嘉義大學

申請科系:資訊工程系

姓名:蔡聿善

目錄

簡歷表

姓　　名	蔡聿善	性　　別	男	
出生年月日	2002/01/24			
聯絡電話				
行動電話				
Email 信箱	jimmy01240397@yahoo.com.tw	興　　趣	寫程式、參與電腦資訊相關活動	
聯絡地址				

學　　歷	畢業學校	始（年/月）	至（年/月）
	新莊國小	97/9	103/6
	七賢國中	103/9	106/6
	高雄高工	106/9	109/6

競賽成果

活動類別	活動名稱	名次或結果
競賽	高雄市 107 年中學生應用程式設計比賽	第二名
	扎根高中職資訊科學教育計畫『107 年飆程式網高中職電腦程式設計線上解題活動』	團體組 金牌獎
	電機電子群科「學生專題暨創意製作競賽」校內初賽	佳作

	第 49 屆全國技能競賽南區分區技能競賽 資訊與網路技術	佳作

專長

- 獨自開發即時線上連線遊戲 APP(不含部分美術)
(程式部分除 GooglePlayGameService API、 MySQL 與 Unity 部分套件外，其餘皆從頭開發含 TCP UDP 伺服器在內。)於 2019/10/15 上傳至 GooglePlay
- 精通 C#程式設計、Unity 遊戲設計、JAVA AND VB 程式設計。
- 熟悉 C AND C++程式設計、WindowsServer AND Linux Debian 伺服器架設、網路管理，並能獨立完成初步的網頁設計、樹梅派、物聯網

社團

	年級	社團
社團參與	一	康輔社
	二	紫錐花社

自傳

一、家庭背景

我出生在一個和樂的家庭，父親經營餐廳，而母親則是家庭主婦。因為家中從事服務業的關係，父母從我國中畢業後，便讓我去打工以賺取生活所需的零用錢，而這也讓我養成了管理金錢的好習慣，能省則省，並能將金錢有效規劃。在我成長的歷程中，父母親的陪伴與包容，讓我能探索學習的興趣，並在其中找到我的專長。

二、求學過程

小學時父母帶我去上樂高NXT的課程，而也是我第一次接觸圖形化程式，並讓我開始對程式設計萌生了興趣。在小學五年級時因電腦課程開始接觸Scratch，而並讓我沉迷於程式設計。當時只要有時間就用Scratch寫了許多小遊戲，並因此我決定要精進程式設計。我開始主動詢問老師如何寫真正的電腦程式，並開始接觸C語言與C++。國中時為了可以開發WindowsForm而去學習C#，自那之後我便用C#寫出了多的作品。因為喜愛程式設計，故決定進入高雄高工資訊科，更在此學會了許多關於電腦與網路的新知識。

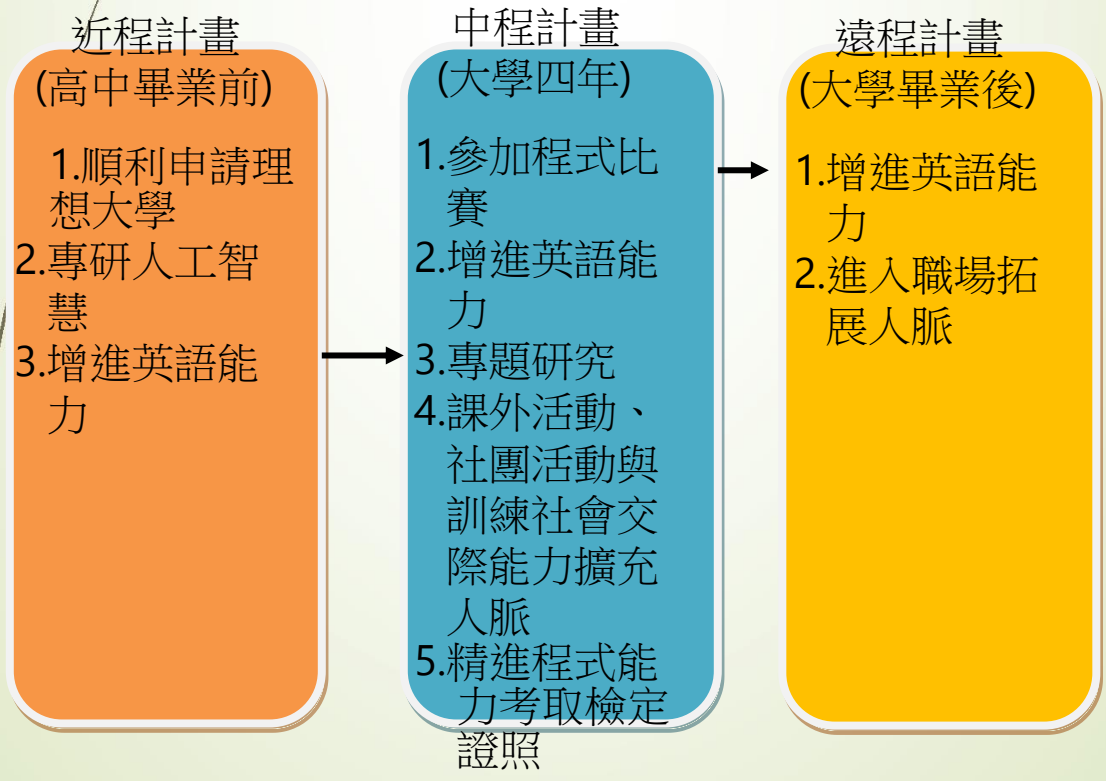
在學習程式的過程中，我了解到學習程式最快的方法是持續不斷的製作新作品，且完成之後，在新的作品便要加深難度。我從既有的應用程式作為練習對象，從中學習並改良，嘗試、嘗試、再嘗試，從不斷的練習中增進我的程式製作能力。

