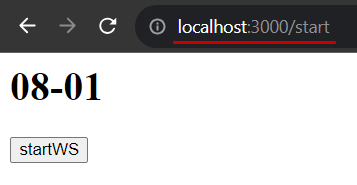
Лабораторная работа 08

ПСКП

**Задание 01**

1. Разработайте приложение **08-01**, представляющее собой HTTP-сервер и WebSocket(WS)-сервер одновременно.
2. HTTP-сервер должен прослушивать порт **3000**.
3. WS-север должен прослушивать порт **4000**.
4. HTTP-сервер реагирует только на GET-запрос **/start**, на остальные запросы HTTP-сервер должен возвращать ответ со статусом **400**.
5. На GET-запрос **/start** HTTP-сервер возвращает HTML-страницу следующего вида:



1. При нажатии кнопки **startWS** должна запускаться JS-функция, отправляющая каждые 3 сек. сообщения WS-серверу следующего вида:

**08-01-client: *n***, где ***n*** – последовательный номер отправляемого серверу сообщения.

1. JS-функция должна отображать сообщения принятые от сервера на странице.
2. JS-функция должна автоматически остановить передачу сообщений через 25 сек и закрыть WS-соединение.
3. WS-сервер принимает сообщения от клиента.
4. WS-сервер отображает принятые сообщения от клиента.
5. WS-сервер каждые 5 сек. отправляет клиенту сообщения следующего вида:

**08-01-server: *n->k***, где ***n*** – номер из последнего сообщения клиента, а ***k*** - последовательный номер отправляемого клиенту сообщения.

1. Проверьте работоспособность приложения. Объясните эффект, наблюдаемый при многократном нажатии на кнопку **startWS.**

**Задание 02**

1. Разработайте приложение **08-02**, представляющее собой WebSocket(WS)-клиент.
2. Приложение **08-02** должно быть функционально аналогично JS-функции, разработанной в первом задании.
3. Проверьте работоспособность приложения совместно с сервером, разработанным в первом задании.
4. Запустите несколько экземпляров приложения **08-02** для взаимодействия с одним общим сервером. Объясните наблюдаемый эффект.

**Задание 03**

1. Разработайте приложение **08-03**, представляющее собой широковещательный WebSocket-сервер.
2. Разработайте приложение **08-03a** - WebSocket-клиент.
3. Продемонстрируйте работоспособность широковещательного WebSocket-сервера **08-03,** взаимодействующего с тремя клиентами **08-03a.**

**Задание 04** Ответьте на следующие вопросы.

1. Поясните разницу между полудуплексным и дуплексным каналами данными.

Полудуплексный и дуплексный каналы данных относятся к способу передачи информации между двумя устройствами или системами. Основное отличие между ними заключается в возможности одновременной двусторонней передачи данных.

Полудуплексный канал данных: В полудуплексной связи передача данных возможна только в одном направлении за раз. То есть, устройство может либо отправлять данные, либо принимать данные, но не может выполнять оба действия одновременно.

Дуплексный канал данных: В дуплексной связи передача данных возможна одновременно в обоих направлениях. Оба устройства могут отправлять и принимать данные одновременно. Такой тип связи позволяет более эффективное использование канала передачи данных и обеспечивает более высокую пропускную способность.

1. Какой тип канала применяется HTTP-протоколом.

HTTP-протокол использует дуплексный канал данных для связи между клиентом и сервером.

1. Поясните понятие **WebSocket**.

WebSocket - это протокол связи, который обеспечивает постоянное двустороннее (дуплексное) соединение между клиентом и сервером через единственный TCP-сокет.

1. Какой тип канала применяется WebSocket-протоколом.

WebSocket-протокол использует дуплексный канал данных для связи между клиентом и сервером.

1. Расскажите процедуру WebSocket-рукопожатия.

Процедура WebSocket-рукопожатия, также известная как WebSocket handshake, является первоначальным этапом установки соединения между клиентом и сервером при использовании WebSocket-протокола. Вот общая последовательность шагов этой процедуры:

1. Клиент отправляет HTTP-запрос со специальными заголовками на сервер, указывая желание установить WebSocket-соединение. Запрос содержит следующие заголовки:

* Upgrade: Значение этого заголовка должно быть "websocket", указывая, что клиент хочет установить WebSocket-соединение.
* Connection: Значение этого заголовка должно содержать "Upgrade", чтобы указать, что соединение должно быть обновлено до WebSocket.
* Sec-WebSocket-Key: Это случайно сгенерированное значение, которое используется для обеспечения безопасности рукопожатия.

1. Сервер получает этот запрос и выполняет проверку наличия заголовков. Если все заголовки корректны, сервер готов принять запрос на установку WebSocket-соединения.
2. Сервер отправляет HTTP-ответ клиенту, подтверждая рукопожатие. Ответ содержит следующие заголовки:

* Upgrade: Значение этого заголовка также должно быть "websocket", указывая на обновление соединения до WebSocket.
* Connection: Значение этого заголовка должно содержать "Upgrade", чтобы указать, что соединение было обновлено до WebSocket.
* Sec-WebSocket-Accept: Значение этого заголовка является хэш-кодом, рассчитанным на основе значения Sec-WebSocket-Key, предоставленного клиентом. Это подтверждение валидности рукопожатия.

1. Клиент получает этот ответ и выполняет проверку заголовков. Если заголовки корректны, клиент знает, что WebSocket-соединение было успешно установлено.
2. Поясните понятия «широковещательное сообщение» (broadcast), «широковещательный сервер».

Широковещательное сообщение (broadcast) - это сообщение, которое отправляется от одного отправителя и получается множеством получателей, расположенных в одной сети или в пределах определенной группы.

Широковещательный сервер (broadcast server) - это сервер, который передает широковещательные сообщения по сети. Он играет роль посредника, который получает сообщение от отправителя и перенаправляет его всем получателям, находящимся в той же сети или в той же группе.