# Лабораторная работа №28 Обмен данными

1. Цель работы
   1. Изучить процесс разработки сетевых приложений с использованием SignalR на C#;
2. Литература
   1. Основы SignalR // Metanit.com – URL: https://metanit.com/sharp/signalr/1.12.php. – Режим доступа: свободный. – Текст : электронный.
3. Подготовка к работе
   1. Повторить теоретический материал (см. п.2).
   2. Изучить описание лабораторной работы.
4. Основное оборудование
   1. Персональный компьютер.
5. Задание
   1. Создание сервера для чата на SignalR
      1. Создать новый проект ASP.Net minimal api.
      2. Подключить в файле Program пакет Microsoft.AspNetCore.SignalR
      3. Зарегистрировать в сервисах SignalR

builder.Services.AddSignalR();

* + 1. Создать класс ChatHub – наследник класса Hub, в котором реализовать асинхронный метод для ретрансляции сообщений, полученных от пользователя другим пользователям с помощью вызова удаленного callback-метода.

await Clients.All.SendAsync(имя\_callback, отправляемые\_данные);

* + 1. Создать endpoint для ChatHub при помощи

app.MapHub<ChatHub>("/адрес");

* 1. Создание клиента для чата на SignalR
     1. Создать новый проект консольного приложения
     2. Установить в проекте пакет Microsoft.AspNetCore.SignalR.Client
     3. Установить подключение к серверу SignalR при помощи

var connection = new HubConnectionBuilder()

.WithUrl("адрес hub")

.Build();

* + 1. Настроить callback для получения сообщения от сервера исходя из данных, отправляемых в п.5.1.4

connection.On<типы\_получаемых\_данных>(имя\_callback, (получаемые\_данные) =>

{

//Код для вывода полученного сообщения

});

* + 1. Запустить соединение к серверу

await connection.StartAsync();

* + 1. Далее необходимо бесконечно запрашивать у пользователя сообщения и отправлять их на сервер при помощи

await connection.InvokeAsync(имя\_метода\_обработки\_сообщений\_hub, отправляемые\_данные);

* 1. Реализация комнат-чатов
     1. Добавить в класс ChatHub два поля типа ConcurrentDictionary<string, string> для хранения данных о именах пользователей и подключениях к комнатам.
     2. Реализовать метод JoinRoom(string roomName, string userName), в котором необходимо сохранить данные пользователя в соответствующие словари

// Получаем соединение

var connectionId = Context.ConnectionId;

// Привязать соединение к комнате и пользователю

ConnectionToRoom[connectionId] = roomName;

ConnectionToUser[connectionId] = userName;

// Добавляем соединение в группу по имени комнаты

await Groups.AddToGroupAsync(connectionId, roomName);

* + 1. Далее при обработке сообщений на сервере можно использовать

// Проверка наличия имени пользователя и комнаты в ConcurrentDictionary

if (ConnectionToRoom.TryGetValue(connectionId, out var room) &&

ConnectionToUser.TryGetValue(connectionId, out var user))

{

// Отправка данных клиентам в определенной группе

await Clients.Group(room).SendAsync("ReceiveMessage", user, message);

}

* + 1. Необходимо обработать отключение пользователя от группы при разрыве соединения с клиентом, для этого необходимо переопределить стандартный метод OnDisconnectedAsync и использовать в нем

await Groups.RemoveFromGroupAsync(connectionId, room);

1. Порядок выполнения работы
   1. Запустить MS Visual Studio и создать оконное приложение C#.
   2. Выполнить все задания из п.5 в одном решении.
   3. Ответить на контрольные вопросы.
2. Содержание отчета
   1. Титульный лист
   2. Цель работы
   3. Ответы на контрольные вопросы
   4. Вывод
3. Контрольные вопросы
   1. Что такое SignalR?
   2. Каким образом осуществляется обмен данными между клиентом и сервером SignalR?