

---

# 練習 WebSocket

---

鄭安翔

ansel\_cheng@hotmail.com

# Lab

## ■ Chat 網路應用專案

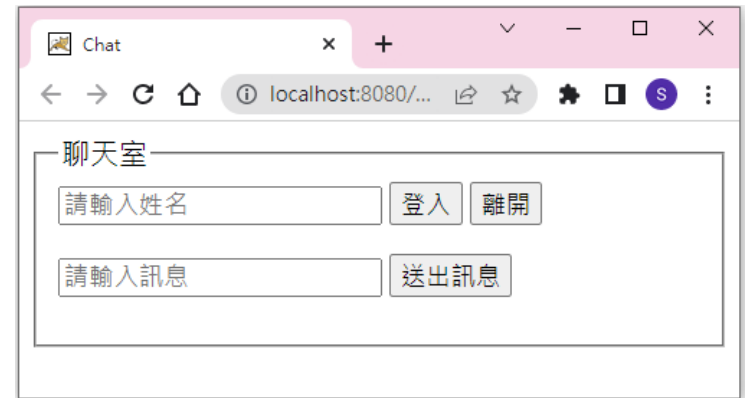
- 使用 WebSocker API 製作線上聊天室
  - 加入 gson-2.8.2.jar
- 建立 com.gjun.ChatMessage Java Bean
  - 記錄聊天內容
  - id / message 屬性
  - getter / setter 方法
- 建立 com.gjun.ChatService
  - WebSocket 伺服器端服務
  - @ServerEndpoint("/chatRoom/{id}")
  - CopyOnWriteArraySet<Session> sessions 儲存Session

# Lab

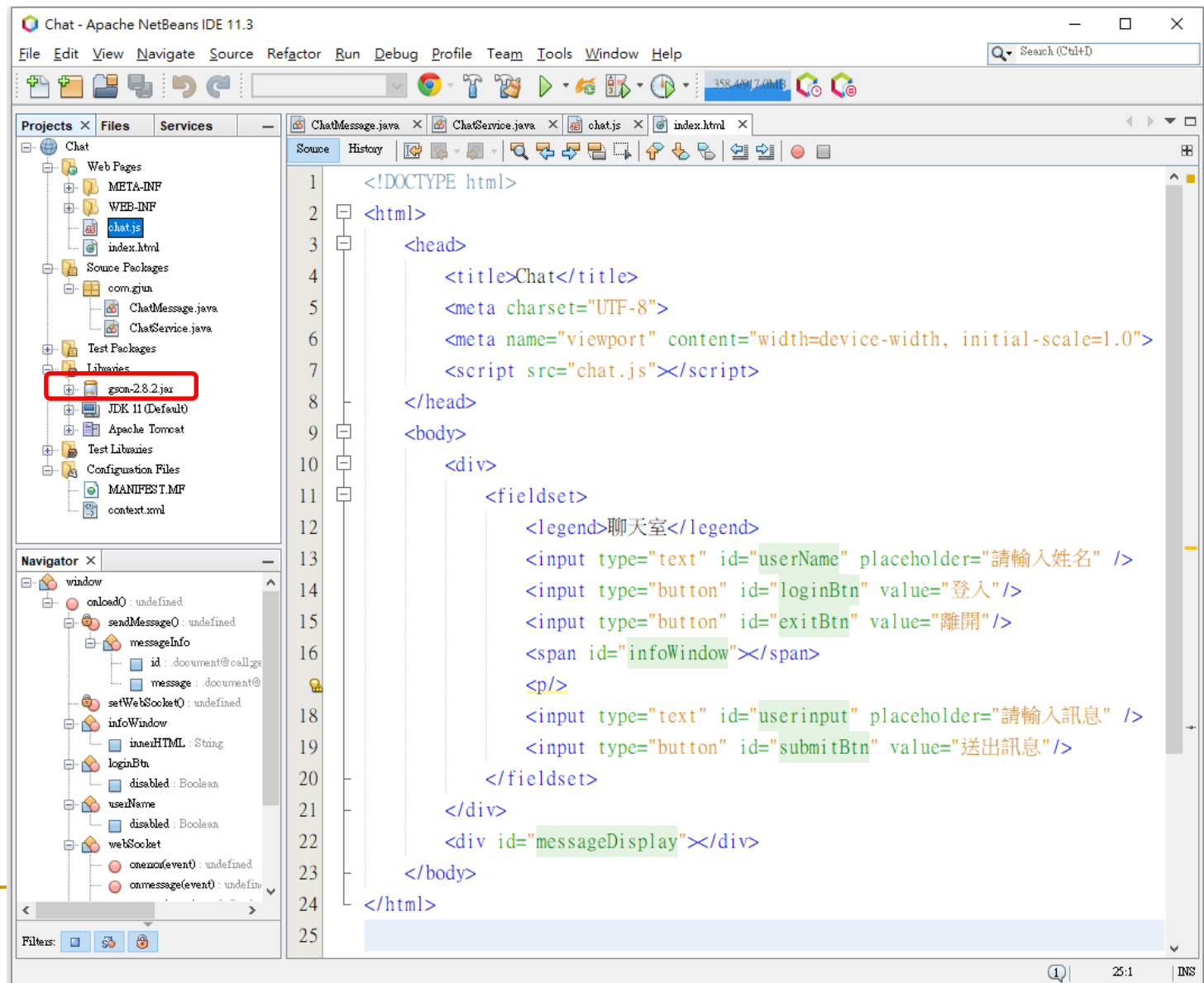
- @OnOpen方法
  - 將session 儲存至 sessions
  - 對每個連接的Client傳送訊息：*系統 : <id> 加入聊天室*
- @OnClose方法
  - 將session 從 sessions中移除
  - 對每個連接的Client傳送訊息：*系統 : <id> 離開聊天室*
- @OnMessage方法
  - 建構Gson序列化與反序列Json物件
  - 反序列化前端傳遞進來的Json字串
  - 對每個連接的Client傳送訊息：*<id> : <message>*

