HTML - WebSockets

WebSockets is a next-generation bidirectional communication technology for web applications that operates over a single socket.

WebSockets allow bidirectional communication, which means both client and server can send data to each other independently and simultaneously.

After establishing a Web Socket connection with the web server, we can send data from browser to server by calling the **`send()`** method and receive data from server to browser by an **`onmessage`** event handler.

Syntax

Following is the API, which creates a new WebSocket object:

```
var Socket = new WebSocket(url, [protocol] );
```

Here the first argument, **`url`**, specifies the URL to which to connect. The second attribute, **`protocol`**, is optional and, if present, specifies a sub-protocol that the server must support for the connection to be successful.

Attributes of WebSocket

Following are the attributes of the WebSocket object. Assuming we created a socket object as mentioned above:

Attribute	Description
Socket.readyState	The readonly attribute readyState represents the state of the connection. It can have the following values: ■ A value of 0 indicates that the connection has not yet been established. ■ A value of 1 indicates that the connection is established and communication is possible. ■ A value of 2 indicates that the connection is going through the closing handshake. ■ A value of 3 indicates that the connection has been closed or could not be opened.
Socket.bufferedAmount	The readonly attribute bufferedAmount represents the number of bytes of UTF-8 text that have been queued using the **`send()`** method.

WebSocket Events

Following are the events associated with the WebSocket object. Assuming we created a socket object as mentioned above:

Event	Values & Event Handler	Values & Description
open	Socket.onopen	This event occurs when socket connection is established.
message	Socket.onmessage	This event occurs when client receives data from server.
error	Socket.onerror	This event occurs when there is any error in communication.
close	Socket.onclose	This event occurs when connection is closed.

WebSocket Methods

Following are the methods associated with the WebSocket object. Assuming we created a socket object as mentioned above:

Method	Description
Socket.send()	The send(data) method transmits data using the connection.
Socket.close()	The close() method would be used to terminate any existing connection.

Setting Up the WebSocket Server with Python

Step 1. Install Python
If you don't have Python installed on your device, download and install it from Python.org

Step 2. Install WebSocket library
After installing python create a folder for your project, and open that folder in the command prompt or terminal. Then run this prompt.
pip install websockets

Step 3. Create the websocket server
Open any text editor and write the below Python code. Then save that as a file in the folder with the name 'server.py'

```
import asyncio
import websockets

async def echo(websocket, path):
```

```
    async for message in websocket:
        print(f"Received message: {message}")
        await websocket.send(f"Server: You said \"{message}\"")

start_server = websockets.serve(echo, "localhost", 8080)

asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

Step 4. Run the server
In the terminal navigate to your project folder, and run this command to start server.
python server.py

Setting up HTML Client for the Server

So far we setup a Python server for websocket. The server will be running on your terminal, so any messages sent to the server will be visible at the terminal. Here we will see how to setup a client that can receive messages from the server and also send messages to the server using HTML and JavaScript.

Create an HTML file 'index.html' in the folder.

</> Open Compiler

```
<!DOCTYPE html>
<html lang="en">

<head>
   <title>WebSocket Example</title>
</head>

<body>
   <h1>WebSocket Client</h1>
   <input type="text"
          id="messageInput"
          placeholder="Type a message..." />
   <button id="sendButton">
      Send
   </button>
   <div id="messages">
   </div>

   <script>
      const socket = new WebSocket('ws://localhost:8080');

      socket.addEventListener('open',
```

```
    () => {
    console.log('Connected to server');
    });

    socket.addEventListener('message',
    (event) => {
    const messageDiv = document.createElement('div');
    messageDiv.textContent = event.data;
    document.getElementById('messages').appendChild(messageDiv);
    });

    document.getElementById('sendButton').addEventListener('click',
    () => {
    const messageInput = document.getElementById('messageInput');
    const message = messageInput.value;
    socket.send(message);
    messageInput.value = '';
    });
  </script>
</body>
</html>
```