

嵌入式系統實驗實驗一：

JavaScript and web programming basics

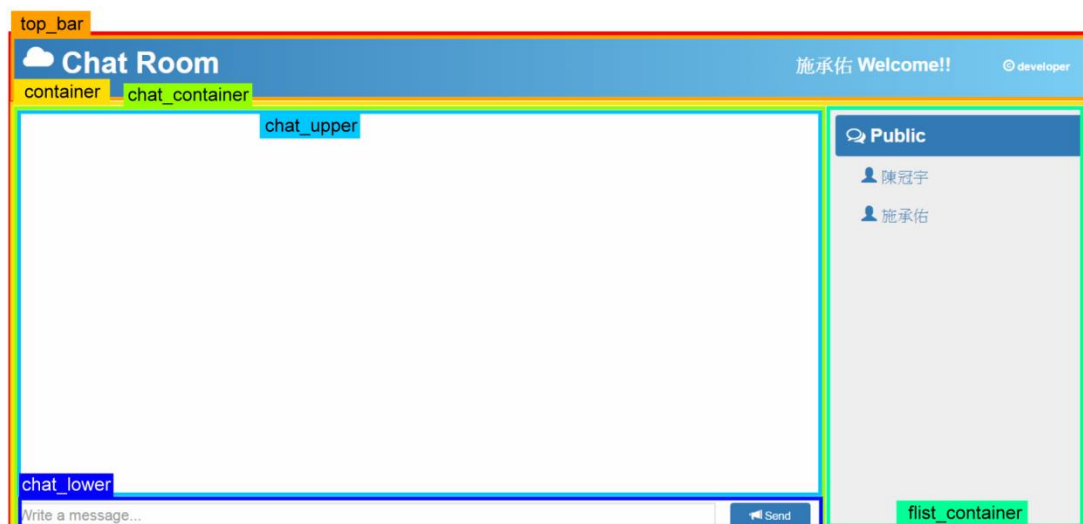
B02901086 施承佑 B02901083 陳冠宇

一、 功能

- I. 使用套件
 - i. Node.js
 - ii. socket.io
 - iii. express
 - iv. bootstrap
- II. 公開聊天室
 - 全部用戶皆可接收訊息的聊天室
- III. 私人聊天室
 - 可建立一對一的私人聊天室
- IV. 個人用戶登入
 - 登記個人用戶名稱以使用聊天室
- V. 顯示訊息傳送時間
 - 游標停置於訊息上時，可顯示訊息傳送時間
- VI. 新訊息提醒
 - 未讀訊息會於該用戶的列表旁邊顯示提醒

