B.Sc. In Software Development. Year 4. Enterprise Development. Semester I. Web Sockets

LIMERICK INSTITUTE OF TECHNOLOGY
SCHOOL OF SCIENCE, ENGINEERING & I.T.
Department of Information Technology

What is a web socket?

- Communication protocol used to send and receive data.
  - Like HTTP but more efficient.

Http	Web Socket
Half duplex (like a walkie-talkie)	Full duplex (like a phone)
Traffic flows in one direction	Bi-directional
Connection typically closes after 1 req/resp pair	Connection stays open
Req from client – server Resp from server – client	Client and server are simultaneously emitting and listening (for events)
150ms to establish each TCP connection for each HTTP message	50ms for message transmission.

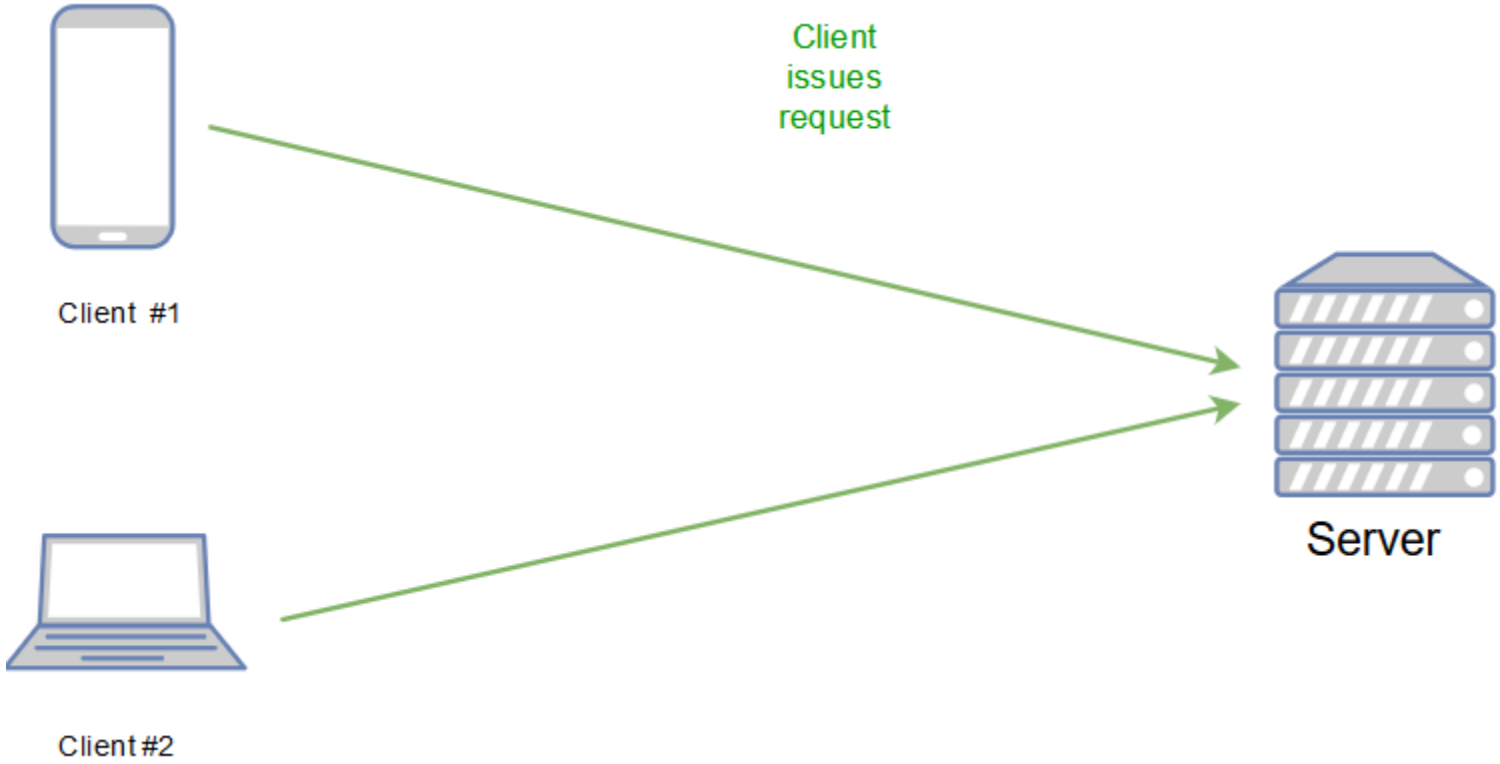
What is a web socket?

