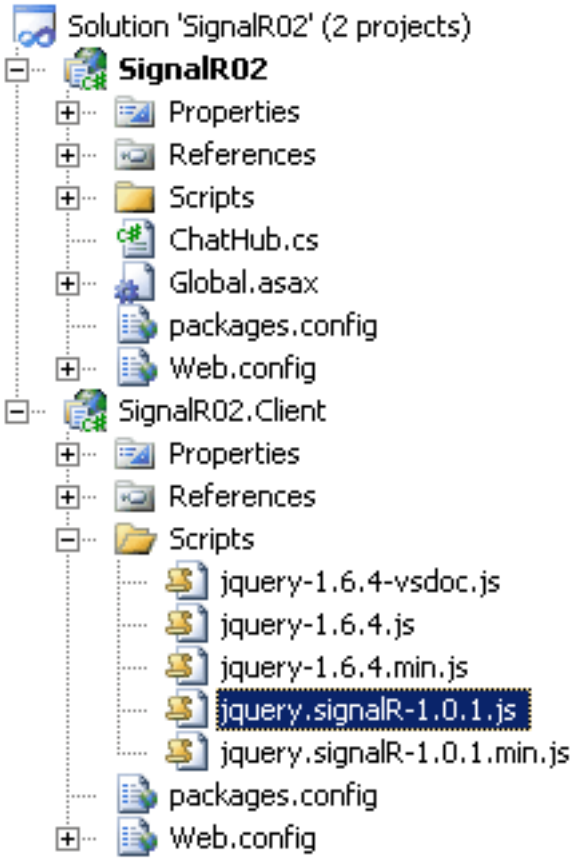
بررسی کلاینت‌های jQuery

با توجه به پروتکل مبتنی بر JSON سیگنال‌آر، استفاده از آن در کتابخانه‌های جاوا اسکریپتی همانند jQuery نیز به سادگی مهیا است. برای نصب آن نیاز است در کنسول پاور شل نوگت، دستور زیر را صادر کنید:

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

برای نمونه به solution پروژه قبل، یک برنامه وب خالی دیگر را اضافه کرده و سپس دستور فوق را بر روی آن اجرا نمائید. در این حالت فقط باید دقت داشت که فرامین بر روی کدام پروژه اجرا می‌شوند:





با استفاده از افزونه jQuery SignalR، به دو طریق می‌توان به یک Hub اتصال برقرار کرد:
الف) استفاده از فایل proxy تولید شده آن (این فایل، در زمان اجرای برنامه تولید می‌شود و یا امکان استفاده از آن به کمک ابزارهای کمکی نیز وجود دارد)
نمونه‌ای از آن‌را در قسمت قبل ملاحظه کردید؛ همان فایل تولید شده در مسیر signalr/hubs/ برنامه. به نوعی به آن Service contract نیز گفته می‌شود (ارائه متادیتا و قراردادهای کار با یک سرویس Hub). این فایل همانطور که عنوان شد به صورت پویا در زمان اجرای برنامه ایجاد می‌شود.
امکان تولید آن توسط برنامه کمکی signalr.exe نیز وجود دارد؛ برای دریافت آن می‌توان از طریق NuGet اقدام کرد (بسته Microsoft.AspNet.SignalR.Utils) که نهایتا در پوشه packages قرار خواهد گرفت. نحوه استفاده از آن نیز به صورت زیر است:

```
Signalr.exe ghp http://localhost/
```

در این دستور ghp مخفف generate hub proxy است و نهایتا فایلی را به نام server.js تولید می‌کند.

ب) بدون استفاده از فایل proxy و به کمک روش late binding (انقیاد دیر هنگام)

برای کار با یک Hub از طریق jQuery مراحل ذیل باید طی شوند:
1) ارجاعی به Hub باید مشخص شود.
2) روال‌های رخدادگردان تنظیم گردند.
3) اتصال به Hub برقرار گردد.
4) متدی فراخوانی شود.

در اینجا باید دقت داشت که امکانات Hub به صورت خواص

```
$.connection
```

در سمت کلاینت جی‌کوئری، در دسترس خواهند بود. برای مثال:

```
$.connection.chatHub
```

و نام‌های بکارگرفته شده در اینجا مطابق روش‌های متداول نام گذاری در جاوا اسکریپت، camel case هستند.

