**計算機網路**

**期末socket專題報告**

**組員：林政儒 陳毅軒**

**學號：101502039 101502050**

目錄

1. **基本功能介紹**
   1. 構想起源
   2. 細節增添
   3. 完整功能
2. **環境使用安裝**
   1. Java standard edition development kit 7
   2. Eclipse
3. **程式碼介紹**
   1. GUI
   2. Server/Client
   3. 自訂義物件IOGroup/Listener
4. **遭遇困難與解決**
   1. 功能實作時間表
5. **組員分工**
   1. 程式碼與報告分工
   2. 總工時
6. **心得感想**
   1. 林政儒
   2. 陳毅軒

**基本功能介紹**

**構想起源：**

以前寫過一些使用socket程式，對於socket不甚了解，因此以文字傳輸為最基本的開始，選擇使用聊天室作為目標。

**細節增添：**

對多執行序(multi-thread)的使用也極為不熟悉，且認為單只是一對一的聊天室較為乏味且不方便使用，因此開始構想往多人聊天室的方式進行實作。

**完整功能(與基本使用)：**

#本程式僅使用於公開IP的Server或個別區域網路。

[開始介面選擇身分]

* + 1. 選擇成為Server端或Client端

[連線設定]

* + 1. Server端設定正確的Port
    2. Client端以正確的IP位址與Port與Server連線
    3. 檢查是否輸入錯誤資訊

[名稱設定]

* + 1. Server端以”admin”的身分顯示於聊天室
    2. Client端自行設定暱稱ID

[在線功能]

* + 1. 支援一位Server身分與多位Client身分的使用者在同一聊天室聊天
    2. 禁止使用者送出空白訊息洗版

[離線通知]

* + 1. 其中某位Client離開會通知所有在線使用者
    2. Server離開會告知所有Client端使用者”Server closed”

[離線後再連]

* + 1. server斷線後重開聊天室(以相同Port)則Client可以輸入Reconnect再次連線

**環境使用安裝**

先說明一下我們這個程式所使用的環境，然後簡單的介紹安裝與使用方法。

作業系統：Window7

選用語言：Java

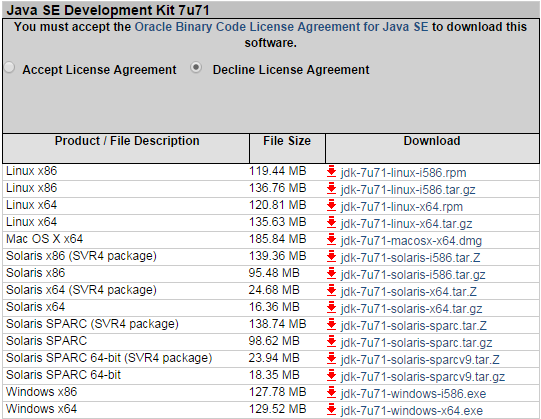
Java開發工具：JDK7(Java standard edition development kit 7)

編寫環境：Eclipse(eclipse-java-luna-SR1-win32-x86\_64.zip)

安裝介紹：

**JDK7**

* + 1. <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>



* + 1. 由控制台->系統及安全性->系統->進階系統設定->環境變數->選擇Path->編輯為C:\Program Files\Java\jdk1.7.0\_06\bin->按下確定即可

PS: Java的版本資訊在控制台->程式及->Java可以找到

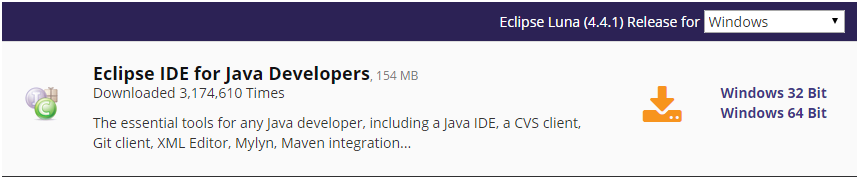
PS: 如果未安裝**JRE(Java Runtime Environment)**，則也需要安裝，這個是用來讓系統執行java，一般電腦原本就已經安裝好了(因為很多軟體都是由java執行的)。

<http://www.oracle.com/technetwork/java/javase/downloads/jre7-downloads-1880261.html>