- Easy to build real-time applications.
  - Chat.
  - Notifications.
  - Online games.
  - Live maps.
  - Collaboration apps.

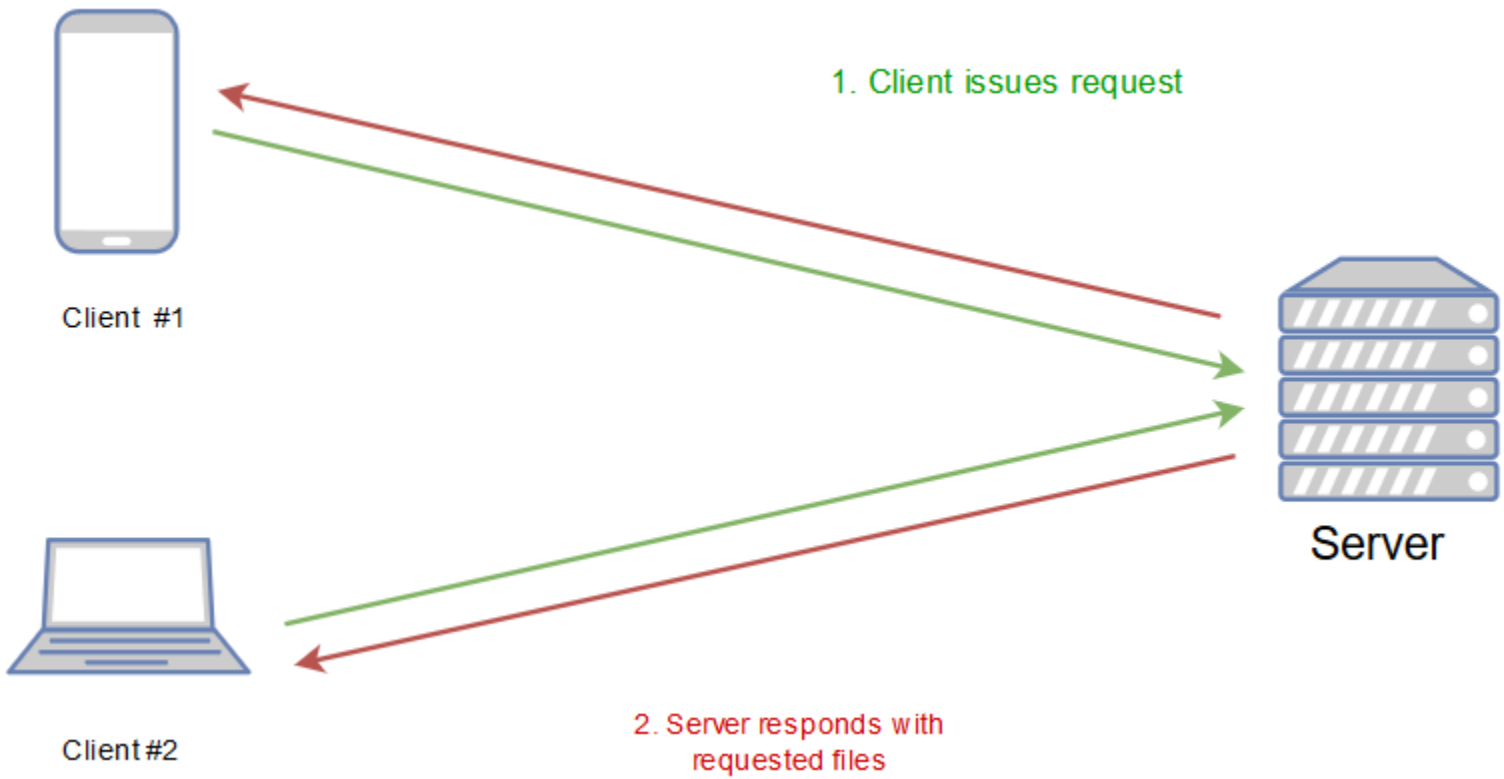
How do web sockets work?