خوب، تا اینجا فرض بر این است که یک پروژه خالی ASP.NET را آغاز و سپس فرمان نصب Microsoft.AspNet.SignalR.JS را نیز همانطور که عنوان شد، صادر کرده‌اید. در ادامه یک فایل ساده html را به نام chat.htm، به این پروژه جدید اضافه کنید (برای استفاده از کتابخانه جاوا اسکریپتی SignalR الزامی به استفاده از صفحات کامل پروژه‌های وب نیست).

```
<!DOCTYPE>
<html>
<head>
    <title></title>
    <script src="Scripts/jquery-1.6.4.min.js" type="text/javascript"></script>
    <script src="Scripts/jquery.signalR-1.0.1.min.js" type="text/javascript"></script>
    <script src="http://localhost:1072/signalr/hubs" type="text/javascript"></script>
</head>
<body>
    <div>
        <input id="txtMsg" type="text" /><input id="send" type="button" value="send msg" />
        <ul id="messages">
        </ul>
    </div>
    <script type="text/javascript">
        $(function () {
            var chat;
            $.connection.hub.logging = true; //اطلاعات بیشتری را در جاوا اسکریپت کنسول مرورگر لاگ
می‌کند
            chat = $.connection.chat; //این نام مستعار پیشتر توسط ویژگی نام هاب تنظیم شده است
            chat.client.hello = function (message) {
                //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
                $("#messages").append("<li>" + message + "</li>");
            };
            $.connection.hub.start(/*{ transport: 'longPolling' }*/); // فاز اولیه ارتباط را آغاز می‌کند

            $("#send").click(function () {
                // Hub's `SendMessage` should be camel case here
                chat.server.sendMessage($("#txtMsg").val());
            });
        });
    </script>
</body>
</html>
```

کدهای آن‌را به نحو فوق تغییر دهید.

توضیحات:

همانطور که ملاحظه می‌کنید ابتدا ارجاعاتی به jquery و jquery.signalR-1.0.1.min.js اضافه شده‌اند. سپس نیاز است مسیر دقیق فایل پروکسی هاب خود را نیز مشخص کنیم. اینکار با تعریف مسیر signalr/hubs انجام شده است.

```
<script src="http://localhost:1072/signalr/hubs" type="text/javascript"></script>
```

در ادامه توسط تنظیم connection.hub.logging سبب خواهیم شد تا اطلاعات بیشتری در javascript console مرورگر لاگ شود.

سپس ارجاعی به هاب تعریف شده، تعریف گردیده است. اگر از قسمت قبل به خاطر داشته باشید، توسط ویژگی HubName، نام chat را برگزیدیم. بنابراین connection.chat ذکر شده دقیقا به این هاب اشاره می‌کند.

سپس سطر chat.client.hello مقدار دهی شده است. متد hello، متدی dynamic و تعریف شده در سمت هاب برنامه است. به این ترتیب می‌توان به پیام‌های رسیده از طرف سرور گوش فرا داد. در اینجا، این پیام‌ها، به li ایی با id مساوی messages اضافه می‌شوند.

سپس توسط فراخوانی متد connection.hub.start، فاز negotiation شروع می‌شود. در اینجا حتی می‌توان نوع transport را نیز صریحا انتخاب کرد که نمونه‌ای از آن را به صورت کامنت شده جهت آشنایی با نحوه تعریف آن مشاهده می‌کنید. مقادیر قابل استفاده در آن به شرح زیر هستند:

```
- webSockets
- forverFrame
- serverSentEvents
- longPolling
```

سپس به رویدادهای کلیک دکمه send گوش فرا داده و در این حین، اطلاعات TextBox ایی با id مساوی txtMsg را به متد SendMessage هاب خود ارسال می‌کنیم. همانطور که پیشتر نیز عنوان شد، در سمت کلاینت، تعریف متد SendMessage باید camel case باشد.

اکنون به صورت جداگانه یکبار برنامه hub را در مرورگر باز کنید. سپس بر روی فایل chat.htm کلیک راست کرده و گزینه مشاهده آن را در مرورگر نیز انتخاب نمائید (گزینه View in browser منوی کلیک راست).

