

網程設期末專題報告

111550057 莊婷馨 / 111550139 郭芷安

GitHub連結

[annguo1106/webProgramming](https://github.com/annguo1106/webProgramming)

簡介

- 專題題目簡介

我們的主題靈感來自於 switch 上的遊戲: Overcooked, 以及眾多經典的廚房遊戲, 融合了在 project proposal 時老師提過的對戰模式, 讓兩個使用者和 server 連線後自動配對成一個房間, 並進行雙人對戰。雙方有各自的顧客需要負責, 需要透過切菜、組裝等等動作完成指定的食物, 先完成目標訂單數量的人就獲得勝利。

會選擇這個主題的原因, 是因為有關於 server 和 client 的應用, 我們覺得最有趣的的就是做線上遊戲, 可以讓使用者在遊戲中體驗即時互動的樂趣, 比起一般單純的文字聊天室更具吸引力。

- 成員分工

- 郭芷安(111550139): 主題發想、client 程式碼、報告
- 莊婷馨(111550057): 主題發想、server 程式碼、報告

- 開發與執行環境

- Ubuntu 22.04
- 運行在作業使用的 unpv13e 資料夾中
- 遵循 README.md 中的安裝指示

研究方法與設計

- Server與Client程式個別功能與分工原則

- Server

處理完兩個使用者的連線之後, 負責接收 client 送過來的資訊, 並判斷要送訊息讓玩家移動、拿起物品、組裝食物、將訂單送出、或是將食物送給顧客, 以及每個動作是否合法。

- Client

負責處理使用者在 UI 上的點擊, 將使用者點擊了何處、該處所有的物品和座標傳送給server。除此之外, 在收到 server 回復的訊息之後, 要負責在 UI 上顯示出對應的圖像或動作。

- Server與Client程式互動規則與資料傳輸格式

- 定義

在決定訊息格式前，我們先標定場景中每個物件以及位置的編號，client 會由座標判斷位置編號，並傳送給 server，而 server 會根據位置編號處理動作。object 編號主要讓 server 通知 client 場景上要放置的是哪些物件。

- location:
 - 0 - hands
 - 1 - chopping board
 - 2 - assemble counter
 - 3 - trash
 - 40 - lettuce 位置
 - 41 - tomato 位置
 - 42 - meat 位置
 - 43 - bread 位置
 - 44 - flavor 1 位置
 - 45 - flavor 2 位置
 - 46 - cone 位置
 - 5 - customer
 - 6 - floor
- object
 - l0 - lettuce
 - l1 - chopped lettuce
 - t0 - tomato
 - t1 - chopped tomato
 - m - meat
 - b - bread
 - f0 - 1st flavor ice cream
 - f1 - 2nd flavor ice cream
 - c - ice cream cone
 - b1111 - lettuce + tomato + meat + bread
 - b1011 - lettuce + meat + bread
 - b0111 - tomato + meat + bread
 - b0011 - meat + bread
 - i0 - 1st flavor ice cream + cone
 - i1 - 2nd flavor ice cream + cone
 - i2 - 2 flavors ice cream + cone
 - empty - empty

○ Client -> Server 的訊息

- 格式:[object] [from x] [from y] [to x] [to y] [to location] [action]
解釋:object 代表 client 點擊的地方或手上的 object, from x, from y 是 client 現在的座標, to x, to y 是 client 滑鼠點擊的座標, 而 location 是滑鼠點擊的地方。

○ Server -> Client 的訊息

- 格式:[10] [client number] [from x] [from y] [to x] [to y] [object] [hand(0) or not(1)]
解釋:這個訊息是用來告訴 client 要在指定地點新增或刪除物品
- 格式:[11] [client number] [from x] [from y] [to x] [to y] [object in hands]

解釋: 這個訊息是用來告訴 client 玩家要移動到甚麼地方

- 格式:[12] [client number] [order complete(1) / customer leaved(0)]

解釋: 這個訊息是用來告訴 client 一個 order 是否成功完成

- 格式:[13] [client number] [order1] [time1] [order2] [time2] [order5] [time5]

解釋: 這個訊息是用來告訴 client 接下來的五筆訂單分別是什麼, 以及每筆訂單的限制時間

- 格式:[14] [client number]

解釋: 這個訊息代表 client 傳給 server 的動作是 invalid command

- 格式:[97] [message to player]

解釋: 這個訊息用於告知 client 遊戲結果。傳送的 message 會顯示在螢幕上通知玩家, client 端準備 shutdown

- 格式:[98] [message to player]

解釋: 這個訊息是用來傳送 message, 會顯示在螢幕上通知玩家

- 例外狀況之分析與處理

當client試圖拿起不能拿起的東西、想組裝不能組裝的物品等, server都會回傳 [14] [client number]告知client該動作無效, 而client會直接忽略這筆訊息, 不對當前任何動作造成影響。

