What Are WebSockets?

If you've ever used a chat app, seen live sports scores update instantly, or watched a live trading platform, you've already experienced what WebSockets can do.

But **what are WebSockets**? And how can you use them in your applications?

Let’s break it down in a way that’s easy to understand and get you started with real examples!

What Is a WebSocket?

In simple terms, **WebSockets** provide a way to **open a persistent, two-way connection** between the client (browser) and server.

Why WebSockets?

- Traditional HTTP is **request-response based** (client asks, server replies).
- WebSockets keep the connection **open**, allowing **real-time communication** in both directions.

Think of it like a phone call instead of sending letters back and forth.

When Should You Use WebSockets?

Use WebSockets when you need **real-time communication**, like:

- Chat applications
- Live notifications
- Multiplayer games
- Live dashboards

WebSockets in Node.js — Let’s Build One!

We’ll use the popular `ws` package to create a simple WebSocket server and connect a client to it.

Step 1: Set Up Your Project

```
mkdir websocket-demo
cd websocket-demo
npm init -y
npm install ws
```

Step 2: Create a Simple WebSocket Server

Create a file called `server.js`:

```
const WebSocket = require('ws');

const wss = new WebSocket.Server({ port: 8080 });

wss.on('connection', function connection(ws) {
  console.log('A new client connected!');

  ws.send('Welcome to the WebSocket server!');

  ws.on('message', function incoming(message) {
    console.log(`Received: ${message}`);

    // Echo the message back to the client
    ws.send(`You said: ${message}`);
  });

  ws.on('close', () => {
    console.log('A client disconnected');
  });
});

console.log('WebSocket server is running on ws://localhost:8080');
```

Step 3: Create a WebSocket Client (HTML Page)

Create a file called `index.html`:

```
<!DOCTYPE html>
<html>
<head>
  <title>WebSocket Chat</title>
</head>
<body>
  <h2>WebSocket Chat</h2>
  <ul></ul>
  <input type="text" placeholder="Type message..." />
  <button>Send</button>

  <script>
      const socket = new WebSocket('ws://localhost:3000');

      // Listen for messages
      socket.onmessage = ({ data }) => {
          const el = document.createElement('li');
          el.innerHTML = event.data;
          document.querySelector('ul').appendChild(el);
      };

      document.querySelector('button').onclick = () => {
          const text = document.querySelector('input').value;
          if (text && socket.readyState === WebSocket.OPEN) {
              socket.send(text);
              document.querySelector('input').value = '';
          }
      };
  </script>
</body>
</html>
```

Open this file in your browser and send messages!

How It Works

1. The browser connects to `ws://localhost:3000`.

2. The server accepts the connection and sends a welcome message.
3. When the user sends a message, it goes to the server.
4. The server echoes the message back to the client.

Just like chatting in real time — no page reloads, no waiting!

Testing Your WebSocket App

1. Start the server:

```
node server.js
```

1. Open `index.html` in your browser.
2. Type a message and click "Send" — you'll see the server respond instantly.

Final Thoughts

WebSockets are a **powerful tool** for adding real-time features to your applications. With just a few lines of code in Node.js, you can build responsive, interactive apps that feel alive.