# Lab

- ❑ 複製 char.js 至專案
  - 檢視 char.js
- ❑ 修改 index.html 如右圖
  - 引入java script 檔案 char.js
- 測試、執行



# Chat 專案



# ChatMessage.java

```
ChatMessage.java
Source History
package com.gjun;

import java.io.Serializable;

public class ChatMessage implements Serializable{
    private String id;
    private String message;

    public ChatMessage(String id, String message) {
        this.id = id;
        this.message = message;
    }

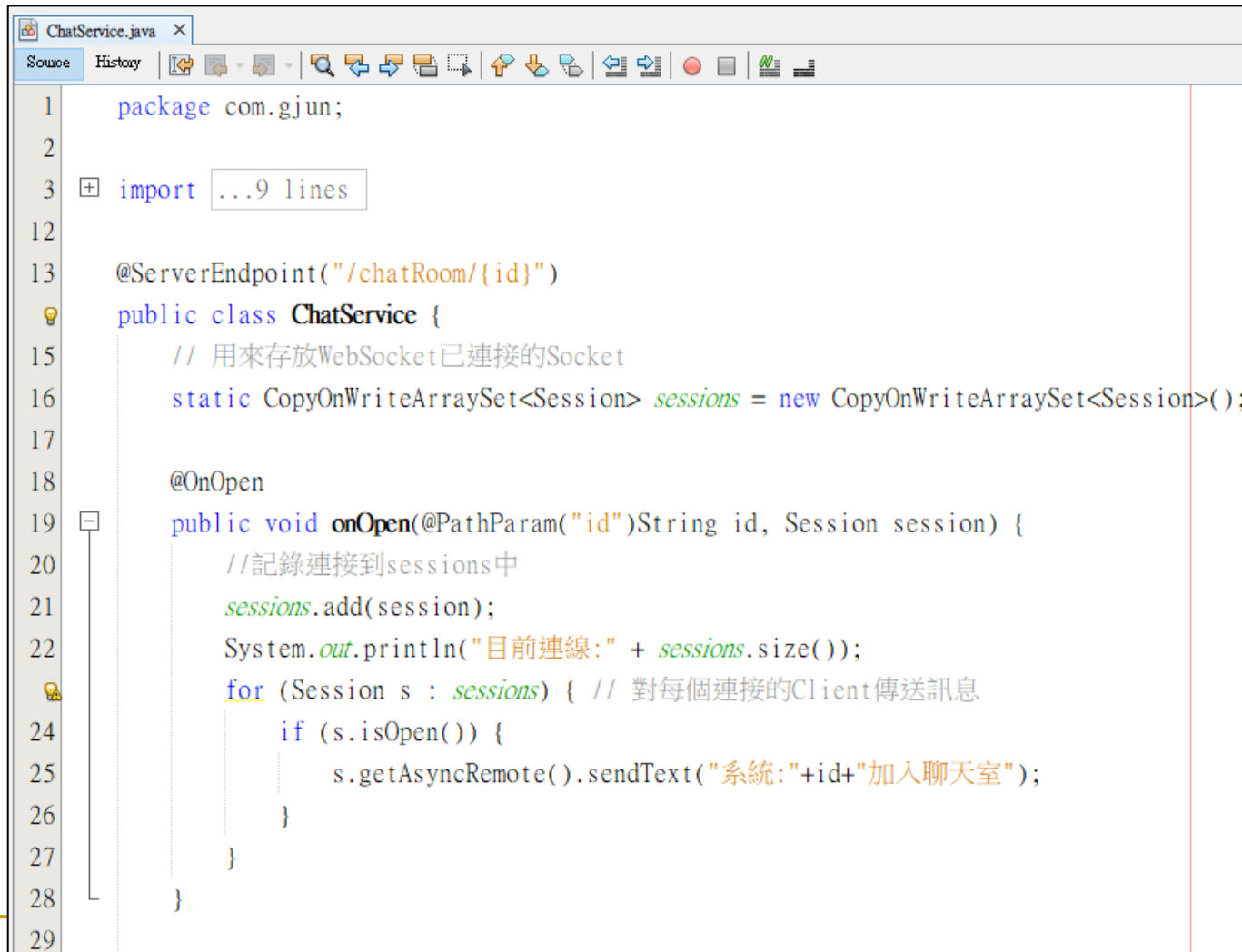
    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }
}
```