成果

- 主要功能

兩個使用者可以連進同一個房間進行對戰, 並根據server送出的訂單完成指定食物並交給顧客。而先完成目標數量訂單的使用者則獲得勝利。

- 特色

這個遊戲的特色是在操作自己的角色時, 也同時可以看到對方的操作, 增加遊戲的緊張感。另外這個遊戲的操作簡單, 只需要滑鼠點擊便可完成所有動作, 是適合所有年齡層的遊戲。

- 截圖



結論

- 專題製作心得 & 遭遇困難及解決經過
 - client:
 1. 最一開始遇到的困難是我不清楚client和ui之間要怎麼互相溝通，並且因為SFML本身有提供網路套件的關係，網路上的資料大都為server與SFML互動的參考。最後，在多方查詢之下，我找到的方法是在執行client後，讓client使用thread將ui執行起來，並讓兩個thread可以使用定義在同一個標頭檔的函式溝通，從而解決了這個問題。
 2. 在之後的設計中，因為這是一份有點小複雜的project，所以我在動手寫之前花了很多時間構思整份程式碼的架構，關於要分成幾個檔案、標頭檔應該要放甚麼等等，但在真正開始寫之後還是遇到了很多問題，常常變數宣告後，要用時卻總是忘記變數名稱，或先前定義過的變數在後來卻又沒用到等等。我認為這個project讓我多累積了很多建構程式骨架的經驗，並且在將來寫到類似的project經這次的經歷加以運用。
 - server:
 1. 首先遇到的困難是如何計時，以及如何判斷當下的 interrupt 是哪個 timer 觸發的。研究之後使用 POSIX timer 作計時，並且在 sigevent 中存放關於 timer 相關動作的資訊。
 2. 我原先使用的架構是 HW5 的程式架構，但該作業中只有 select read 的 socket，除此之外，因為我們的 project 要和多個 socket 溝通，根據收到的訊息做不同的回覆，如果在程式碼到處呼叫 write，很有可能造成堵塞或是 corrupt 訊息。因此我參考上課教的 non-blocking read write，對兩個 client 分別維護兩個 buffer，將要傳送的不同 message 在處理訊息時先放入 buffer，並在 writable 時把 buffer 中的訊息 write 到 client。
 3. 再來遇到的困難是 read 和 write 可能被頻繁觸發的 timer interrupt 打斷，對此我在 read 和 write 時加入 EINTR 的 error handling。
 - 總體：
 1. 我們花了很多時間討論client和server的訊息格式，也定義了很多簡易的代碼讓我們可以更有效的溝通。由於一開始兩個人對同一個訊息的理解多少有些出入，導致前期在寫的時候我們雙方都寫得有點卡，不確定對方會傳給自己甚麼，也不確定對方需要自己傳甚麼。但在著手寫一段時間後，因為有實際運作的關係，我們能更清楚自己需要甚麼而成功地更順暢的和對方溝通，並且為了確認對方真的能懂彼此的意思，我們也使用了很多情境模擬去一個一個推敲彼此會傳送甚麼訊息。雖然後來發現很多一開始規定的格式最後其實用不到，甚至可能帶來了不必要的麻煩，但這個過程還是讓彼此獲益良多。
 2. 在寫完這次 project 後，我們對於 stateful 和 stateless 的概念有更深的認識，原先以為很多資訊 client 也要存起來，但在實際操作之後，發現幾乎所有的資訊都可以存在 server 就好，client 可以單純的傳送操作動作，其他都交由 server 判斷。
- 成果未來改進或延伸方向
 1. 我們希望未來可以增加兩個使用者之間更多的互動。例如：兩個使用者可以達成特定條件獲得技能卡，而技能卡則能拿來攻擊對方使用者，使對方的client離開、減少對方訂單時間、或增加對方訂單難度，藉此增加遊戲競爭性。
 2. 希望能增加菜單變化性，讓關卡更加豐富。

3. 為了因應將來要加入的新功能及挑戰模式，我們也想要修改勝利條件或評分方式，讓勝利的條件不要過於單一。
4. 由於現在在ui介面的坐標軸是寫死的，而不會根據視窗大小改變，所以若是使用者改變了視窗大小，將會使座標定位不準確，而導致滑鼠點擊的地方卻不是client真正會互動的區塊。在未來希望能讓使用者可以任意調整視窗大小，也不影響操作。

參考文獻與附錄

- 課程教材以及這學期的作業(unpv13e)
- POSIX timer: [POSIX timers](#)
- SFML 前端設計: [IT 邦幫忙::一起幫忙解決難題, 拯救 IT 人的一天](#)