二、 架構

- I. 聊天室頁面（主要區塊）

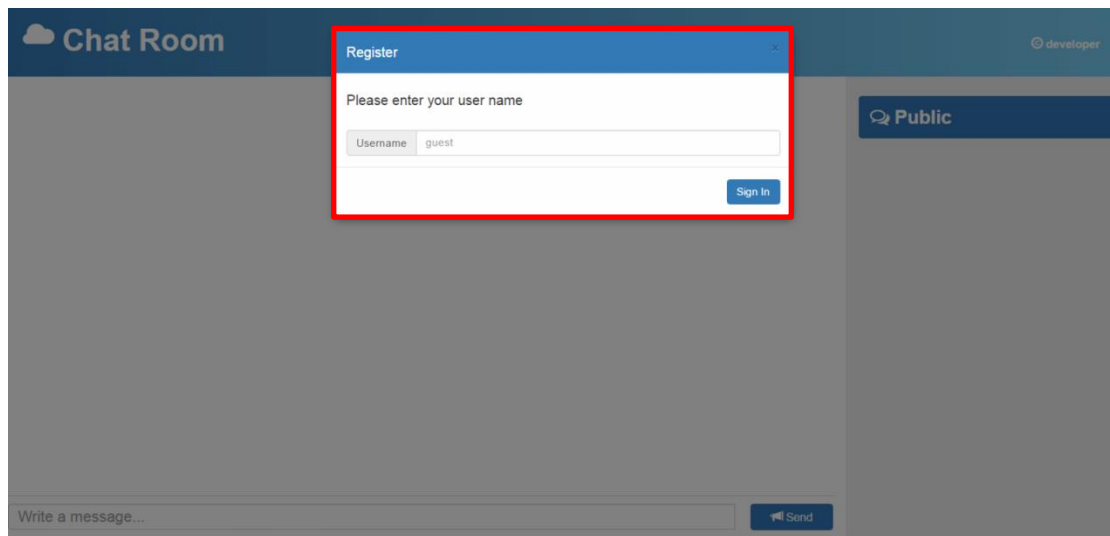


```

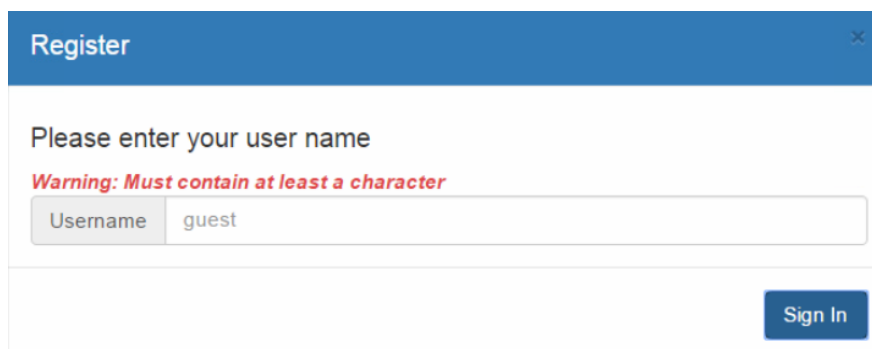
<div id="top_bar" class="header">
</div>
<div id="container" class="row">
  <div id="chat_container" class="col-md-9">
    <div id="chat_upper" style = "overflow: auto; height:90%">
    </div>
    <div id="chat_lower" style = "overflow: auto; height:10%">
    </div>
  </div>
  <div id="flist_container" class="col-md-3">
    <div id="flist_body" style="overflow: auto; height:90%">
    </div>
  </div>
</div>

```

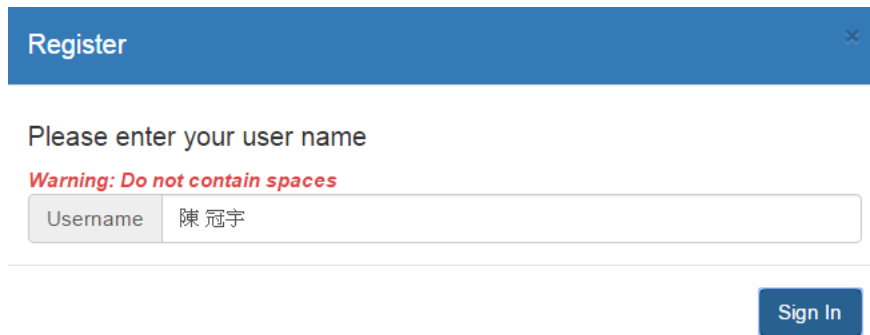
II. 登入畫面（彈出視窗）



- i. 此視窗讓使用者輸入用戶名稱登入聊天室
- ii. 若輸入名稱有問題會顯示錯誤訊息
 1. 無輸入任何文字

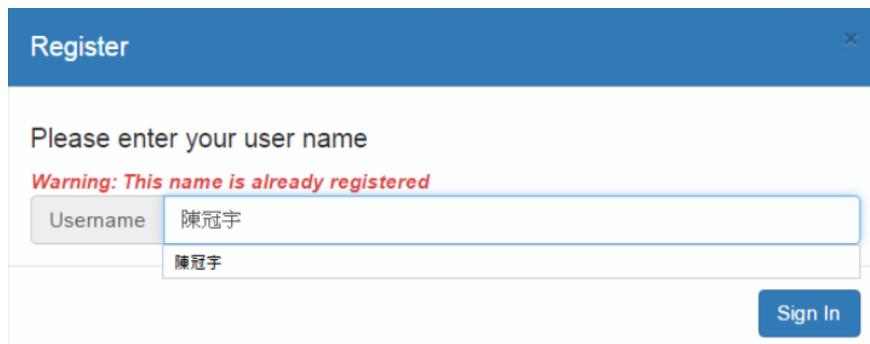


2. 名稱內含有空格



The image shows a 'Register' modal window. It has a blue header with the title 'Register' and a close button. Below the header, it says 'Please enter your user name'. A red warning message reads 'Warning: Do not contain spaces'. There is a text input field with the label 'Username' and the value '陳冠宇'. A 'Sign In' button is located at the bottom right.