# ChatService.java



```
1  package com.gjun;
2
3  import ...9 lines
12
13  @ServerEndpoint("/chatRoom/{id}")
14  public class ChatService {
15      // 用來存放WebSocket已連接的Socket
16      static CopyOnWriteArraySet<Session> sessions = new CopyOnWriteArraySet<Session>();
17
18      @OnOpen
19      public void onOpen(@PathParam("id")String id, Session session) {
20          //記錄連接到sessions中
21          sessions.add(session);
22          System.out.println("目前連線:" + sessions.size());
23          for (Session s : sessions) { // 對每個連接的Client傳送訊息
24              if (s.isOpen()) {
25                  s.getAsyncRemote().sendText("系統:"+id+"加入聊天室");
26              }
27          }
28      }
29  }
```

# ChatService.java

```
30     @OnClose
31     public void onClose(@PathParam("id")String id, Session session) {
32         //將連從sessions中移除
33         sessions.remove(session);
34         System.out.println("目前連線:" + sessions.size());
35         for (Session s : sessions) { // 對每個連接的Client傳送訊息
36             if (s.isOpen()) {
37                 s.getAsyncRemote().sendText("系統:"+id+"離開聊天室");
38             }
39         }
40     }
41
42     @OnMessage
43     public void onMessage(String message,Session session) {
44         //建構Gson序列化與反序列Json物件
45         Gson gson=new Gson();
46         //反序列化前端傳遞進來的Json字串
47         ChatMessage chatmsg = gson.fromJson(message, ChatMessage.class);
48         for (Session s : sessions) { // 對每個連接的Client傳送訊息
49             if (s.isOpen()) {
50                 s.getAsyncRemote().sendText(chatmsg.getId() + ":" + chatmsg.getMessage());
51             }
52         }
53     }
54 }
```



# 檢視chat.js

```
chat.js x
Source History
1 window.onload = function () {
2     //獲取 DOM 元件
3     var userName = document.getElementById("userName");
4     var loginBtn = document.getElementById("loginBtn");
5     var exitBtn = document.getElementById("exitBtn");
6     var submitBtn = document.getElementById("submitBtn");
7     var infoWindow = document.getElementById("infoWindow");
8     var userInput = document.getElementById("userinput");
9     var messageDisplay = document.getElementById("messageDisplay");
10    var websocket;
11    var isConnectedSuccess = false;
```

