**明新科技大學工程學院**

資 訊 工 程 系

視窗程式設計

TCP/IP硬體聊天室

指導老師：江家頡

研究學生：

B10170095 李佾蔙

**目錄**

[**第一章 緒論** 2](#_Toc200844119)

[1.1 研究背景與動機 2](#_Toc200844120)

[1.2 研究目的 2](#_Toc200844121)

[1.3 問題陳述 2](#_Toc200844122)

[1.4 預期貢獻 3](#_Toc200844123)

[**第二章 系統設計** 4](#_Toc200844124)

[2.1 系統架構圖 4](#_Toc200844125)

[2.2 功能模組說明 5](#_Toc200844126)

[2.3 類別圖、循序圖等UML圖表 7](#_Toc200844127)

[**第三章 系統實現** 8](#_Toc200844128)

[3.1 開發環境（Visual Studio版本、.NET版本等） 8](#_Toc200844129)

[3.2 核心功能代碼解析 8](#_Toc200844130)

[3.3 關鍵技術實現說明 9](#_Toc200844131)

[**第四章 測試與驗證** 19](#_Toc200844132)

[4.1 單元測試案例 19](#_Toc200844133)

[4.2 系統整合測試 19](#_Toc200844134)

[**第五章** **結論與未來工作** 20](#_Toc200844135)

[5.1 研究成果總結 20](#_Toc200844136)

[5.2 專題限制 20](#_Toc200844137)

[5.3 未來改進方向 20](#_Toc200844138)

[**參考文獻** 21](#_Toc200844139)

**第一章 緒論**

## **1.1 研究背景與動機**

在現今資訊快速傳遞的時代，即時通訊已成為人們生活中不可或缺的一部分。LINE、Messenger等軟體不僅提供文字訊息，還加入了貼圖、emoji等功能，讓溝通更具情感與互動性。我在學習C#與網路通訊的過程中，發現多數教學著重在基本的文字傳輸，鮮少有整合貼圖與表情符號的範例。因此，我想挑戰自己實作一套具備圖片與表情傳輸功能的TCP/IP聊天室，讓學習網路程式設計的過程更貼近現代實務，也更有趣。

## **1.2 研究目的**

本研究目的是開發一套具備圖形化介面的雙人聊天室，透過TCP/IP實現Server與Client之間的訊息互傳，並加入貼圖與emoji功能。希望讓使用者能模擬出近似LINE的聊天體驗，同時熟悉C#、WinForms與Socket 通訊的整合應用。

## **1.3 問題陳述**

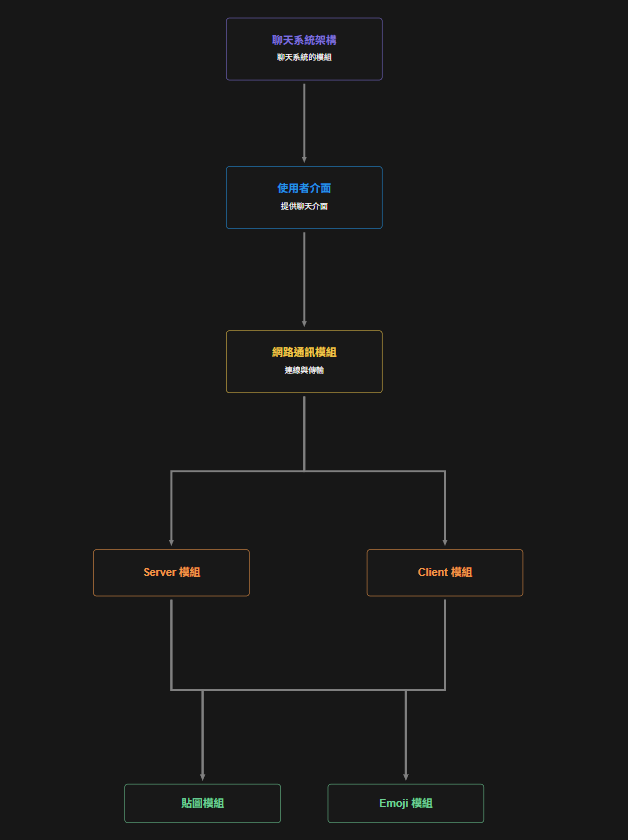
目前市面上的通訊軟體大多為商用產品，相關的開源專案較少，且大多僅涵蓋純文字聊天。對於像我這樣正在學習C#視窗應用與網路基礎的學生來說，缺乏一個完整結合前端介面與後端通訊邏輯的參考範例。尤其在圖像處理、訊息同步、界面排版等部分，往往需要花費大量時間摸索，因此我希望透過本專題建構一個具代表性的學習範例。

