

|  |
| --- |
| Oefening signalr  Ouassim Abdellaoui & Christophe Mathieu  Graduaat Informatica Programmeren  Webapplicaties |

1. Oefening signalr

Opgave

In deze oefening wordt stap voor stap een chat webapplicatie gebouwd met behulp van ASP.NET SignalR. We voegen SignalR toe aan een MVC webapplicatie en maken een chatweergave om berichten te verzenden en weer te geven.

Het gewenste eindresultaat is als volgt:

Graphical user interface, text, application

Description automatically generated

Vooraf

Om het project te bouwen moeten we als eerst een lege ASP.NET MVC applicatie aanmaken. Daarna voegen wij de SignalR library toe en bouwen wij uiteindelijk de chat webapplicatie.

1. Open Visual Studio en kies voor **C# ASP.NET Web Application**. Klik op **Next**.

Graphical user interface, text

Description automatically generated

1. Noem de applicatie ***SignalRChatApplication*** en deze moet gericht zijn op een **.NET framework 4.5 of hoger**. Klik dan op **Create**.

Graphical user interface, text, application, Teams

Description automatically generated

1. In **Create a new ASP.NET Web Application**, selecteer **MVC** en kies voor **No Authentication**. Klik daarna op **Create**.

Graphical user interface, text

Description automatically generated

ChatHub

We beginnen met het toevoegen van de ChatHub class. Je kan de ChatHub het best vergelijken met een server. Deze gaat luisteren naar berichten met behulp van een methode en deze doorsturen naar alle clienten. Voor het aanmaken van de ChatHub class volgen we de onderstaande stappen:

1. Klik in **Solution Explorer** met de rechtermuisknop op het project en selecteer **Add** > **New item**.

Graphical user interface, text, website

Description automatically generated

1. In window Add New Item, ga naar **Visual C#** > **Web** > **SignalR** en selecteer vervolgens **SignalR Hub Class (v2)**. Noem de class ***ChatHub*** en klik op **Add**.

A screenshot of a computer

Description automatically generated with medium confidence

Deze stap maakt de class ChatHub.cs aan en voegt een set scriptbestanden en assembly-referenties toe die SignalR ondersteunen aan het project.

1. Vervang de code in ChatHub.cs door deze code:

A screenshot of a computer

Description automatically generated with medium confidence

De methode SendMessage() staat in voor het versturen van berichten. Als er iemand de SendMessage methode oproept gaan we een gebruiker en bericht meegeven. Wanneer dit gebeurt kijkt de methode welke gebruikers momenteel verbonden zijn en krijgen ze allemaal dit bericht met behulp van de .addNewMessageToPage().

Startup.cs

1. Klik in **Solution Explorer** met de rechtermuisknop op het project en selecteer **Add** > **Class**.
2. Noem de nieuwe class ***Startup*** en voeg deze toe aan het project.
3. Vervang de code in de class Startup.cs door deze code:

Text

Description automatically generated

HomeController.cs

1. Selecteer in Solution Explorer **Controllers** > **HomeController.cs**.
2. Voeg de volgende methode toe aan HomeController.cs.

Text

Description automatically generated

Deze methode geeft weer de chatweergave die wij in de volgende stap maken.

Indexpagina

Nu gaan we onze viewpagina aanmaken.

1. Klik in **Solution Explorer** met de rechtermuisknop op **Views** > **Home** en selecteer **Add** > **View**.
2. Geef in **Add View** de nieuwe weergave de naam ***Chat*** en selecteer **Add**.

Text

Description automatically generated

1. Vervang de inhoud van Chat.cshtml door deze code:

@{

ViewBag.Title = "Chat";

}

<h2>Chat</h2>

<div class="container">

<input type="text" id="message" />

<input type="button" id="sendmessage" value="Send" />

<input type="hidden" id="displayname" />

<ul id="discussion">

</ul>

</div>

@section scripts {

<!--Script references. -->

<!--The jQuery library is required and is referenced by default in \_Layout.cshtml. -->

<!--Reference the SignalR library. -->

<script src="~/Scripts/jquery.signalR-2.2.2.min.js"></script>

<!--Reference the autogenerated SignalR hub script. -->

<script src="~/signalr/hubs"></script>

<!--SignalR script to update the chat page and send messages.-->

<script>

$(function () {

// Reference the auto-generated proxy for the hub.

var chat = $.connection.chatHub;

// Create a function that the hub can call back to display messages.

chat.client.addNewMessageToPage = function (name, message) {

// Add the message to the page.

$('#discussion').append('<li><strong>' + htmlEncode(name)

+ '</strong>: ' + htmlEncode(message) + '</li>');

};

// Get the user name and store it to prepend to messages.

$('#displayname').val(prompt('Enter your name:', ''));

// Set initial focus to message input box.

$('#message').focus();

// Start the connection.

$.connection.hub.start().done(function () {

$('#sendmessage').click(function () {

// Call the Send method on the hub.

chat.server.send($('#displayname').val(), $('#message').val());

// Clear text box and reset focus for next comment.

$('#message').val('').focus();

});

});

});

// This optional function html-encodes messages for display in the page.

function htmlEncode(value) {

var encodedValue = $('<div />').text(value).html();

return encodedValue;

}

</script>

}

Check scripts

Scriptbibliotheken voor jQuery en SignalR zijn zichtbaar in het project. Het is namelijk zo dat de geïnstalleerde jQuery scripts wel van minimaal versie 2.1.0 moeten zijn, aangezien wij .NET framework 4.5 of hoger hanteren. Via de volgende stappen kunnen wij het versienummer checken.

1. Vouw in Solution Explorer het mapje **Scripts** uit. Als het goed is zien wij vervolgens het volgende:

Text

Description automatically generated

1. Controleer of de scriptverwijzingen in de view Chat.cshtml overeenkomen met de versies van de scriptbestanden in het project. Het is mogelijk dat de Package Manager een vorige of latere versie van de SignalR-scripts geïnstalleerd heeft.

Text

Description automatically generated

Mochten ze niet matchen, dan moet het .cshtml file aangepast worden.

1. Klik daarna op **File** > **Save All**

Applicatie runnen

De chatapplicatie is nu klaar en deze kunnen wij runnen om te kijken naar de werking hiervan.

1. Klik op F5 om de applicatie te runnen.
2. Vul in je naam. Hierbij een voorbeeld:

Graphical user interface

Description automatically generated

1. Je krijgt de applicatie te zien. Voordat je hiermee aan de slag gaat, kopieer de url en plak deze in een nieuwe browser.

Graphical user interface, text, application

Description automatically generated

1. In de nieuwe window moet je een andere naam invullen.

Graphical user interface

Description automatically generated with medium confidence

1. Voeg nu een opmerking toe en selecteer **Send**. Herhaal dat in de andere browser. De berichten verschijnen in realtime.

Graphical user interface, text, application, website

Description automatically generated

Alhoewel de berichten in realtime verschijnen, is het namelijk zo dat deze eenvoudige chattoepassing de discussiecontext niet op de server onderhoudt. De hub zendt opmerkingen uit naar alle huidige gebruikers. Gebruikers die later deelnemen aan de chat zien berichten die zijn toegevoegd VANAF het moment dat ze deelnemen.

Check in Solution Explorer in folder **Scripts** voor de actieve toepassing. Er is een scriptbestand met de naam **hubs** dat de SignalR-bibliotheek tijdens runtime genereert. Dit bestand beheert de communicatie tussen jQuery-script en server-side code.