خوب! پروژه کار نمی‌کند! برای اینکه مشکلات را بهتر بتوانید مشاهده کنید نیاز است به JavaScript Console مرورگر خود مراجعه نمائید. برای مثال در مرورگر کروم دکمه F12 را فشرده و برگه Console آن‌را باز کنید. در اینجا اعلام می‌کند که فاز negotiation قابل انجام نیست؛ چون مسیر پیش فرضی را که انتخاب کرده است، همین مسیر پروژه دومی است که اضافه کرده‌ایم (کلاینت ما در پروژه دوم قرار دارد و نه در همان پروژه اول هاب).
برای اینکه مسیر دقیق hub را در این حالت مشخص کنیم، سطر زیر را به ابتدای کدهای جاوا اسکریپتی فوق اضافه نمائید:

```
$.connection.hub.url = 'http://localhost:1072/signalr'; //چون در یک پروژه دیگر قرار داریم
```

اکنون اگر مجدا سعی کنید، باز هم برنامه کار نمی‌کند و پیام می‌دهد که امکان دسترسی به این سرویس از خارج از دومین آن میسر نیست. برای اینکه این مجوز را صادر کنیم نیاز است تنظیمات مسیریابی پروژه هاب را به نحو ذیل ویرایش نمائیم:

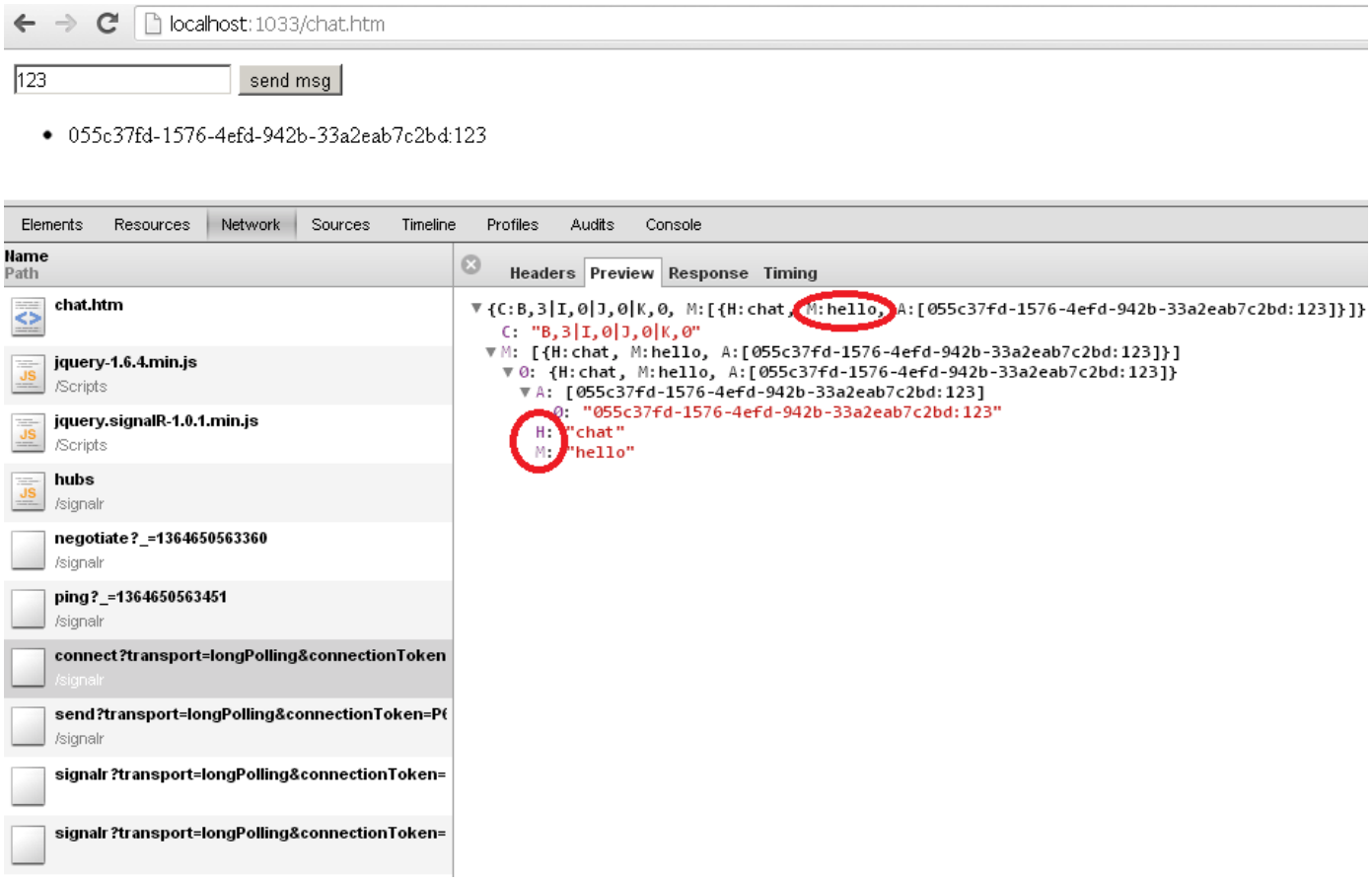
```
using System;
using System.Web;
using System.Web.Routing;
using Microsoft.AspNet.SignalR;

namespace SignalR02
{
    public class Global : HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            // Register the default hubs route: ~/signalr
            RouteTable.Routes.MapHubs(new HubConfiguration
            {
                EnableCrossDomain = true
            });
        }
    }
}
```