## **1.4 預期貢獻**

本專題預期成果為一個完整可執行的TCP/IP雙人聊天室，支援文字、貼圖與emoji傳輸。它不僅可以作為個人學習歷程的成果展示，也能提供後續學生或初學者作為學習Socket通訊、WinForms GUI與圖像傳輸等技術的範例。此外，這個架構具有良好擴充性，未來可進一步支援語音、GIF 或群聊等功能。

**第二章 系統設計**

## **2.1 系統架構圖**



## **2.2 功能模組說明**

1. 使用者介面模組

•ServerForm.cs/ClientForm.cs

•提供聊天介面（輸入框、發送按鈕、聊天訊息顯示區）。

•提供貼圖與Emoji按鈕，開啟貼圖與表情選擇視窗。

•管理與顯示傳送/接收訊息（文字 + 圖像）。

2. 網路通訊模組

•使用TcpListener（Server）與TcpClient（Client）建立連線。

•使用NetworkStream傳送與接收訊息。

•實作雙向即時通訊：Server與Client均能主動發送與接收資料。

•支援UTF-8編碼文字、貼圖指令（[Sticker]）、Emoji標籤（:emoji:）。

3. 訊息處理模組

* 負責辨識訊息內容為：

•一般文字

•[Sticker]檔名格式貼圖訊息

•:emoji:標籤訊息

* 將訊息適當轉為Label（文字）或PictureBox（圖片）顯示在介面中。

4. 貼圖模組

* 負StickerForm.cs

•自動載入「Stickers」資料夾內所有PNG圖片。

•顯示為可點擊的PictureBox。

•選擇貼圖後傳送[Sticker]貼圖檔名訊息給對方。

5. Emoji 模組

* EmojiForm.cs

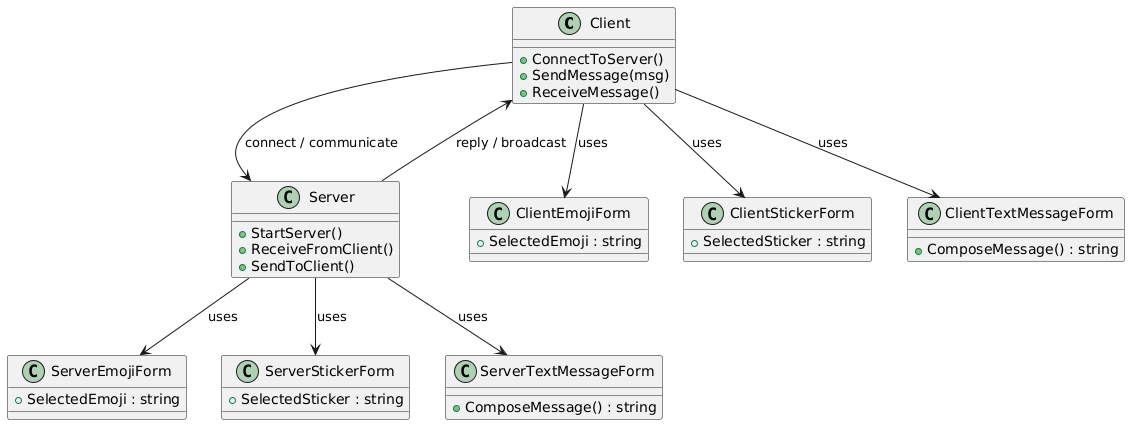
•自動載入「Emojis」資料夾內所有PNG表情圖。

•點選後將:emoji名稱:插入訊息文字框中。

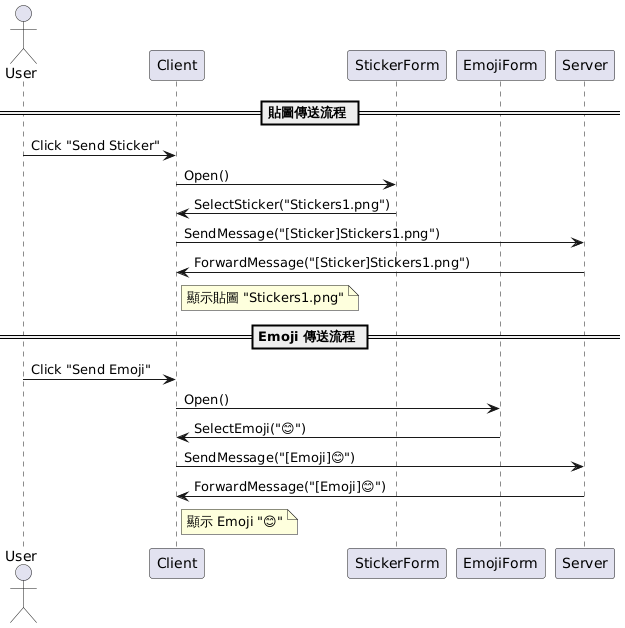
•透過訊息解析時轉換為圖像顯示。

## **2.3 類別圖、****循序圖等UML圖表**

**類別圖:**



**循序圖:**



**第三章 系統實現**

## **開發環境（Visual Studio版本、.NET版本等）**

•**語言：**C#

•**平台：**Windows Forms App(.NET Framework)