三、申請動機

我從小便對資訊領域有極大的興趣，因此我常常會為了學習資訊的知識與製作作品而廢寢忘食，雖然發憤忘食的狀況有時會影響生活作息，但這份對於程式創作的熱情是我生活中最為重要的事情。

在高三的時候，我剛好在貴校的資訊系網站中發現，貴校在軟體工程、網路安全、電腦網路領域進行研究引起了我的興趣，***而這也是我申請進入嘉義大學資訊工程學系的原因。***

四、讀書計畫



增進英語能力

英語作為世界通用的語言，許多的資源都是用英文寫的，若要與世界接軌，英文絕對是不可或缺的。因為我在雄工時的英文成績並不理想，所以我未來必須要更認真的練習英文能力才行。

近程計畫(高中畢業前)

人工智慧

人工智慧作為現今資訊界的新趨勢，世界上的各大公司都對其進行開發，而我這一年也對這個領域自主學習，無論從書籍或是網路資訊的學習，讓我產生更多的興趣，並且知道自我的不足。期待未來製作的遊戲能加入人工智慧，使之更為有趣。

中程計畫(大學四年)

(一)參與比賽

在我二年級時我第一次參加全國技能競賽，因為當時練習不足的緣故，我只能比到南區佳作，因此我未來我計畫挑戰更多資訊與程式競賽，從中磨練自己的專業能力。

(二)精進程式能力考取檢定證照

會更加積極的學習程式，畢業之前將會去考取大學程式能力檢定(CPE)目標為專業級(A)，還有Cisco的CCNA與CCNP，而可這些證照題目大部分都是英文寫得所以文能力也是必須的。

(三)課外活動、拓展人脈

拓展人脈對職場生涯而言相當重要。好的關係能夠讓彼此互惠，幫助彼此在工作、生涯發展都更加順利。而這個也是進入社會最為重要的一個要素，因此要多多的參與其他課外的的活動，認識更多的人。