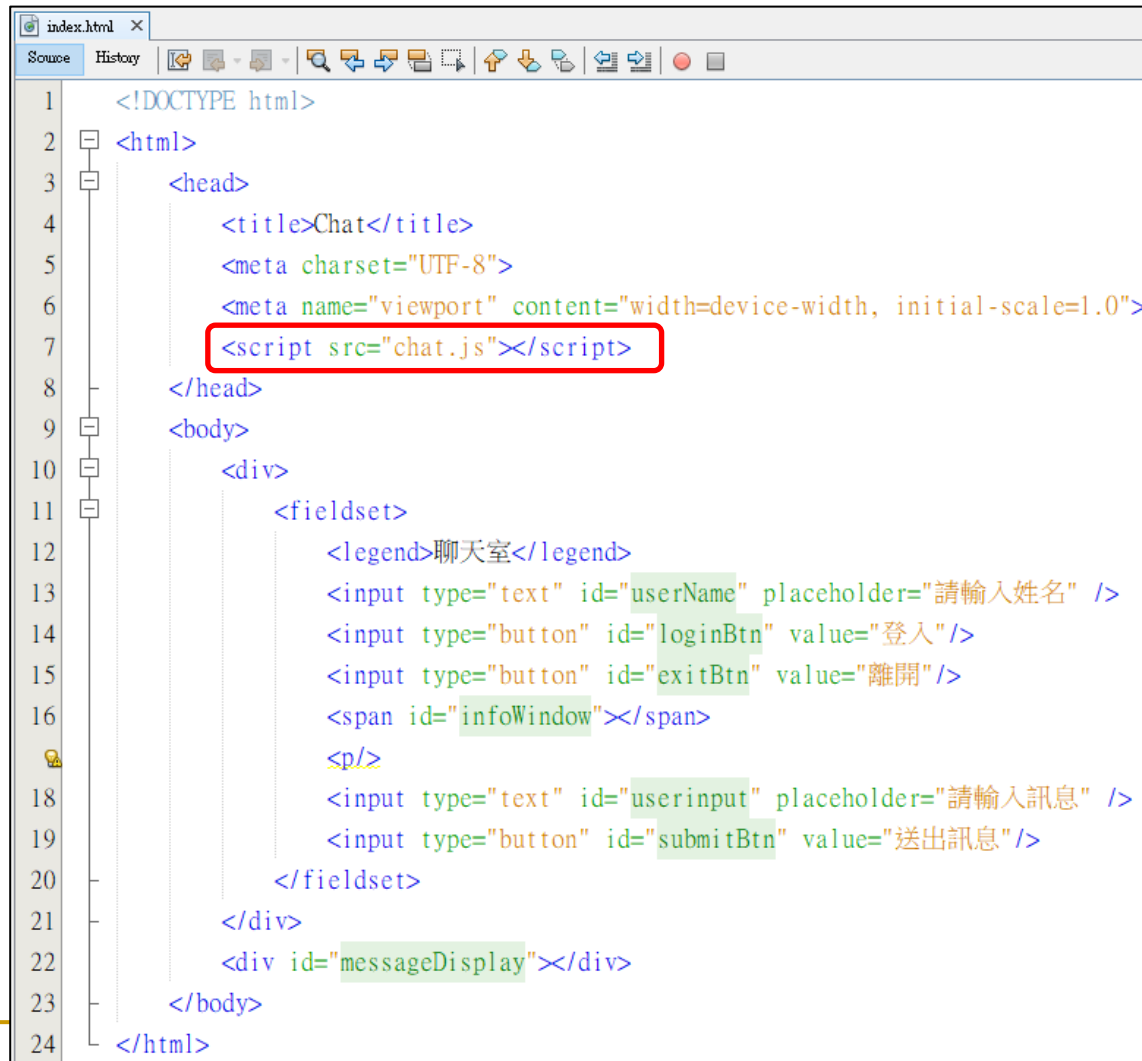
```
13    //設置登入鈕的動作，登入才可發言
14    loginBtn.addEventListener("click", function () {
15        //檢查有無輸入名稱
16        if (userName.value !== "") {
17            setWebSocket(); //設置WebSocket連接
18        } else {
19            infoWindow.innerHTML = "請輸入名稱";
20        }
21    });
22
23    exitBtn.addEventListener("click", function () {
24        location.reload();
25    });
26
27    //Submit Form時送出訊息
28    submitBtn.addEventListener("click", function () {
29        sendMessage();
30    });
31
```