•**架構：**NET Framework4.7.2

•**開發工具：**Visual Studio 2022

•**通訊協定：**TCP/IP

•**伺服器IP：**127.0.0.1

## **3.2 核心功能代碼解析**

* ServerForm.cs

•啟動TCP伺服器並接受Client連線。

•處理文字訊息、貼圖與emoji的接收與顯示。

•使用FlowLayoutPanel顯示訊息、圖片與表情符號。

* ClientForm.cs

•連線至Server並發送訊息。

•可傳送[Sticker]貼圖檔名或插入:emoji:。

•顯示傳入的貼圖與emoji圖像。

* EmojiForm.cs

•讀取Emojis資料夾中的.png 圖片。

•點擊emoji圖像回傳:emoji:給Client/Server。

* StickerForm.cs

•讀取Stickers資料夾圖片並顯示。

•點擊貼圖傳送[Sticker]貼圖。

## **3.3 關鍵技術實現說明**

* TCP傳輸封裝：

•所有訊息都經由NetworkStream傳送與接收。。

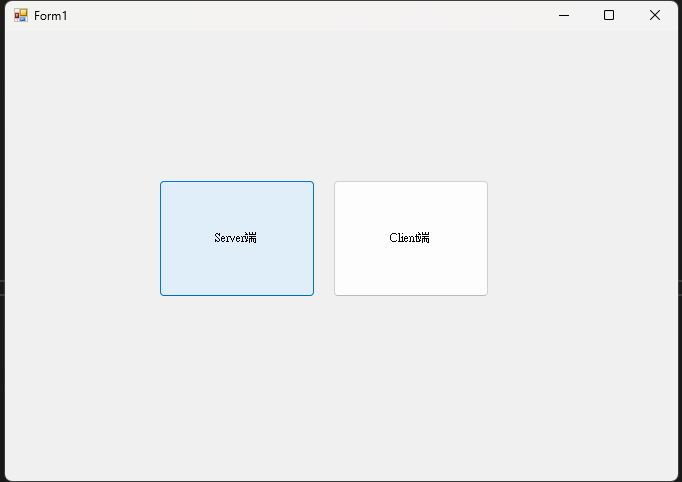
•支援純文字、貼圖標記、emoji標記。

* 圖像載入顯示：

•使用PictureBox顯示表情與貼圖圖像。