證照

1.MTA Networking Fundamentals

MTA考試在技能競賽之前，這是我第一次詳細接觸網路的知識。

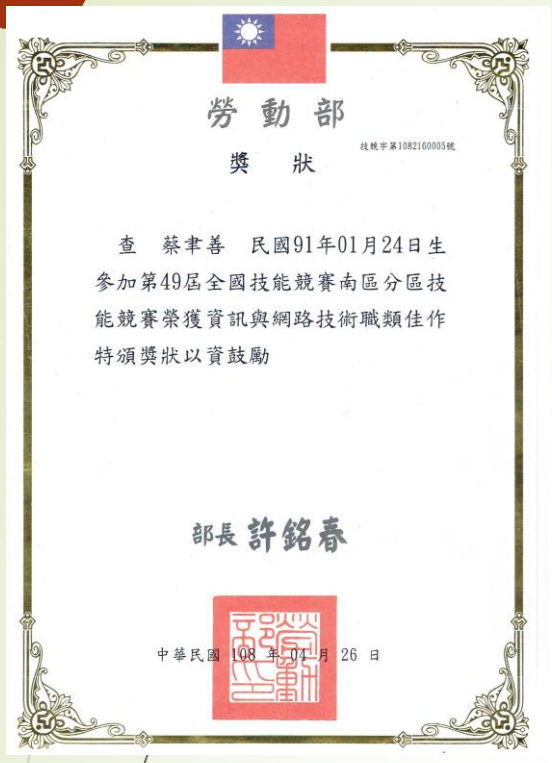
2.機器人實作能力檢定 初級

這是國中時期參加的一個檢定，讓我對這個領域有初步的認識。

3. 丙級軟體應用

4. 乙級軟體應用

獎狀



1. 第49屆全國技能競賽南區分區技能競賽 資訊與網路技術 佳作

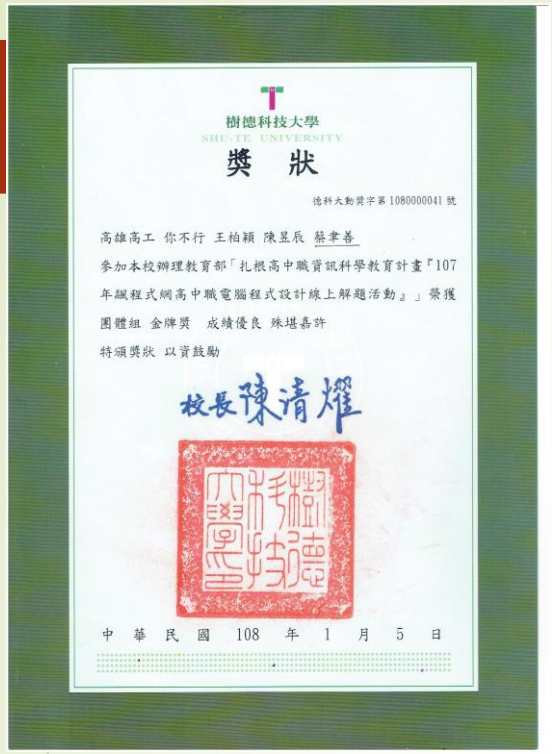
這是我第一次參加全國技藝競賽當時相 當很緊張，我也了解到我的準備不，而無法得到更好的成果。這也讓我學習到必須作更充分的準備

2. 高雄市107年中學生應用程式設計比賽 第二名

當初製作StickFightOnline是為了要參加這個比賽，而最後輸給了一個是做人工智慧下棋的人。從這時開始我便希望以後可以挑戰將人工智慧加入遊戲中。

因為這次的經驗，我將StickFightOnline重新製作，而成為了現在這份作品。

這次的比賽讓我對人工智慧開始進開始研究，並買了tensorflow的相關書籍開始著手。



樹德科技大學
SHU-TE UNIVERSITY

獎　狀

德科大勤獎字第1080000041號

高雄高工　你不行　王柏穎　陳昱辰　蔡聿善
參加本校辦理教育部「扎根高中職資訊科學教育計畫『107年飆程式網高中職電腦程式設計線上解題活動』」榮獲
團體組　金牌獎　成績優良　殊堪嘉許
特頒獎狀　以資鼓勵

校長 陳清燿

中　華　民　國　108　年　1　月　5　日

3. 扎根高中職資訊科學教育計畫『107年飆程式網高中職電腦程式設計線上解題活動』 團體組 金牌獎

高雄市立高雄高級工業職業學校獎狀
Kaohsiung Municipal Kaohsiung Industrial High School

一、姓　　名：蔡聿善
二、班　　級：資訊二乙
三、年　　度：108 年度
四、受獎事由：
參加電機電子群科「學生專題暨創意製作競賽」校內初賽，經評審獲得佳作。 特頒此狀，以資鼓勵

校長 楊秋龍

中華民國　108 年 1 月 8 日

4. 電機電子群科「學生專題暨創意製作競賽」校內初賽 佳作

高雄市立高雄高級工業職業學校獎狀
Kaohsiung Municipal Kaohsiung Industrial High School

一、姓　　名：蔡聿善
二、班　　級：資訊三乙
三、年　　度：108 學年度
四、受獎事由：
參加全國高級中等學校 108 學年度工業類學生技藝競賽校內初賽電腦修護職種，經評審獲得第二名。
特頒此狀，以資鼓勵。

校長 楊秋龍

中華民國　108 年 08 月 26 日

高雄市立高雄高級工業職業學校獎狀
Kaohsiung Municipal Kaohsiung Industrial High School

一、姓　　名：蔡聿善
二、班　　級：資訊三乙
三、年　　度：108 學年度
四、受獎事由：
參加全國高級中等學校 108 學年度工業類學生技藝競賽校內初賽電腦軟體設計職種，經評審獲得第三名。
特頒此狀，以資鼓勵。

校長 楊秋龍

中華民國　108 年 08 月 26 日

5. 工科賽校內初賽 軟體設計第三名 電腦修復工第二名

作品集

挑戰從頭到尾含伺服器自己製作的2D線上即時對戰遊戲 StickFight Online

GitHub:

