Advanced Web Lab (Websockets)

Explore and Observe

Use the Console to observe how the readyState property changes during each stage of a WebSocket connection.

Before Opening the Connection

Open the Console and type:
socket.readyState

1. What number do you see? What does that value represent in the WebSocket lifecycle?

Answer: there was no number, I got an error

Immediately After Clicking “Open Connection”

Type again:
socket.readyState

2. What value do you see now? Which constant does it match (CONNECTING, OPEN, CLOSING, or CLOSED)?

Answer: I got a number of 0, assuming that is for connecting.

When the “Status: Connected ✅” Appears

Check socket.readyState again.

3. What value is displayed? What does that tell you about the connection’s ability to send and receive data?

Answer: constant of 1 for connected/open.

While the Connection Is Still Open

Type several messages using the Send button.

4. Does socket.readyState change during normal message exchange? Why or why not?

Answer: It does not change

After Clicking “Close Connection”

Check socket.readyState immediately after you click Close.

5. What value do you see while the connection is shutting down? What value do you see a few seconds later, after it fully closes?

Answer: There are two different numbers, there is a 2 (when the connection is closing), and a constant of 3 after t has closed.

Reflection Questions

1. How is a WebSocket connection different from an HTTP request?
(Think about persistence, who initiates communication, and when the connection closes.)

Answer: An HTTP request is just a normal "I send a request, I receive a response", then the connection is closed. On the other hand, websockets are persistent and bidirectional, this means the connection is not closed after the first connection, the websocket closes only if the client or the server decide to close it.

---

2. Why are WebSockets ideal for real-time applications such as chats, dashboards, or multiplayer games?
(Explain how the connection model supports continuous updates.)

Answer: Websockets are ideal for real-time applications because they allow for continuous updates. For example in a chat application, if a message is sent the person who its sent to immediately receives it.
Websockets also provide a reduced latency because a connection is not opened and closed for every update but the first connection is used for all future updates.

---

3. Why is WebSocket communication considered stateful?
(Relate this to how readyState and event handlers maintain ongoing connection context.)

Answer: They are considered stateful because the client and the server have information about the connection's state. these states, or readyState as I used above are CONNECTING, OPEN, CLOSING, and CLOSED.

---

4. What do the readyState values reveal about the lifecycle of a WebSocket?
(How does each state—CONNECTING, OPEN, CLOSING, CLOSED—help a developer manage program logic?)

Answer: CONNECTING (0) means the application is connecting to the websocket but not yet connected. Once connected it moves to) OPEN (1) which means connected and is ready for data transfer. When closing the state goes to CLOSING(2) which the websocket shutting down. After shutting down it changes to CLOSED(3) which means the connection has been terminated.

---

5. If two users connect to the same WebSocket server at the same time, how might each user's readyState differ during setup or closing?
(Think about concurrency and timing differences between independent client sessions.)

Answer: If two user connect to the same websocket their states will differ if the connect at a different time, and also because each websocket connection is a session on its own, not the same as the other user.