- To establish a WebSocket connection, the client opens a connection and sends an opening handshake request to the server in the form of an HTTP upgraded header
- The client than waits for the response from the server.
- Upon receiving the handshake request, the server parses the request header to obtain necessary information. This information is used to frame the handshake response in the form of another HTTP header. If the handshake is successful, the server opens the connection to accept incoming data.
- Because the client had been waiting for the server's response, upon receiving it, it confirms the handshake and a constant connection is established for message transmission.
- The connection remains open for communication until explicitly closed. During the open phase, the client/server can send/receive messages at will.

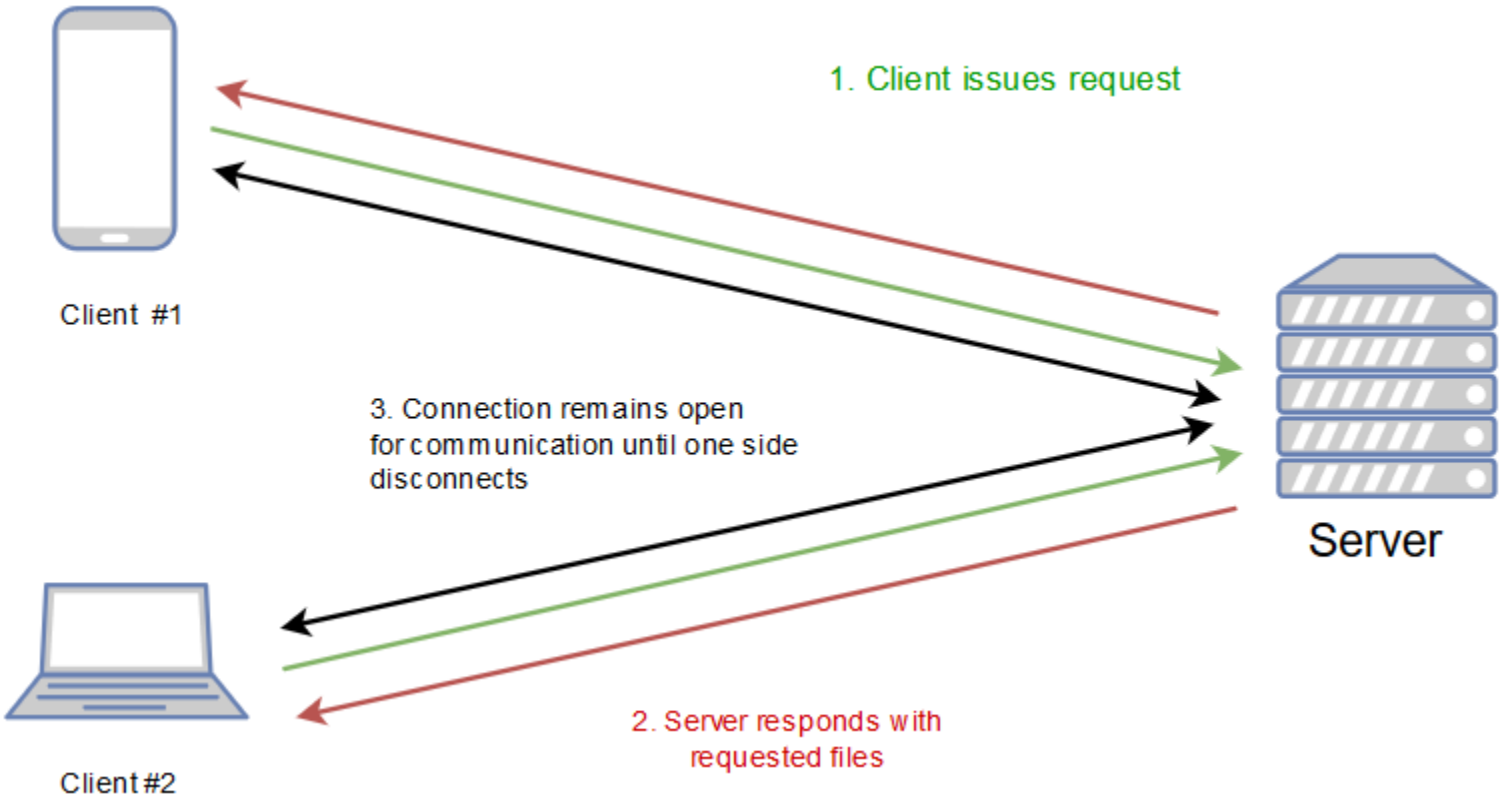
What is a web socket?