3. 已被註冊過的用戶名稱



The image shows a 'Register' modal window. It has a blue header with the title 'Register' and a close button. Below the header, it says 'Please enter your user name'. A red warning message reads 'Warning: This name is already registered'. There is a text input field with the label 'Username' and the value '陳冠宇'. A 'Sign In' button is located at the bottom right.

- iii. 此彈出視窗為 bootstrap 裡面的 modal 元件，頁面一打開便預設開啟。並設定無法任意關閉。

```
$(document).ready(function(){
    $("#User_info").modal({
        backdrop: 'static', \\ 滑鼠點擊背景不會關掉彈出視窗
        keyboard: false    \\ "ESC"鍵不會關掉彈出視窗
    });
    $("#User_info").modal('show');
});
```

- iv. 使用者輸入後觸發`$("#form").submit()`，並經由 `socket.io` 送出使用者的用戶名稱到 `server` 端，接著關閉彈出視窗。`Server` 端將收到的用戶名稱收集，加入獨立的 `room`，並廣播通知其他 `client` 端。

1. Client 端

```
$('#sign').submit(function(e){
    e.preventDefault();
    username = $('#modal_name').val();
    $("#User_info").modal('hide');
    socket.emit('username', username);
    return false;
});
```

2. Server 端

```
socket.on('username', function(name){
    socket.name = name;
    socket.join(name); \\ 和一個獨立的 room 連接
    users.push(name); \\ 存入用戶名單
    io.emit('online', socket.name);
});
```

III. 聊天框 (chat_container)

i. 訊息輸入 (chat_lower)

1. 以<form>的形式,藉由其submit()的function將訊息內容傳出。
包含一個<input>和一個<button>。

```
<form id="input_bar" class="input-group">
    <input id="input" type="text" class="form-control" placeholder=
    "Write a message..." autocomplete="off" />
    <button id="send" class="btn btn-primary">Send</button>
</form>
```

2. 使用者輸入訊息後送出,將內容與聊天對象紀錄成JSON經由socket.io送回Server端,並清空輸入框。Server端判斷聊天對象後,將內容傳至目標所在的Room。

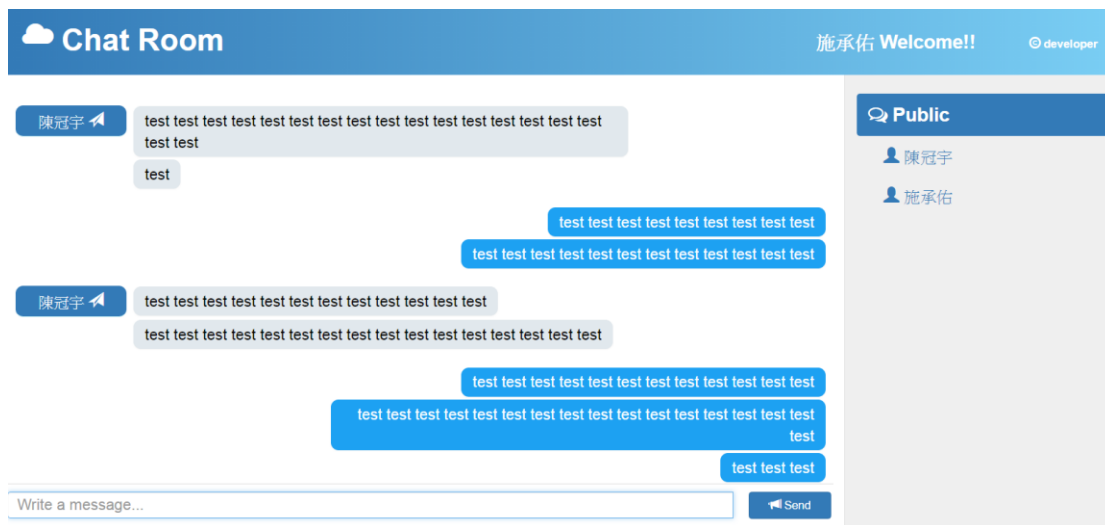
甲、Client 端

```
$('#input_bar').submit(function(){
    socket.emit('chat message', {msg: $('#input').val(), obj: window.substr(1)});
    $('#input').val(''); \\清空輸入框
    return false;
});
```