ServerAPI: https://github.com/Jimmy01240397/ServerAPI
Fight GameONLINEServer: https://github.com/Jimmy01240397/Fight-GameONLINEServer

遊戲連結：
https://play.google.com/store/apps/details?id=com.Jimmiker.FightGame

線上回報單：https://forms.gle/t1VsdcFpfVdQTdK89

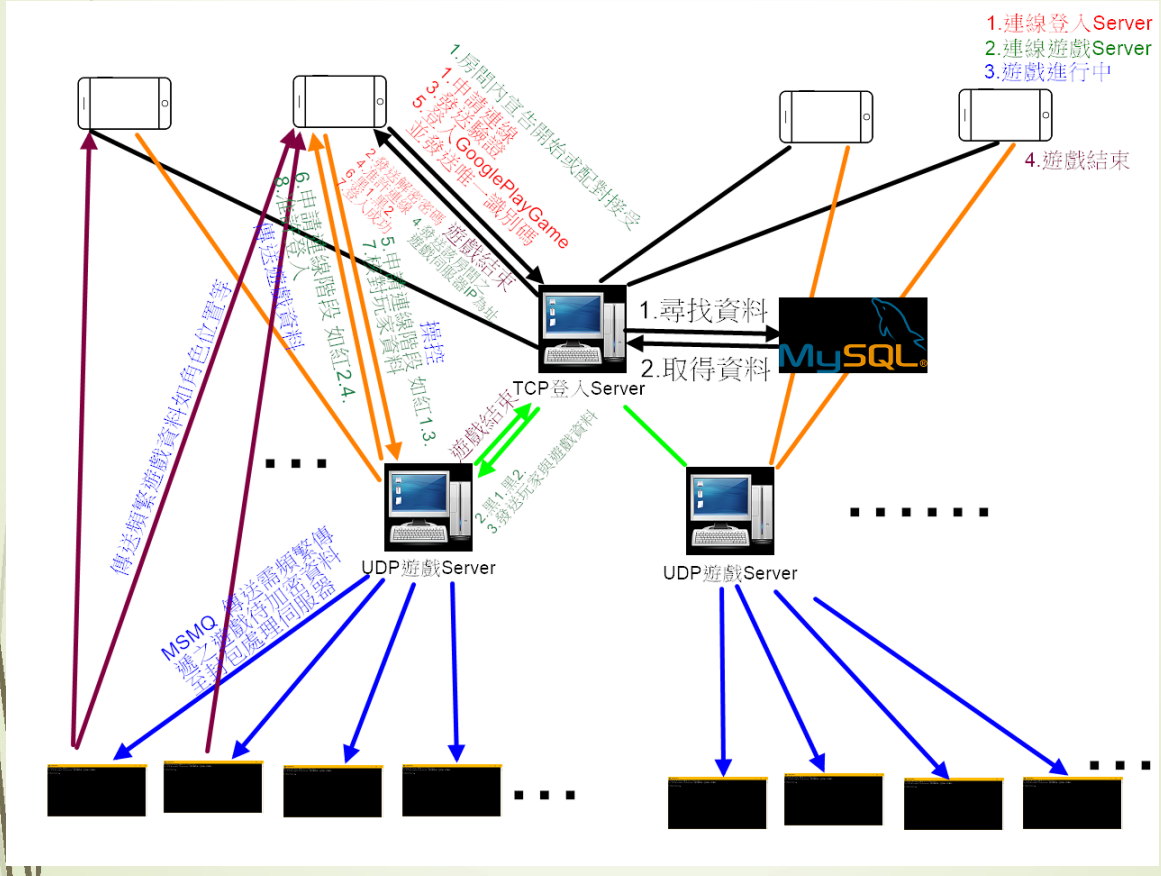
壹、製作動機

目前市面上有許多有名的線上遊戲，如LOL、鬥陣特攻、絕地求生等等，雖然這些遊戲帶給了我許多的樂趣，但也不禁得讓我感到好奇，這些遊戲的運作方式到底是甚麼，因此我決定自己從0開始設計一款線上即時對戰遊戲來進行研究。