What is a web socket?



What is a web socket?



On the client side

- WebSocket API.

```
<script>
        var webSocket = new WebSocket("ws://localhost:8080/ChatWebSocket/endpoint");

        webSocket.onmessage = function (message) {
            alert(message);
        }

        webSocket.onopen = function () {
            alert("connection opened");
        };

        webSocket.onclose = function () {
            alert("connection closed");
        };

        webSocket.onerror = function wserror(message) {
            alert("error: " + message);
        }

</script>
```

On the server side

- WebSocket endpoint lifecycle events are handled by the following annotations.

  @ServerEndpoint: If decorated with *@ServerEndpoint,* the container ensures availability of the class as a *WebSocket* server listening to a specific URI space

  @OnOpen: Is invoked by the container when a new *WebSocket* connection is initiated

  @OnMessage: A Java method to receive the information from the *WebSocket* container when a message is sent to the endpoint

  @OnError: is invoked when there is a problem with the communication

  @OnClose: is called by the container when the *WebSocket* connection closes

On the server side

```
10  @ServerEndpoint("/uri-goes-here")
11  public class MySocketEndpoint {
12
13      public MySocketEndpoint() { }
14
15      @OnOpen
16      public void onOpen(Session session) {
17          // Get session and WebSocket connection
18      }
19
20      @OnMessage
21      public void onMessage(String message, Session session) {
22          // Handle new messages
23      }
24
25      @OnError
26      public void onError(Throwable e) {
27          //Handle errors here
28      }
29
30      @OnClose
31      public void onClose(Session session) {
32          // WebSocket connection closes
33      }
34  }//end class
```

Example 1

localhost:8080/ChatWebSocket

localhost:8080/ChatWebSocket/chat_to_server.html

send

Server> : Connection Successful
Client> Hi Server
Server> : We received your message: Hi Server
Client> What time is it where you are?
Server> : We received your message: What time is it where you are?
Client> This is a bit of an echo chamber
Server> : We received your message: This is a bit of an echo chamber

Output

ChatWebSocket (run) | Apache Tomcat9 Log | Apache Tomcat9

```
Class loaded class sd4.com.src.ChatEndpoint
Session opened, id: cMessage received. Session id: c Message: Hi Server
Message received. Session id: c Message: What time is it where you are?
Message received. Session id: c Message: This is a bit of an echo chamber
Session closed with id: c
```

Example 1

ChatEndpoint.java

Source History

```java
@ServerEndpoint("/chat")
public class ChatEndpoint {

    public ChatEndpoint() {
        System.out.println("Class loaded " + this.getClass());
    }

    @OnOpen
    public void onOpen(Session session) {
        System.out.printf("Session opened, id: " + session.getId());
        try {
            session.getBasicRemote().sendText("Connection Successful");
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }

```

Example 1

```
28      @OnMessage
29      public void onMessage(String message, Session session) {
30          System.out.printf("Message received. Session id: %s Message: %s%n",
31                  session.getId(), message);
32          try {
33              session.getBasicRemote().sendText(String.format("We received your message: %s%n", message));
34          } catch (IOException ex) {
            ex.printStackTrace();
36          }
37      }
38
39      @OnError
40      public void onError(Throwable e) {
            e.printStackTrace();
42      }
43
44      @OnClose
45      public void onClose(Session session) {
46          System.out.printf("Session closed with id: %s%n", session.getId());
47      }
48  }//end class
```

Example 1

chat_to_server.html

Source History

```
3   <body style="margin: 35px">
4   <form>
5       <input id="messageField" type="text">
6       <input onclick="sendMsg();" value="send" type="button">
7   </form>
8
9   <div id="msg-box" style="width:500px; height: 400px; background: #eee; overflow:auto;"></div>
10
11  <script>
12      var webSocket = new WebSocket("ws://localhost:8080/ChatWebSocket/chat");
13      var msgField = document.getEl`ementById("messageField");
14      var divMsg = document.getElementById("msg-box");
15
16      function sendMsg() {
17          var msgToSend = msgField.value;
18          webSocket.send(msgToSend);
19          divMsg.innerHTML += "<div style='color:red'>Client> " + msgToSend +
20                                  "</div>"
21          msgField.value = "";
22      } //end sendMsg()
```