•使用MemoryStream讀取圖片避免鎖檔問題。

**3.4 界面設計與使用者操作流程**

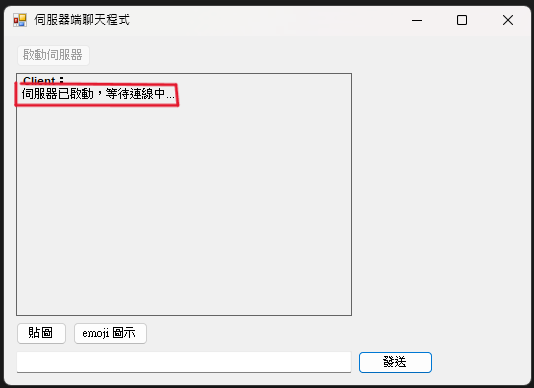


* **Server 端：**

1.點擊「Server端」後開啟。

2.啟動伺服器。

3.等待Client連線。

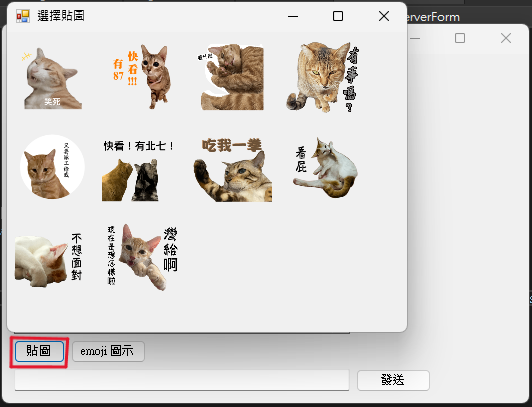


4.Client連線。



5.傳送貼圖。

按下貼圖即可選擇要傳送的貼圖。

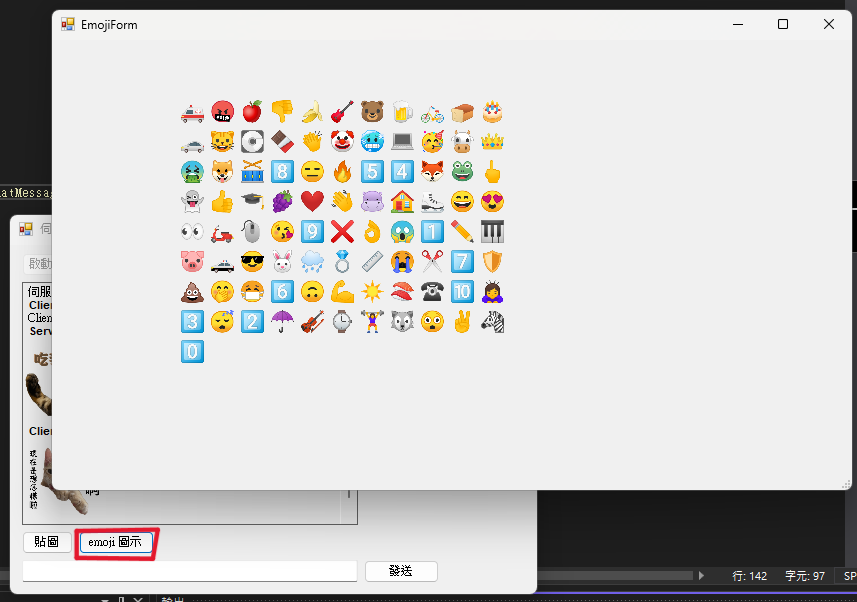


紅色框為Server端自己傳送，藍色框為Client端傳送。

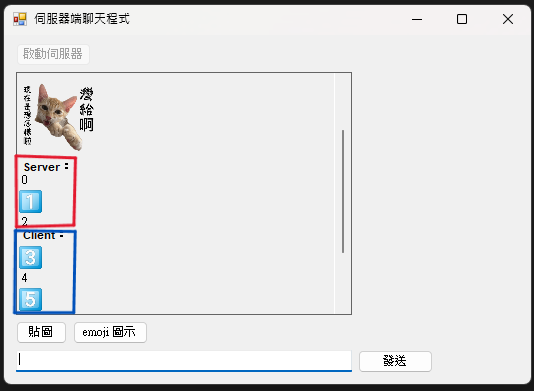


6.傳送Emoji。

按下emoji圖示即可選擇要傳送的Emoji。



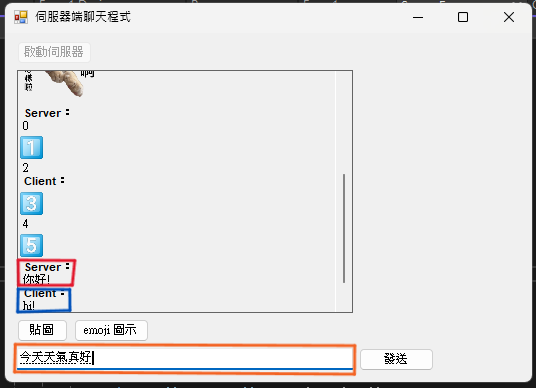
紅色框為Server端自己傳送，藍色框為Client端傳送。



7.傳送文字。

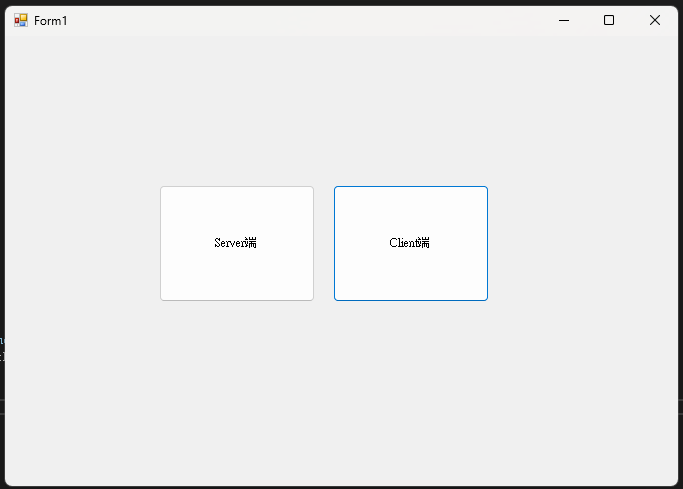
在橘色框內輸入文字。

紅色框為Server端自己傳送，藍色框為Client端傳送。



* **Client 端：**