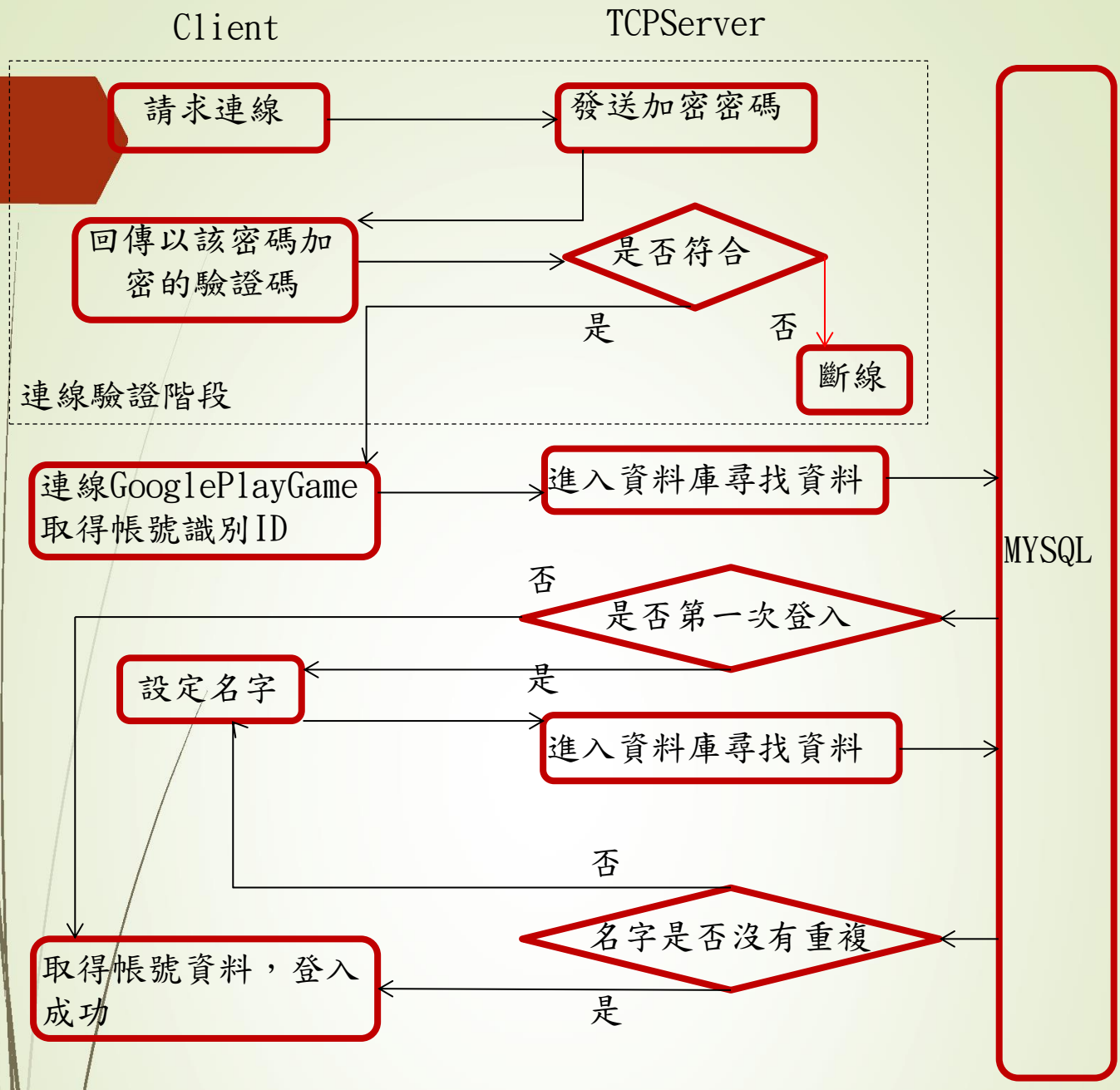
貳、製作目的

希望可以寫出一個完全出於自己製造的伺服器架構與遊戲。
希望能夠透過這款遊戲增進朋友之間的情感交流。
希望自己可以透過這一次的製作來理解主從式伺服器與線上遊戲的運作方式。