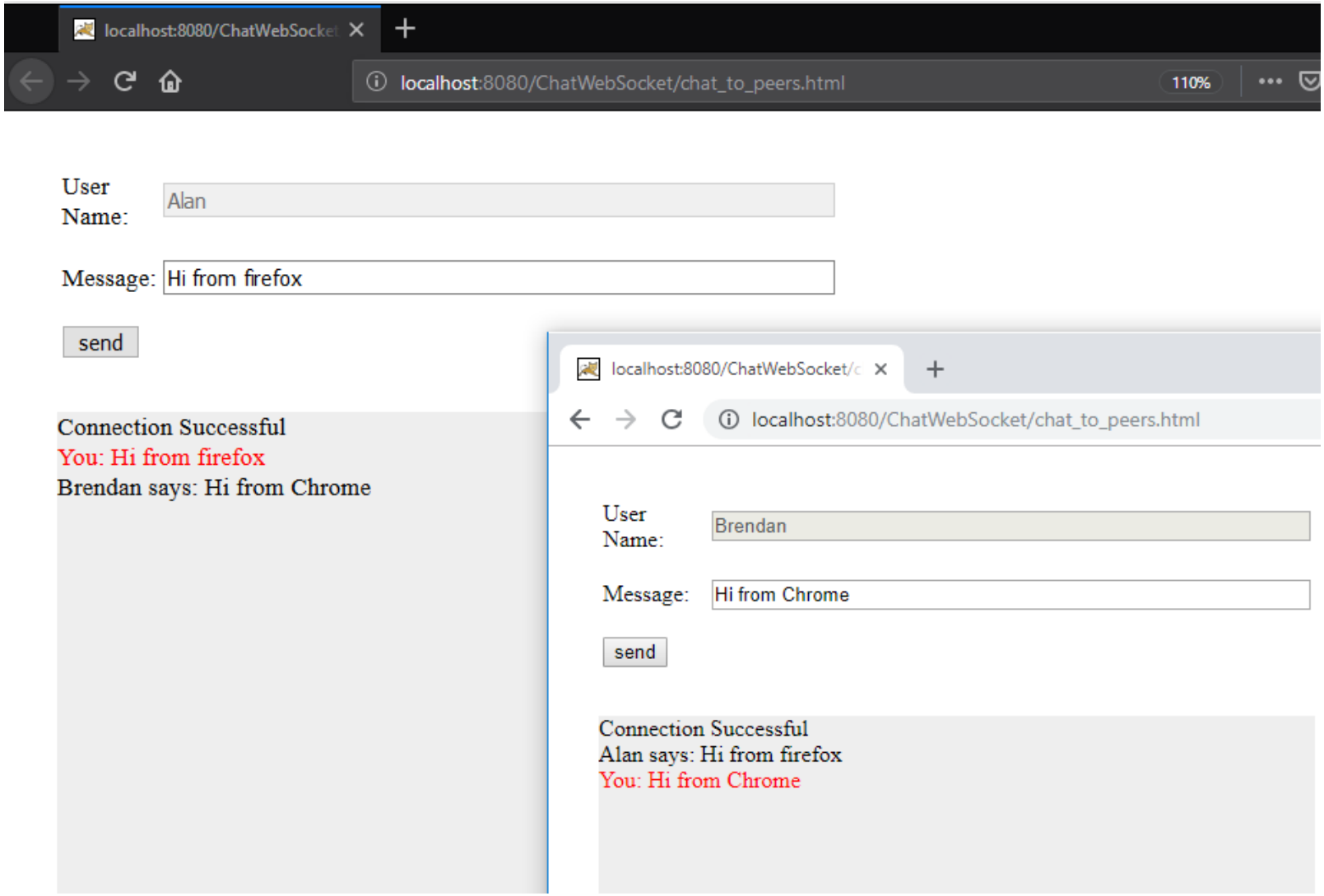
Example 1

chat_to_server.html

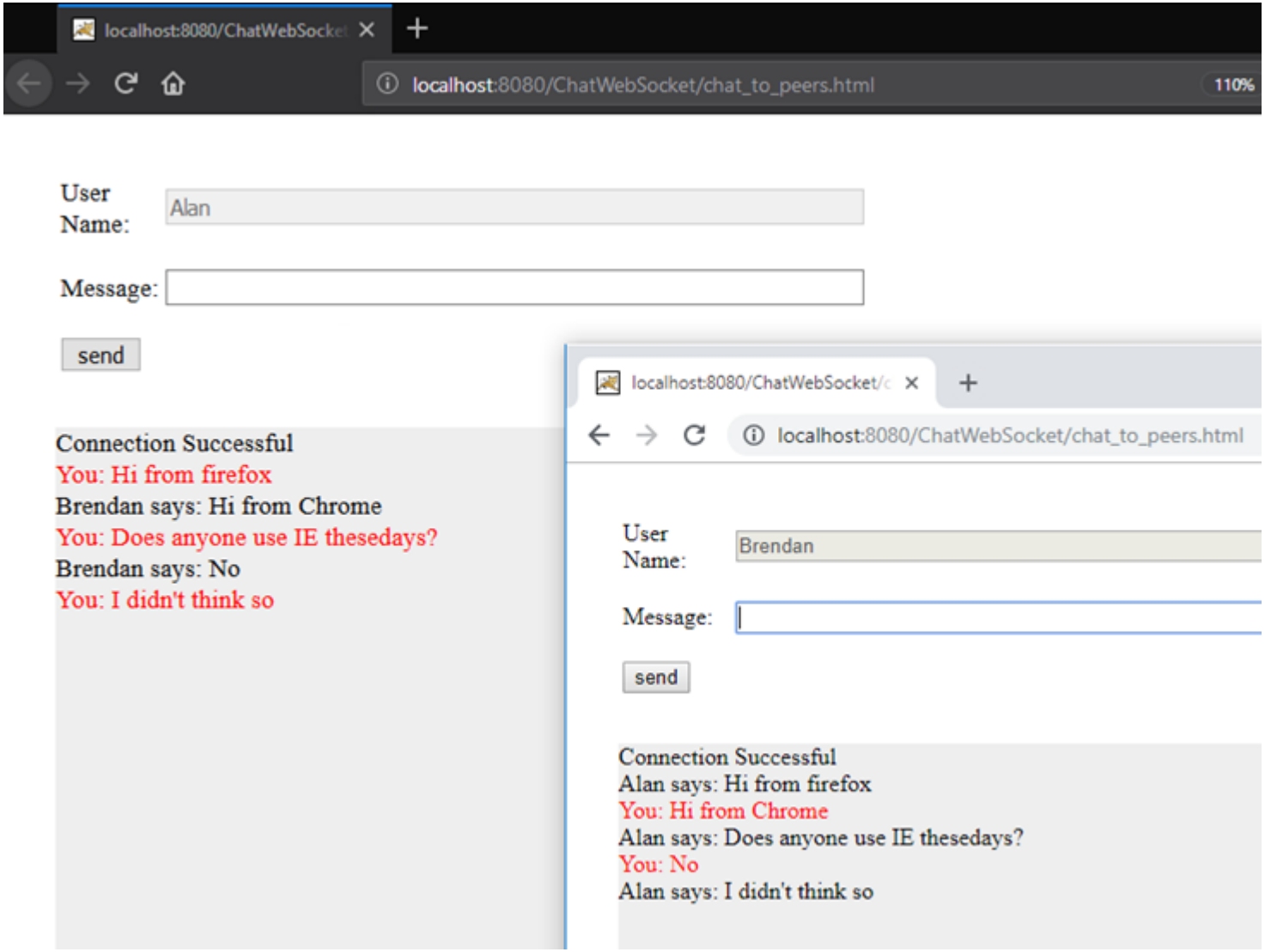
Source History

```
24        webSocket.onmessage = function(message) {
25                    divMsg.innerHTML += "Server> : " + message.data;
26
27        }
28
29        webSocket.onopen = function() {
30            console.log("connection opened");
31        };
32
33        webSocket.onclose = function() {
34            console.log("connection closed");
35        };
36
37        webSocket.onerror = function wserror(message) {
38            console.log("error: " + message);
39        }
40    </script>
```

Example 2


localhost:8080/ChatWebSocket
localhost:8080/ChatWebSocket/chat_to_peers.html
110%
User Name:
Alan
Message:
Hi from firefox
send
Connection Successful
You: Hi from firefox
Brendan says: Hi from Chrome
localhost:8080/ChatWebSocket/c
localhost:8080/ChatWebSocket/chat_to_peers.html
User Name:
Brendan
Message:
Hi from Chrome
send
Connection Successful
Alan says: Hi from firefox
You: Hi from Chrome

