package techbysample.websocket.sample1;

import java.io.IOException;

import java.util.Vector;

import javax.websocket.OnClose;

import javax.websocket.OnError;

import javax.websocket.OnMessage;

import javax.websocket.OnOpen;

import javax.websocket.server.ServerEndpoint;

import javax.websocket.Session;

/\*\*

\*

\* @author TechBySample.com

\*

\*/

@ServerEndpoint("/server")

public class Server {

/\*\* static= per ogni istanza il vettore è lo stesso.

\*\*/

static Vector<Session> connessione=new Vector<Session>();

/\*\*

\*

\*Il metodo viene chiamato quando la connessione è stabilita.

\*

\* @param sessione

\*/

@OnOpen

public void onOpen(Session sessione) {

System.out.println(sessione.getId() + " has opened a connection");

try {

sessione.getBasicRemote().sendText("Connessione stabilita");

} catch (IOException ex) {

ex.printStackTrace();

}

}

/\*\*

\* Il metodo viene chiamato quando si chiude la connessione.

\*\*/

@OnClose

public void onClose(Session session) {

System.out.println("Session " + session.getId() + " has ended");

connessione.removeElement(session);

}

/\*\*

\* Il metodo è chiamato quando si riceve un messaggio a questo server endpoint. Il metodo

\* intercetta il messaggio e permette di agire di conseguenza.

\*/

@OnMessage

public void onMessage(String message, Session session) {

System.out.println("Message from " + session.getId() + ": " + message);

String username[] =new String[3] ; // vengono memorizzati utenti

username[0]="mancimichela";

username[1]="panfiliedoardo";

username[2]="melettimarco";

String password[]=new String[3];

password[0]="perioli";

password[1]="informatica";

password[2]="info";

try {

String x[] = message.split("#"); // quando arriva una richiesta, viene richiesto //il login, se è giusto la risposta è L#Si al contrario L#No

if(x[0].equals("R")){

String risposta="L#No";

for(int i=0;i<username.length;i++) {

if(x[1].equals(username[i]) && x[2].equals(password[i])) {

risposta="L#Si";

}

}

session.getBasicRemote().sendText(risposta);

if(risposta.equals("L#Si")) {

connessione.add(session);

}

}else if(x[0].equals("M")) { // se il client invia un messaggio, il server lo //legge e lo rimanda a tutti gli utenti connessi

for (int i=0 ; i<connessione.size() ; i++){

System.out.println("Messaggio");

connessione.get(i).getBasicRemote().sendText(message);

}

}

}

catch (NumberFormatException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* Il metodo viene chiamato quando c'è un errore.

\*

\* @param e

\*/

@OnError

public void onError(Throwable e) {

e.printStackTrace();

}

}

---------------------------------------------------------------------------------------------------------------

<!DOCTYPE html>

<html>

<head>

<meta charset=UTF-8>

<title>Tomcat WebSocket Chat</title>

<script>

var ws = new WebSocket("ws://" + location.host + "/chat/server");

ws.onopen = function(){

console.log("open!");

};

ws.onmessage = function(message){

document.getElementById("chatlog").textContent += message.data + "\n";

};

function postToServer(){

ws.send(document.getElementById("msg").value);

document.getElementById("msg").value = "";

}

function closeConnect(){

ws.close();

}

</script>

</head>

<body>

<textarea id="chatlog" readonly></textarea><br/>

<input id="msg" type="text" />

<button onClick="postToServer()">Send!</button>

<button onClick="closeConnect()">End</button>

</body>

</html>