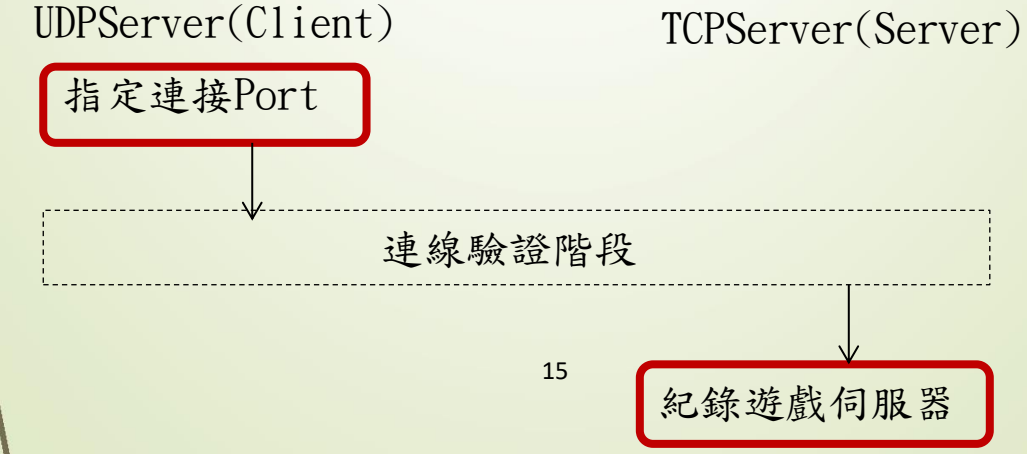
參、連線架構



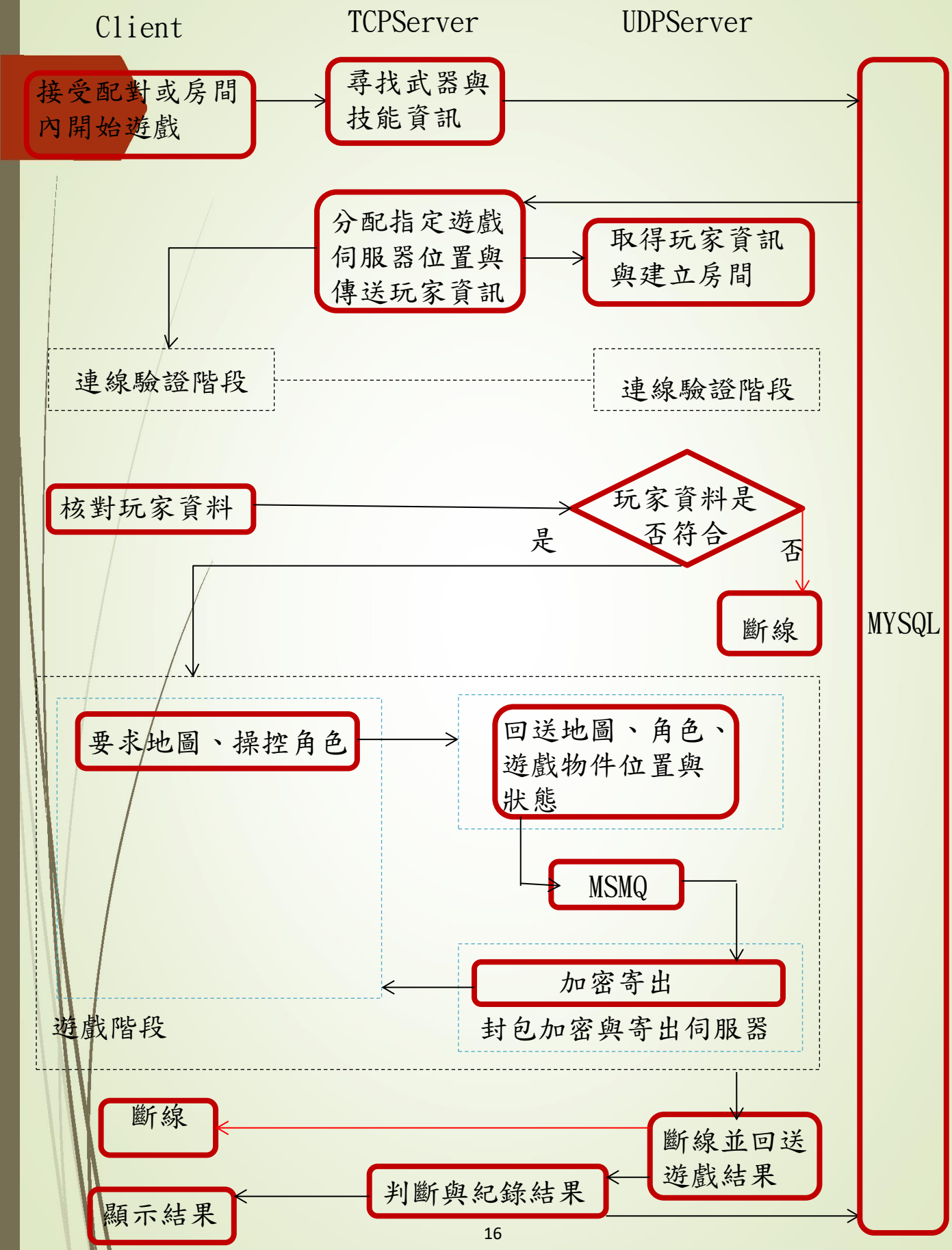
一、TCP登入伺服器部分



二、TCP登入伺服器與UDP遊戲伺服器連線部分(以下為TCP連線)