با تنظیم EnableCrossDomain به true اینبار فازهای آغاز ارتباط با سرور برقرار می‌شوند:

```
SignalR: Auto detected cross domain url. jquery.signalR-1.0.1.min.js:10
SignalR: Negotiating with 'http://localhost:1072/signalr/negotiate'. jquery.signalR-1.0.1.min.js:10
SignalR: SignalR: Initializing long polling connection with server. jquery.signalR-1.0.1.min.js:10
SignalR: Attempting to connect to
'http://localhost:1072/signalr/connect?transport=longPolling&connectionToken…NRh72omzsPkKqhKw2&connecti
onData=%5B%7B%22name%22%3A%22chat%22%7D%5D&tid=3' using longPolling. jquery.signalR-1.0.1.min.js:10
SignalR: Longpolling connected jquery.signalR-1.0.1.min.js:10
```

مطابق این لاگ‌ها ابتدا فاز negotiation انجام می‌شود. سپس حالت long polling را به صورت خودکار انتخاب می‌کند.



در برگه شبکه، مطابق شکل فوق، امکان آنالیز اطلاعات رد و بدل شده مهیا است. برای مثال در حالتیکه سرور پیام دریافتی را به کلیه کلاینت‌ها ارسال می‌کند، نام متد و نام هاب و سایر پارامترها در اطلاعات به فرمت JSON آن به خوبی قابل مشاهده هستند.

یک نکته:

اگر از ویندوز 8 (یعنی IIS8) و VS 2012 استفاده می‌کنید، برای استفاده از حالت Web socket، ابتدا فایل وب کانفیگ برنامه را باز کرده و در قسمت httpRunTime، مقدار ویژگی targetFramework را بر روی 4.5 تنظیم کنید. اینبار اگر مراحل negotiation را بررسی کنید در همان مرحله اول برقراری اتصال، از روش Web socket استفاده گردیده است.

تمرین 1

به پروژه ساده و ابتدایی فوق یک تکست باکس دیگر به نام Room را اضافه کنید؛ به همراه دکمه join. سپس نکات قسمت قبل را در مورد الحاق به یک گروه و سپس ارسال پیام به اعضای گروه را پیاده سازی نمائید. (تمام نکات آن با مطلب فوق پوشش داده شده است و در اینجا باید صرفا فراخوانی متدهای عمومی دیگری در سمت هاب، صورت گیرد)

تمرین 2

در انتهای قسمت دوم به نحوه ارسال پیام از یک هاب به هابی دیگر اشاره شد. این MonitorHub را ایجاد کرده و همچنین یک کلاینت جاوا اسکریپتی را نیز برای آن تهیه کنید تا بتوان اتصال و قطع اتصال کلیه کاربران سیستم را مانیتور و مشاهده کرد.

پیاده سازی کلاینت jQuery بدون استفاده از کلاس Proxy

در مثال قبل، از پروکسی پویای مهیای در آدرس signalr/hubs استفاده کردیم. در اینجا قصد داریم، بدون استفاده از آن نیز کار برپایی کلاینت را بررسی کنیم.