# 檢視chat.js

```
32 //使用webSocket擁有的function, send(), 送出訊息
33 function sendMessage() {
34     //檢查WebSocket連接狀態
35     if (webSocket && isConnectedSuccess) {
36         var messageInfo = {
37             id: userName.value,
38             message: userInput.value
39         };
40         webSocket.send(JSON.stringify(messageInfo));
41     } else {
42         infoWindow.innerHTML = "未登入";
43     }
44 }
45
```

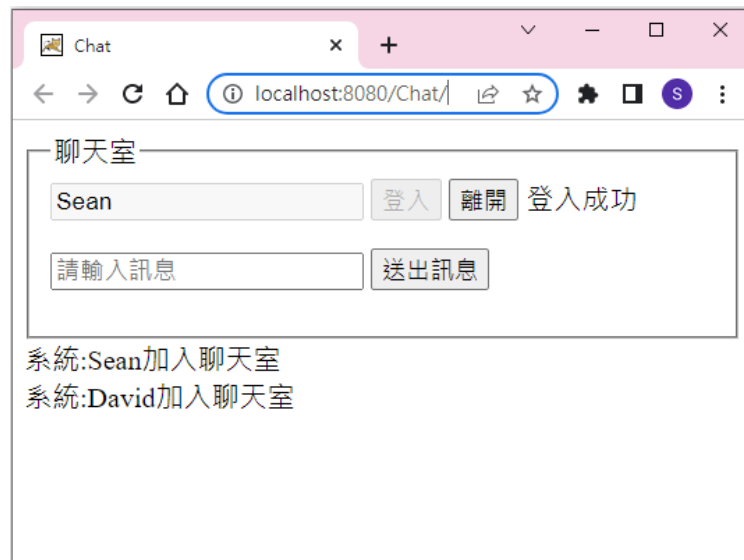
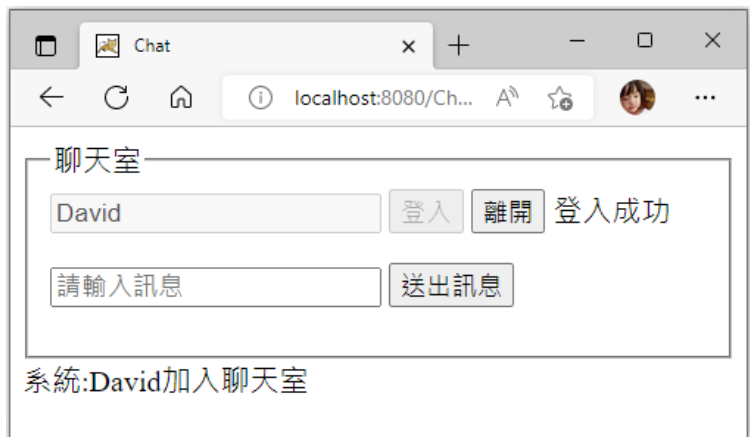
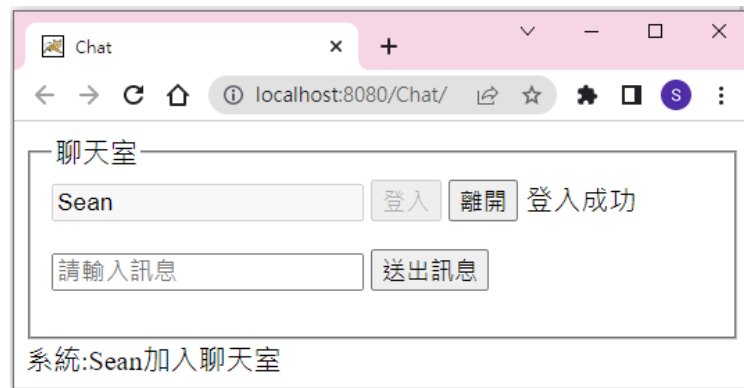
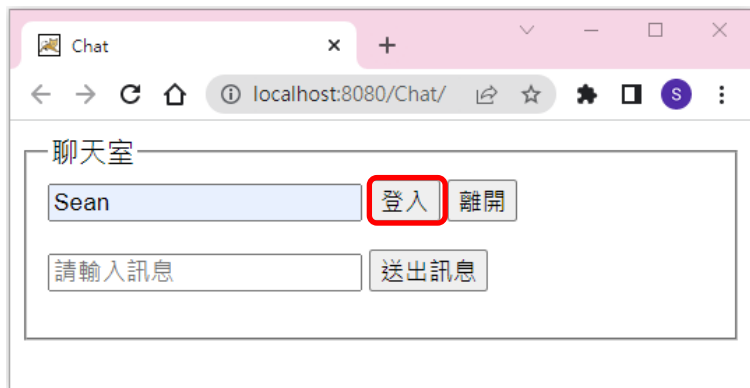
```
46 //設置 WebSocket
47 function setWebSocket() {
48     var url = 'ws://localhost:8080/Chat/chatRoom/'+userName.value;
49     console.log(url);
50     // 開始 WebSocket 連線
51     webSocket = new WebSocket(url);
52     //以下開始偵測WebSocket的各種事件
53     // onerror , 連線錯誤時觸發
54     webSocket.onerror = function (event) {
55         loginBtn.disabled = false;
56         userName.disabled = false;
57         infoWindow.innerHTML = "登入失敗";
58     };
59     // onopen , 連線成功時觸發
60     webSocket.onopen = function (event) {
61         isConnectedSuccess = true;
62         loginBtn.disabled = true;
63         userName.disabled = true;
64         infoWindow.innerHTML = "登入成功";
65     };
66
67     // onmessage , 接收到來自Server的訊息時觸發
68     webSocket.onmessage = function (event) {
69         //var messageObject = JSON.parse(event.data);
70         messageDisplay.innerHTML += event.data + "<br/>";
71     };
72 }
73
74
```

