**Exercise: SignalR**

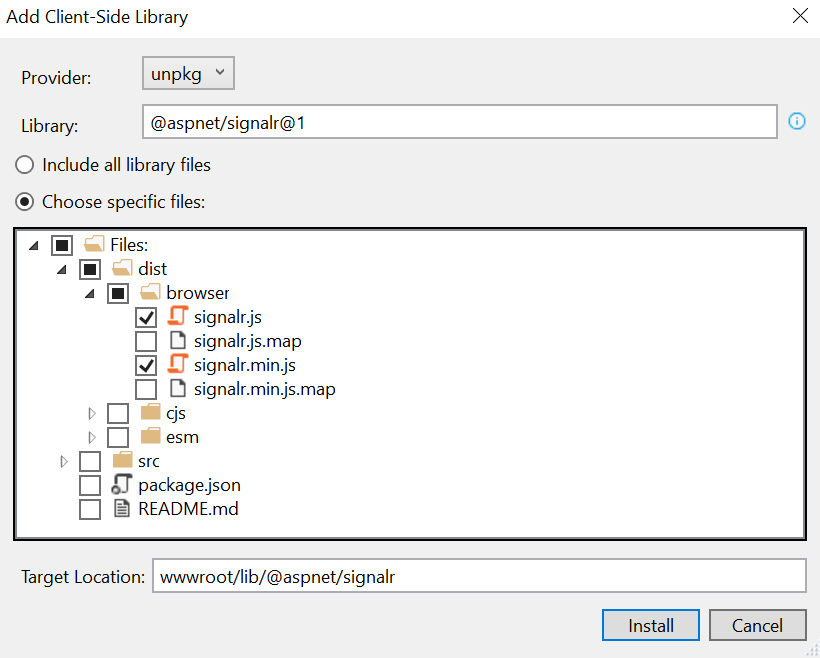
Problems for exercises and homework for the [“ASP.NET Core” course @ SoftUni](https://softuni.bg/trainings/2354/aspnet-core-july-2019).

## Chat

Let’s create a simple **Chat** application in **ASP.NET Core**. The app will let users type a username and send a message.

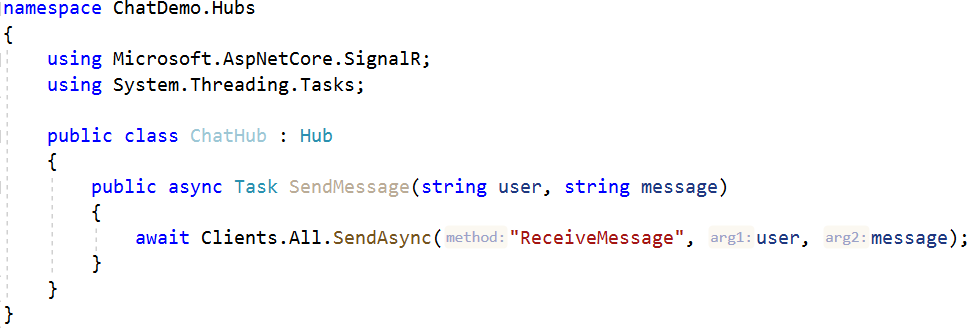
## Create Project and include SignalR

Create new Solution and add new **ASP.NET Core Web Application**. In **Solution Explorer**, right-click on the project, **Add -> Client-Side Library**.



## Create Hub class

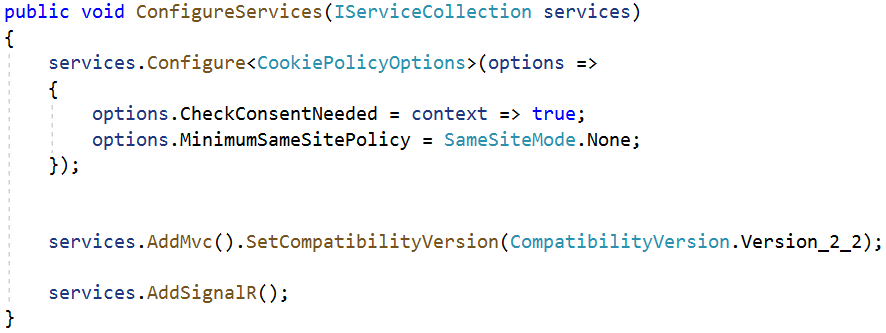
Create a **Hubs** folder in the project and create a **ChatHub.cs** class file.



The **ChatHub** class inherits from the **SignalR Hub** class. The **Hub** class manages connections, groups and messaging. The **SendMessage** method sends the received message to all clients and can be called by any connected client.

## Configuring SignalR

Configuring **SignalR** in our application is pretty simple. Go to **Startup.cs**:



And in **Configure()** method we configure the endpoint:



## View

Not it’s time for the client-side. Let’s replace the content of **Index.cshtml**.



## JavaScript code

Create a **chat.js** file in the **wwwroot/js** folder. First we initialize a **SignalR connection**. When the back-end calls the method for **receiving a message**, a function will run for **showing** the sent content on our page. We also have an **error handler** for catching errors if something is wrong during the connection. Last, we have a function called when we click on **Send** button. The client-side calls the back-end method for **sending a message** and takes the **input values.**



## Test the Application

Now we are ready to test our application. Let’s **run the app** and open **2 browser instances** on the **URL** of the app. Enter a **Username** and a **Message** on one of the browsers and send it. Check the other one.