بنابراین یک فایل جدید html را مثلا به نام chat_np.html به پروژه دوم برنامه اضافه کنید. سپس محتویات آن‌را به نحو زیر تغییر دهید:

```
<!DOCTYPE>
<html>
<head>
    <title></title>
    <script src="Scripts/jquery-1.6.4.min.js" type="text/javascript"></script>
    <script src="Scripts/jquery.signalR-1.0.1.min.js" type="text/javascript"></script>
</head>
<body>
    <div>
        <input id="txtMsg" type="text" /><input id="send" type="button" value="send msg" />
        <ul id="messages">
        </ul>
    </div>
    <script type="text/javascript">
        $(function () {
            $.connection.hub.logging = true; //اطلاعات بیشتری را در جاوا اسکریپت کنسول مرورگر لاگ می‌کند

            var connection = $.hubConnection();
            connection.url = 'http://localhost:1072/signalr'; //چون در یک پروژه دیگر قرار داریم
            var proxy = connection.createHubProxy('chat');

            proxy.on('hello', function (message) {
                //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
                $("#messages").append("<li>" + message + "</li>");
            });

            $("#send").click(function () {
                // Hub's `SendMessage` should be camel case here
                proxy.invoke('sendMessage', $("#txtMsg").val());
            });

            connection.start();
        });
    </script>
</body>
</html>
```

در اینجا سطر مرتبط با تعریف مسیر اسکریپت‌های پویای signalr/hubs را دیگر در ابتدای فایل مشاهده نمی‌کنید. کار تشکیل proxy اینبار از طریق کدنویسی صورت گرفته است. پس از ایجاد پروکسی، برای گوش فرا دادن به متدهای فراخوانی شده از طرف سرور از متد proxy.on و نام متد فراخوانی شده سمت سرور استفاده می‌کنیم و یا برای ارسال اطلاعات به سرور از متد proxy.invoke به همراه نام متد سمت سرور استفاده خواهد شد.

کلاینت‌های دات نتی SignalR

تا کنون Solution ما حاوی یک پروژه Hub و یک پروژه وب کلاینت جی‌کوئری است. به همین Solution، یک پروژه کلاینت کنسول ویندوزی را نیز اضافه کنید.

سپس در خط فرمان پاور شل نوگت دستور زیر را صادر نمائید تا فایل‌های مورد نیاز به پروژه کنسول اضافه شوند:

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

در اینجا نیز باید دقت داشت تا دستور بر روی default project صحیحی اجرا شود (حالت پیش فرض، اولین پروژه موجود در solution است).

پس از نصب آن اگر به پوشه packages مراجعه کنید، نگارش‌های مختلف آن‌را مخصوص سیلورلایت، دات نت‌های 4 و 4.5، WinRT و ویندوز فون8 نیز می‌توانید در پوشه Microsoft.AspNet.SignalR.Client ملاحظه نمائید. البته در ابتدای نصب، انتخاب نگارش مناسب، بر اساس نوع پروژه جاری به صورت خودکار صورت می‌گیرد.

مدل برنامه نویسی آن نیز بسیار شبیه است به حالت عدم استفاده از پروکسی در حین استفاده از jQuery که در قسمت قبل بررسی گردید و شامل این مراحل است:

1) یک وهله از شیء HubConnection را ایجاد کنید.

2) پروکسی مورد نیاز را جهت اتصال به Hub از طریق متد CreateProxy تهیه کنید.
3) رویدادگردان‌ها را همانند نمونه کدهای جاوا اسکریپتی قسمت قبل، توسط متد On تعریف کنید.
4) به کمک متد Start، اتصال را آغاز نمائید.
5) متدها را به کمک متد Invoke فراخوانی نمائید.

```
using System;
using Microsoft.AspNet.SignalR.Client.Hubs;

namespace SignalR02.WinClient
{
    class Program
    {
        static void Main(string[] args)
        {
            var hubConnection = new HubConnection(url: "http://localhost:1072/signalr");
            var chat = hubConnection.CreateHubProxy(hubName: "chat");

            chat.On<string>("hello", msg => {
                Console.WriteLine(msg);
            });

            hubConnection.Start().Wait();

            chat.Invoke<string>("sendMessage", "Hello!");

            Console.WriteLine("Press a key to terminate the client...");
            Console.Read();
        }
    }
}
```

نمونه‌ای از این پیاده سازی را در کدهای فوق ملاحظه می‌کنید که از لحاظ طراحی آنچنان تفاوتی با نمونه ذهنی جاوا اسکریپتی ندارد.

نکته مهم

کلیه فراخوانی‌هایی که در اینجا ملاحظه می‌کنید غیرهمزمان هستند.