若.jar檔仍未能開啟，右鍵點選檔案->內容->變更開啟程式->選取最新版本JRE

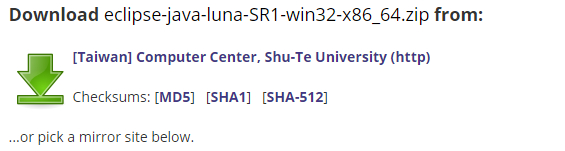
**Eclipse**

* + - * 1. <http://www.eclipse.org/downloads/>

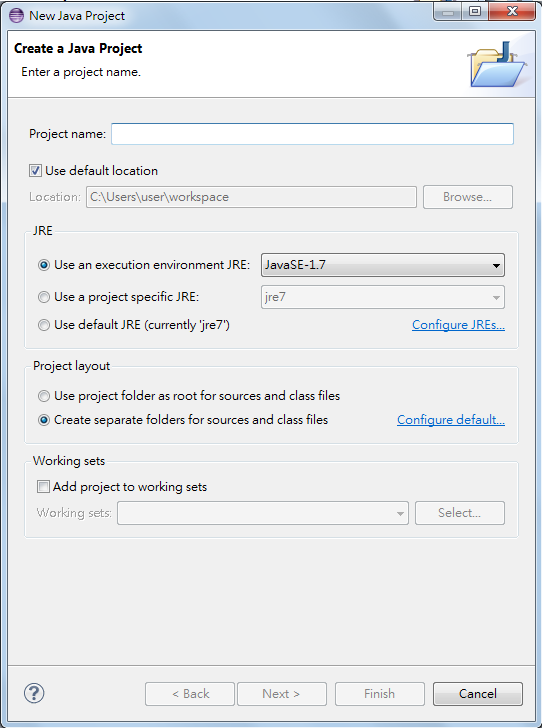


* + - * 1. 點選後進入

<http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/luna/SR1/eclipse-java-luna-SR1-win32-x86_64.zip>



使用方法介紹：

1. 點選eclipse捷徑
2. File->New->Java Project
3. 輸入專案名稱按下確定
4. 點選專案->File->New->Class即可開始編寫java程式

**程式碼介紹**

首先說明程式碼大致架構共具有GUI、Client、Server、Listener、IOGroup五個class

**先由GUI開始實作**

主要以awt及swing兩個package實作。

共實作GUI、setFrame、setWorkingPanel、setStartPanel、openServer、openClient五個函式以達到各項需要的功能。

以JFrame為視窗框架，具有兩個不同的Panel(JPanel)，第一個為**StartPanel**，在開始時提供選擇身分(Server/Client)的介面。選擇Server需填入Port，而選擇Client端時需填入ID、IP、Port，用JTextField、JTextArea、JLabel及JButton來實作，而當使用者輸入的項目不足或錯誤時以JOptionPane來實作提醒視窗，告知使用者。

以ActionEvent和AtionListener來來對Button(JButton)進行操作，得知使用者選擇的是Server或Client，第一個使用者須選擇Server來聯繫接下來各個使用者，由此進入**聊天室部分的WorkingPanel**，以身分的選擇會設定Status為Server或Client然後以此來判斷傳送訊息需呼叫的函式，以”Enter”鍵來確認訊息輸入完成並傳輸(用KeyEvent和KeyListener來實作)。

**再進行Server、Client兩個物件的實作**

Server和Client兩個class分別寫入Server和Client兩個物件的呼叫函式。並以IOGroup及Listener兩個Class輔助Server建立多個Client相互連線的機制。

**Server-**

*Input[Port, 聊天室輸入的testField, 聊天室輸出的textArea]*

以getClient(clientThread物件)用多執行序管理現有的clientList和serversocket，並實作sentMessage和close兩個功能。

註:**ClientThread –**

*Input及內含的物件[Server, serverSocket, clientList, 聊天室的輸出textArea]*

以多執行序的方式來不斷讀取是否有新的Client加入，以serverSocket確認新Client的加入，然後加入IOGroup物件的linkedList(此命名為clientList)，最後以server對各個已在線上的client傳達有新的使用者加入聊天室。

**Client-**

*Input[暱稱ID, IP, Port, 聊天室輸出的textArea,聊天室輸入的textField]*

包含Server的socket、output/input的objectStream、online四個非公開物件

以Listener(clientListener物件)用多執行序管理現有的client對server的socket傳送output和從server接收的input，並實作sentMessage和close兩個功能。

Online這個布林值是用以判斷是否有連上線(作為重新連線的判斷值)。

註:**ClientListener** –

*Input及內含的物件[作為server的socket, input/output的objectStream, 聊天*

*室輸出的textArea]*

從server取得聊天室訊息，若發生問題則顯示server已關閉。

**以IOGroup來實作clientList的物件**

*Input及內含的物件[Socket, output/input的objectStream]*

作為Client身分的物件，僅實作close的功能。

**以Listener來實作thread的物件**

*Input及內含的物件[server, clientList, client的IOGroup物件group, 聊天室輸出的textArea, client的暱稱ID]*

用來讀取訊息以利server傳遞，或者client離線時從clientList刪除。

**遭遇困難與解決**

[11/18]

1. 初始GUI架設:介面構造相較大一計算機實習的作業來說較為簡單，無問題。
2. Socket基本功能測試:在console直接測試簡易socket，無問題。
3. 套用GUI:直接將console內容放入GUI的button執行功能內，**有error**。