1.點擊「Client端」後開啟。



2.點擊連線



3.顯示已連線Server端



4. 傳送貼圖

按下貼圖即可選擇要傳送的貼圖。

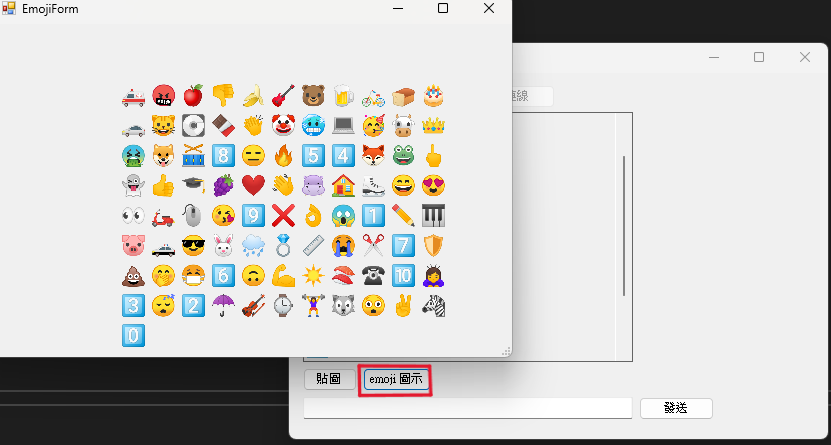


紅色框為Server端傳送，藍色框為Client端自己傳送。

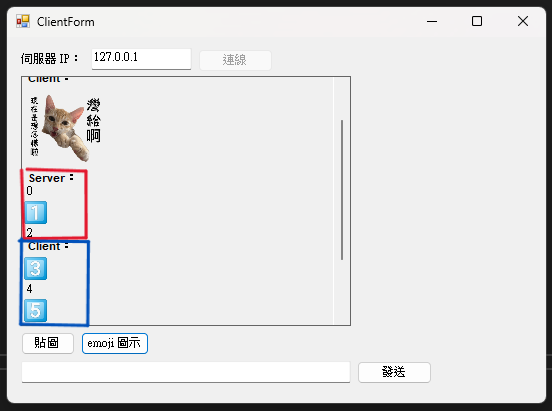


6.傳送Emoji。

按下emoji圖示即可選擇要傳送的Emoji。



紅色框為Server端傳送，藍色框為Client端自己傳送。



7.傳送文字。

在橘色框內輸入文字。

紅色框為Server端自己傳送，藍色框為Client端傳送。



**第四章 測試與驗證**

## **4.1 單元測試案例**

|  |  |  |
| --- | --- | --- |
| **測試項目** | **預期結果** | **結果** |
| 連線成功 | Server顯示 "Client已連線" | 通過 |
| 傳送純文字訊息 | 對方顯示正確內容 | 通過 |
| 插入emoji | 對方能正確顯示emoji圖片 | 通過 |
| 傳送貼圖 | 對方顯示對應貼圖 | 通過 |
| 資料夾不存在時 | 顯示警告提示（如貼圖/emoji遺失） | 通過 |