به همین جهت پس از متد Start، متد Wait ذکر شده‌است تا در این برنامه ساده، پس از برقراری کامل اتصال، کار invoke صورت گیرد و یا زمانیکه callback تعریف شده توسط متد chat.On فراخوانی می‌شود نیز این فراخوانی غیرهمزمان است و خصوصا اگر نیاز است رابط کاربری برنامه را در این بین به روز کنید باید به نکات به روز رسانی رابط کاربری از طریق یک ترد دیگر دقت داشت.

نظرات خوانندگان

نویسنده: Alireza Godazchian
تاریخ: ۱۳۹۲/۰۱/۱۴ ۱۵:۲

بسیار عالی بود. متشکریم....

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۲/۱۰ ۲۳:۴۶

ممنون. آموزشهای خیلی خوبی بود
به نظرتون چطوری میشه یک ارتباط رو فقط بین دو کلاینت ایجاد کرد؟ یعنی چت فقط بین دو نفر باشه و اون دو نفر هم مشخصاتشون از روی جداول دیتابیس خونده بشه (مثلا نام کاربری و نام و نام خانوادگی اونها)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۰ ۲۳:۵۵

نیاز هست به قسمت قبل و طراحی Hub رجوع کنید. خواندن اطلاعات از بانک اطلاعاتی در Hub صورت خواهد گرفت. همچنین هر اتصالی که به سرور برقرار می‌شود دارای یک Context.ConnectionId منحصربفرد است. بر این اساس برای ارسال پیام‌ها به دو شخص خاص باید این ConnectionIdها مدیریت شوند و زمانیکه این Id را داشتید، برای انتقال پیام به او فقط کافی است در سمت هاب متد زیر را فراخوانی کنید:

```
Clients.Client(SomeId).hello(msg)
```

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۲/۰۲/۱۱ ۰:۲۱

ConnectionIdها کجا نگهداری می‌شوند؟ مثلا برای یک کاربر خاص قصد داریم تا ConnectionId او را به دست بیاوریم. مثلا بر اساس user name او

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۱ ۰:۴۷

Context.ConnectionId رو مثلا چیزی شبیه به سشن آی دی یک کاربر در ASP.NET درنظر بگیرید. دقیقا همان لحظه که به سرور و هاب متصل می‌شود، یک Context.ConnectionId منحصربفرد برای او تولید می‌شود. بر این اساس می‌شود به صورت اختصاصی به یک کاربر دسترسی یافت.
حالا در سمت کلاینت در این مثال بحث جاری پیغام سلام ارسال شده (برای توضیح مفاهیم). کاربر و کلاینت می‌تونه نام کاربری و کلمه عبور را در ابتدا به هاب ارسال کند. سپس بر این اساس سرور او را معتبر شمرده و Context.ConnectionId او را مورد پذیرش و پردازش قرار خواهد داد (یا خیر). بجای chat.server.sendMessage در مثال جاری مثلا یک chat.server.login را طراحی کنید. این متدی از Hub است که توسط کلاینت فراخوانی می‌شود. در اینجا پس از موفقیت آمیز بودن لاگین، ConnectionId او را معتبر شمرده و استفاده کنید.

نویسنده: علی نایییی
تاریخ: ۱۳۹۳/۰۲/۲۱ ۱۴:۲۳

اگه بخوایم از تو چند تا page به یه هاب وصل بشیم ، connectionIdها مدام عوض میشه. چه راه حلی برای این موضوع وجود داره؟
مثلا شما فرض کنید میخواهید در حین ورود به سیستم لیست یوزرها رو بگیرید ($.connection.hub) و یه جایی از برنامه میخواهید ورود به چت روم داشته باشید (($.connection.hub) و به این صورت آیدیها برای یه یوزر دو تا آیدی بوجود میاد ، راه

حل شما برای این مسئله چیه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۱ ۱۴:۲۵

« مدیریت نگاشت ConnectionIdها در SignalR به کاربران واقعی سیستم »

نویسنده: غلامرضا ربال
تاریخ: ۱۳۹۴/۰۴/۲۰ ۰:۳

تمرین 1

به پروژه ساده و ابتدایی فوق یک تکست باکس دیگر به نام Room را اضافه کنید؛ به همراه دکمه join. سپس نکات قسمت قبل را در مورد الحاق به یک گروه و سپس ارسال پیام به اعضای گروه را پیاده سازی نمائید. (تمام نکات آن با مطلب فوق پوشش داده شده است و در اینجا باید صرفا فراخوانی متدهای عمومی دیگری در سمت هاب، صورت گیرد) یک چت گروهی ساده با توجه به مطالب گفته شده