[11/19]

🡪先以一行行code丟入button嘗試執行，了解各行差異，最後成功在同一個class的main(String)裡面執行。

1. 測試IOStream連接:字串傳輸的測試，無問題。
2. 使用thread監聽訊息:以listenerThread來在server端接收各client端傳來的訊息，無問題。

[11/20]

1. Client/Server端斷線後重連線:Client沒問題，但Server端斷線後沒辦法自動重新回到等待連線的狀態。
2. 將以上功能加入GUI:有了前次經驗，無問題。
3. Server清除名單:當Client斷線或登出的時候清除clientList中關掉的IOGroup
4. 全面同步textArea:用server端統一發送訊息，無問題。

[11/22]

1. 設定IP和Port:取代原本用以測試的固定IP和Port，無問題。
2. 開始介面設定:Server只讀port，Client讀取ID、IP、Port，無問題。

[12/10]

1. 以系館電腦教室測試多人連線:電腦教室不可測試socket。

🡪在各自宿舍嘗試

[1/2]

1. 防呆功能:連線基本資料是否正確、送出前檢查訊息是否為空，無問題。

[1/5]

1. Server以相同port重開，Client重連，無問題。

**組員分工**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 程式碼分工 | 報告分工 | 總工時 | 總貢獻度 |
| 林政儒 | 80 | 20 | 60 | 65 |
| 陳毅軒 | 20 | 80 | 40 | 35 |

**總工時**

程式碼編撰：32小時

報告整理：5小時

總時數：40小時

**心得感想**

**林政儒—**

大一時首次接觸socket第一個實作的應用卻是簡單的伺服端對客戶端傳值，當時覺得有些不滿意但是時間的關係就暫且放一邊等到假期時間再繼續研究。到了寒暑假時，有嘗試著做連線可是忘了引入執行緒的概念，導致server-client之間的對話只能以一言我一語，無法同時說話。

接著就是這次的socket實作，一開始先做好GUI，接著很天真的想把socket的連線function直接寫進button listener裡面，卻發生不知名的錯誤。最後決定從最基本的連線開始打起。

第一版本，一對一的server-client架構，直接在main string裡面實作，用console啟動。第二版本，server和client加入multi-thread的概念，使得等待輸入資料的同時也能夠接收資料。

經過一、二兩個版本的簡單訓練後，決定將程式直接套用到GUI上，結果又error。很明顯，我們不能將main String的一長串程式碼直接帶入。後來決定從GUI的架構下逐一加入socket的程式碼，同時打算直接嘗試多人連線，讓服務端每次連接好一個客戶端後，回到等待動作，結果意外的成功。我想成功的點在於，server有回到 server.accept這個步驟的關係。詳細原因還沒仔細釐清，不過目前的程式已經可以透過127.0.0.1做測試並成功。

另外這份多人聊天室的IOGroup可以說是通訊時的核心物件。當Server收到某一個client的訊息後，必須要廣播到其他的client。我們將連線好的socket和io stream在啟動直行緒後，綁在一起，一同塞進一個list裡面。每當要廣播時，將訊息傳遍每一個IOGroup就行了，算是自己做的廣播系統。

這次作業算是達成一個簡單的心願，親手做出一個通訊軟體，雖然簡單，但是實用。

**陳毅軒—**

這次的作業，因為我對自己的時間安排出了問題，沒能投入很多心血進去，只有透過林政儒的幫忙，才記得許多socket與muti-thread的語法應用，很感謝有他幫忙。不過這次功能比較有許多都不是計網上課學到的東西，反而比較多是透過google來學習的感覺有點可惜，不過計網真的算是有學到一些對socket觀念性的認識，以這個程式設計作為這學期的整體報告對我來說也算是不錯的結果，其實還蠻開心的。

而主要對我們這次程式碼的認識，並不是剛開始看林政儒打出程式主軸的時候，而是在後來要進行一些改寫的時候才有更多的認識，對於java物件導向有重新複習的感覺也進行進一步的摸索，這學期對程式碼的運用多為單一個class，幸好有計算機網路這個socket作業意外地複習了一些早已遺忘的概念，是一個很大的進步。還有這次的程式碼中多次使用了try-catch語法，是我以前較不常用的部分，意外地學到一個簡單好用的語法，但對於其特性或說對程式執行時間的影響仍不清楚有待深究。最後，計網比較底層的概念似乎我們實作這次作業就比較沒有那麼有感觸了。

其實這次作業仍有可以改進的地方與一些不太懂的觀念仍值得釐清，可是卻因為真的我自己拖到快期末考才來最後的整理導致時間的不足，甚感遺憾。有些東西真的當下不做之後或許就沒時間了，希望以後能夠有其他機會以不同的形式對網路溝通聯絡能有更進一步的方式來認識。