## **4.2 系統整合測試**

| **測試項目** | **操作步驟** | **預期結果** |
| --- | --- | --- |
| Client成功連線Server | 啟動 Server，再啟動 Client 並連線 | Server顯示「Client已連線」，Client顯示「連線成功」 |
| 傳送一般文字訊息 | 雙方輸入文字並按下「送出」 | 對方能即時看到正確文字內容顯示於聊天區 |
| 傳送貼圖 | 點擊「貼圖」按鈕→選擇貼圖 → 傳送 | 對方能正確看到貼圖，無錯誤訊息 |
| 插入並傳送Emoji表情符號 | 點擊「Emoji」按鈕→選擇一個Emoji→插入訊息並傳送 | 對方看到表情圖示顯示於訊息中，插圖尺寸與排版正確 |
| 混合訊息測試（貼圖＋文字） | 傳送貼圖後立即傳送一般文字 | 聊天介面依順序顯示貼圖與文字，無干擾或錯亂 |
| 混合訊息測試（Emoji＋文字） | 傳送包含Emoji的文字，例如「你好:smile:」 | 顯示「你好」+「」，Emoji 正確被圖示化 |

**第五章** **結論與未來工作**

## **5.1 研究成果總結**

本專題成功完成一套以C#開發的雙人聊天室系統，並整合了文字訊息、貼圖與emoji的即時顯示功能。透過這次實作，我不只更熟悉TCP/IP的通訊流程，也學會了如何將前端介面與後端邏輯結合，提升了自己在GUI設計與網路編程方面的能力。此外，模仿市面上的通訊軟體設計貼圖與emoji功能，也讓我開始思考使用者體驗在軟體開發中的重要性

## **5.2 專題限制**

由於開發時間與資源有限，目前系統僅支援一對一聊天，尚未實現多人聊天室、訊息加密或跨平台應用。此外，在嘗試加入GIF傳輸與即時翻譯功能時，也遇到一些技術上的挑戰，例如圖片檔案過大造成畫面延遲，翻譯功能整合流程較為複雜，因此最終未納入此次專題範圍。不過這些功能都具有實用價值，也將作為未來系統改進的重要方向。

## **5.3 未來改進方向**

* **語音訊息**

•加入錄音與播放功能，提升訊息的表達力。

* **即時翻譯功能**

•結合翻譯API，讓跨語言使用者也能順暢溝通。

* **GIF動圖功能**

•讓表達更生動，補足目前僅有靜態貼圖的限制。

* **訊息狀態**

•新增「已讀」、「送出中」等標示，提升使用者回饋。

* **多用戶與群組聊天**

•讓目前的雙人聊天室擴展為群組通訊平台。

**參考文獻**

[**三款好用的繪圖工具來解決系統架構或流程圖**](https://blog.wu-boy.com/2022/09/three-tools-design-system-architecture-and-flow/)

[**貼圖在我的 Visual Studio Raylib 專案裡載入不了**](https://www.reddit.com/r/raylib/comments/147ev82/ong_texture_not_loading_in_my_visual_studio/?tl=zh-hant)

[**将图片资源添加到VS中，并在代码中引用**](https://blog.csdn.net/qq_43474959/article/details/108370223)

[**將控件新增至圖片查看器 Windows Forms 應用程式 （.NET Framework）**](https://learn.microsoft.com/zh-tw/visualstudio/get-started/csharp/tutorial-windows-forms-picture-viewer-controls?view=vs-2022)

[**在Visual Studio 2017 寫一個聊天室**](https://ithelp.ithome.com.tw/articles/10206754)

[**簡易聊天室**](https://kingofamani.gitbooks.io/visual-studio/content/sample/ex1_hello_world.html)

[**C# Development | 如何快速建立 TCP/IP 連線並傳送資料**](C# Development | 如何快速建立 TCP/IP 連線並傳送資料)

[**C++实现网络聊天室-作业-visual-studio**](https://blog.csdn.net/2301_79973723/article/details/144322206)

[**使用C#的TCP/IP實作簡單的訊息傳送**](https://dotch01.github.io/unitywithtcpip/)

[**TCP 概觀**](https://learn.microsoft.com/zh-tw/dotnet/fundamentals/networking/sockets/tcp-classes)

[**C# 多人聊天室（基于TCP协议）**](https://www.cnblogs.com/zhangdezhang/p/17382738.html)

[**TCP/IP 網路程式設計**](https://cheer-link.tech/courses/tcp-ip/)