# index.html

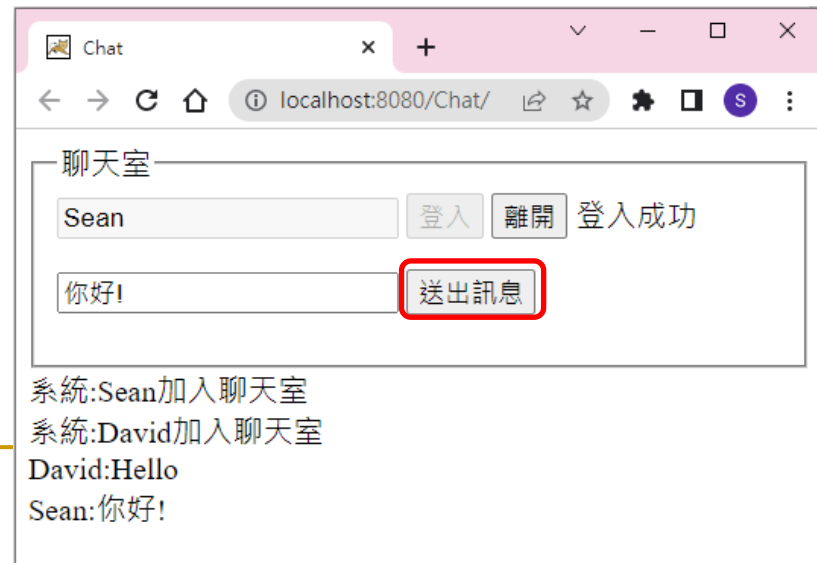
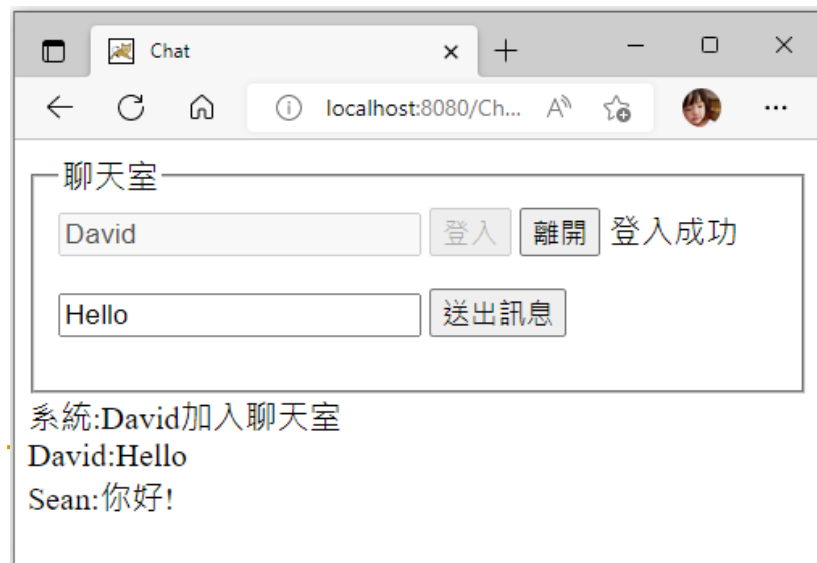
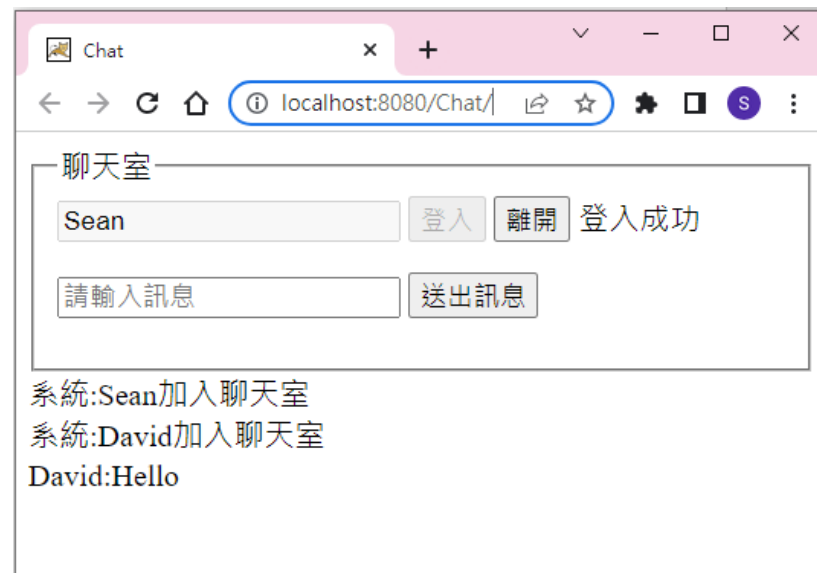
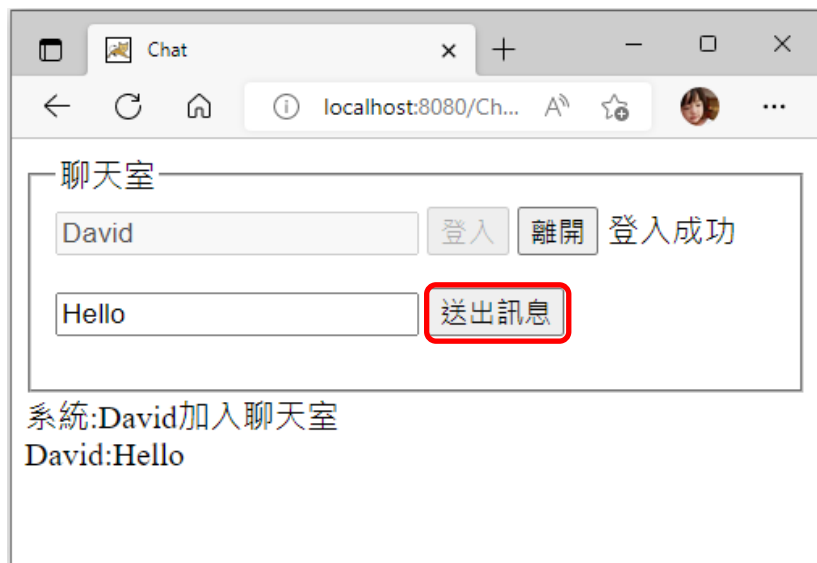


```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Chat</title>
5     <meta charset="UTF-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <script src="chat.js">/script>
8   </head>
9   <body>
10    <div>
11      <fieldset>
12        <legend>聊天室</legend>
13        <input type="text" id="userName" placeholder="請輸入姓名" />
14        <input type="button" id="loginBtn" value="登入" />
15        <input type="button" id="exitBtn" value="離開" />
16        <span id="infoWindow">/span>
17      <p/>
18        <input type="text" id="userinput" placeholder="請輸入訊息" />
19        <input type="button" id="submitBtn" value="送出訊息" />
20      </fieldset>
21    </div>
22    <div id="messageDisplay"></div>
23  </body>
24 </html>
```

# 測試、執行



# 測試、執行



# 測試、執行