Example 2


localhost:8080/ChatWebSocket
localhost:8080/ChatWebSocket/chat_to_peers.html
110%
User Name:
Alan
Message:
send
Connection Successful
You: Hi from firefox
Brendan says: Hi from Chrome
You: Does anyone use IE thesedays?
Brendan says: No
You: I didn't think so
localhost:8080/ChatWebSocket/c
localhost:8080/ChatWebSocket/chat_to_peers.html
User Name:
Brendan
Message:
send
Connection Successful
Alan says: Hi from firefox
You: Hi from Chrome
Alan says: Does anyone use IE thesedays?
You: No
Alan says: I didn't think so

Example 2

PeerChatEndpoint.java

Source History

```java
15  @ServerEndpoint("/peerchatendpoint")
16  public class PeerChatEndpoint {
17
19      private static Set<Session> peers = Collections.synchronizedSet(new HashSet<Session>());
20
20      public PeerChatEndpoint() {
21          System.out.println("Class loaded " + this.getClass());
22      }
23
24      @OnOpen
25      public void onOpen(Session session) {
26          peers.add(session);
27          System.out.printf("Session opened for id %s%n ", session.getId());
28          try {
29              session.getBasicRemote().sendText("Connection Successful");
30          } catch (IOException ex) {
31              System.out.println(ex);
32          }
33      }
```

Example 2

PeerChatEndpoint.java

Source History

```java
    @OnMessage
    public void onMessage(String message, Session session) throws IOException, EncodeException {
        String[] args = message.split(":");
        System.out.printf("Message from %s to broadcast: %s%n ", args[0], args[1]);
        for (Session peer : peers) {
            if (!peer.equals(session)) {
                peer.getBasicRemote().sendText(args[0] + " says: " + args[1]);
            }
        }
    }

    @OnError
    public void onError(Throwable e) {
        System.out.println(e);
    }

    @OnClose
    public void onClose(Session session) {
        peers.remove(session);
        System.out.printf("Session closed with id: %s%n", session.getId());
    }
}//end class
```

Example 2

chat_to_peers.html

Source History

```html
<form>
    <table width="500" border="0">
        <tbody>
            <tr class="spaceUnder">
                <td><label for="userField">User Name:</label></td>
                <td><input id="userField" type="text" size="55"><br></td>
            </tr>
            <tr class="spaceUnder">
                <td><label for="messageField">Message:</label> </td>
                <td><input id="messageField" type="text" size="55"></td>
            </tr>
            <tr class="spaceUnder">
                <td>   <input onclick="sendMsg();" value="send" type="button"> </td>
            </tr>
        </tbody>
    </table>
</form>
```

Example 2

chat_to_peers.html

Source History

```
37          <div id="msg-box" style="width:500px; height: 400px; background: #eee; overflow:auto;"></div>
38          <script>
39              var webSocket = new WebSocket("ws://localhost:8080/ChatWebSocket/peerchatendpoint");
40              var message = document.getElementById("messageField");
41              var userName = document.getElementById("userField");
42
43              var divMsg = document.getElementById("msg-box");
44
45              function sendMsg() {
46                  document.getElementById("userField").disabled = true;
47                  var msgToSend = message.value;
48                  var userNameToSend = userName.value;
49                  webSocket.send(userNameToSend + ":" + msgToSend);
50                  divMsg.innerHTML += "<div style='color:red'>You: " + msgToSend +
                        "</div>"
52                  messageField.value = "";
53              }
54
55              webSocket.onmessage = function (message) {
56                  divMsg.innerHTML += "<div style='green'>" +message.data + "</div>";
                }
```

Example 2

chat_to_peers.html

Source History

```
59          webSocket.onopen = function () {
60              console.log("connection opened");
61          };
62
63          webSocket.onclose = function () {
64              console.log("connection closed");
65          };
66
67          webSocket.onerror = function wserror(message) {
68              console.log("error: " + message);
            }
70
71
72      </script>
```

References

https://www.baeldung.com/java-websockets

https://www.developer.com/java/ent/getting-started-with-websocket-apis-in-java.html

https://www.oracle.com/webfolder/technetwork/tutorials/obe/java/HomeWebsocket/WebsocketHome.html

http://www.websocket.org/

https://netbeans.org/kb/docs/javaee/maven-websocketapi.html