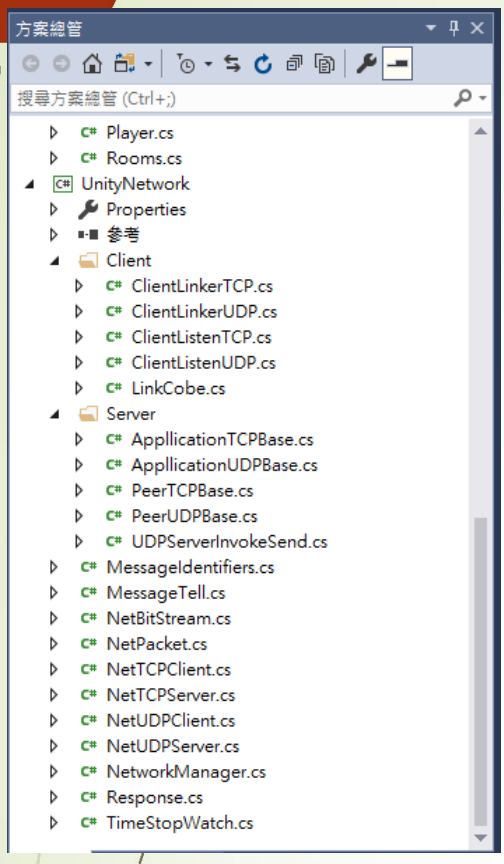


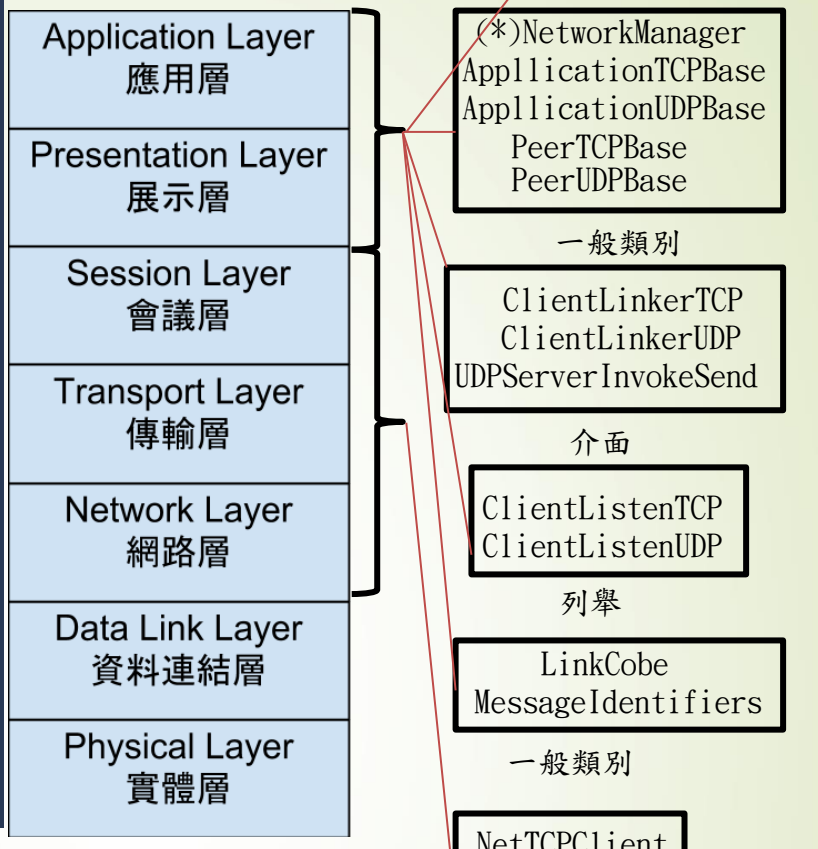
三、UDP遊戲伺服器部分



肆、製作方法

參考書本與網路，使用System.Net.Socket下的Socket、TcpListener、TcpClient、UDPClient、建立一個完整的伺服器與客戶端框架UnityNetWork.dll。

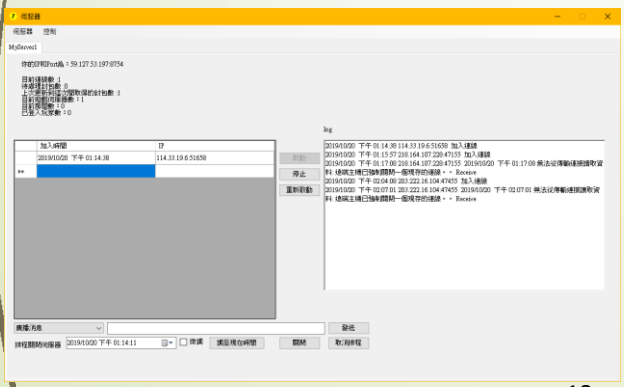




一、下層之一般類別
用於做選擇傳輸方式、連線驗證階段、非同步封包寄送與接收。
二、上層之一般類別
用於給予客戶端進行連線與UDP封包加密與寄出伺服器進行傳輸資料。
三、繼承用類別
用於給予伺服器端繼承用。
(一) NetworkManager為Application系列與ClientLinker系列的最上層父類別，給予最基本的控制架構。