乙、Server 端

```
socket.on('chat message', function(msg){
    let m = {
        head: socket.name,
        message: msg.msg,
        to: msg.obj
    }
    if (msg.obj === "public"){
        io.emit('chat', m); \\公開訊息
    }else{ io.sockets.in(msg.obj).emit('chat', m); } \\私人訊息
});
```

ii. 對話框 (chat_upper)



1. 依照訊息的發送者分成兩種 class：panel-body, panel-body-friend。前者設定 css 為 float: right, 後者則 float: left, 讓顯示上有所區隔。

```
socket.on('chat', function(msg){
  if (msg.head === username){
    let message = $("

", { "class": "panel-body", text: msg.message});
    let clr = $("

", { "class": "panel-after"});
    div.append(message, clr);
  }else {
    let head = $("

", {"class": "panel-heading", text: msg.head + " "});
    let message = $("

", { "class": "panel-body-friend", text: msg.message });
    let clr = $("

", { "class": "panel-after"});
    div.append(head, message, clr);
  }
});

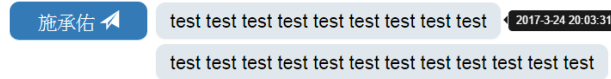

```

2. 每當 client 端從 socket.io 收到“chat”的傳送時,判斷訊息來源,將訊息內容 append 到該區塊。

```
if (msg.to === "public"){
  $("#public").append(div);
}else{
  $("#" + msg.head).append(div);
}
```

3. 利用 bootstrap 裡面的 tooltip, 將時間資訊附加在訊息上。當

滑鼠移到訊息上時會顯示。



甲、取得時間資訊

```
function now() {
    var date = new Date();
    var time = date.getFullYear() + '-' + (date.getMonth() + 1) + '-' + date.getDate() + ' ' + date.getHours() + ':' + (date.getMinutes() < 10 ? ('0' + date.getMinutes()) : date.getMinutes()) + ':' + (date.getSeconds() < 10 ? ('0' + date.getSeconds()) : date.getSeconds());
    return time;
}
```

乙、附加於訊息

```
message.tooltip({title:now(),"trigger":"hover",placement:"right"});
```

iii. 用戶列表 (flist_container)



1. 利用 **bootstrap** 裡面的 **nav-pills** 顯示所有上線用戶和共同聊天區，由 **server** 端通知新用戶上線，經由 **socket.io** 接收後，更新用戶。

```
socket.on('online',function(data){
    $("#flist").append(
        $("<li>",{id: "online_" + data }).append(
            $("<a>",{href: "#" + data}).append(" "+data)));
});
```

- 點選特定用戶，可傳送私人訊息。並改變對話框，顯示該用戶的相關訊息。利用 `hide()` 和 `stop()` 調整顯示的區塊。

```
$("#online_" + data).click(function() {
    _clickTab = $this.find('a').attr('href');
    $_clickTab.stop(false, true).fadeIn().siblings().hide();
});
```

3. 利用 bootstrap 的 badge，當其他用戶傳新訊息時，會有新訊息的數量顯示通知。



```
if (badge.length === 0){
    $("#online_" + msg.head).find("a").append(
        $("<span>", {"class": "badge"}).text(1)
    );
}else{
    let unread = +badge.text();
    badge.text(unread + 1);
}
```