（二）ApplicationBase 系列為給予伺服器控制中心進行控制用的父類別。
（三）PeerBase 系列為伺服器用來代表一個客戶端之類別的父類別。
四、介面
給予客戶端類別的介面架構。
五、列舉
LinkCobe 為目前連線狀態之列舉 MessageIdentifiers 接收到之資料種類之列舉。
六、資料包裝
內部消息會外部封包。
（一）NetPacket 代表一個資料。
（二）NetBitStream 只有 NetPacket 內含封包時才會使用，代表一個封包。
（三）Response 只有伺服器使用 Reply 或 Tell、客戶端使用 ask、發送加密密碼或者對外宣布斷線時才會使用，代表一個傳送的主要資訊。
值得一提的是 Response 原本是打算只有在伺服器做回傳資料時使用，但後來為了方便所以才讓他可以用在其他地方，而類別名稱也因為一旦更改許多其他地方也得改的緣故而決定不去更改類別名稱。

用 UintyNetWork.Server 命名空間中的 TCP 類別做一個 WindowsServer 程式，並且另外做一個視窗主控台執行登入伺服器。



遊戲伺服器則因為需要運用 Unity 的物理運算，因此使用 Unity 與 UintyNetWork.Server 命名空間中的 UDP 類別製作一個 PC 端遊戲代替遊戲伺服器。為何使用 UDP？那是因為遊戲伺服器需要頻繁發送地圖、角色、遊戲物件位置與狀態資訊，相比 TCP，UDP 更為節省時間因此選用 UDP。

為何需要另外準備封包加密與寄出伺服器？那是因為封包在家密語壓縮的時間過久。因此先將未加密的 Bytes 資料存進電腦中的 MSMQ 中將封包分給其他處理程序進行加密壓縮並發送封包以減少單一處理程序之工作壓力。客戶端必須以 UDP 連線遊戲伺服器與所有封包加密與寄出伺服器。

加密方式：使用一組 16 進位的 Key 並且將封包的 Bytes 轉為 16 進位，在開始加密前以 2 進位從 00~11 中選擇一個數字所代表的兩種加密方式進行加密並記錄在封包的開頭：

第一格 0:+
　　1:-
第二格 0:*
　　1:/

將封包依照選擇的加密法用 Key 進行加密後會取得的封包大小為原本的封包的兩倍，之後以 System.IO.Compression 中的 GZipStream 類別作壓縮。

資料序列化：原本序列化方式是想用 Serializable 進行序列化，但是過程中我發現.NET Framework 與 Mono 也就是 Unity 程式所進行的序列化得出來 Bytes 是不同的，因此決定用 MemoryStream 與

BinaryWriter 進行序列化其中最難處裡的毫無疑問是如何以最短的封包長度去表示最多的資料。
SQL 隱碼：這個算是最重要的一個部份了，如果沒做好防範最嚴重可能會造成所有資料被竊取或刪除。防範方法：將一些敏感的字元或字串以另一種字串格式代替，例如： \n => [_\\n_]、[=> [_[_]、DELETE => [_D_E_L_E_T_E_]

伍、最後功能

遊戲影片：
https://drive.google.com/open?id=1mBYBDzDptEq6qAMsUmkXnhrDje7h2mic

進入遊戲後以 GooglePlayGame 登入並取一個好聽的名字。之後便可以選擇創建房間或配對遊戲，在此之前也可以去好友列表與別人建立好友關係，或者是去裝備調整進入遊戲時的裝備，也可以進入設定調整操控模式或填寫遊戲問題回報單。

陸、展望

雖然目前下載人數因宣傳方法有限因此並不多，但未來將會尋找更多的宣傳方法以讓更多人可以看見這款遊戲。同時製作更多的新模式新功能與新武器技能。

專題製作

聲控家電小幫手

GitHub: https://github.com/Jimmy01240397/Home

摘要

本研究利用 樹莓派(Raspberry Pi) 連接網路，設計手機 APP 利用 Google 語音辨識物件，來達成口說控制家電功能。

一、樹莓派(Raspberry Pi)

二、Google 語音辨識

三、Xamarin

我們利用手機內設計好的 APP 程式，當使用者要控制家中電器物品時，只要開口說出指令，系統即會做出相對應動作

例如：對使用者用說出「(開啟/關閉)X 樓電燈」或「(開啟/關閉)屋頂風扇」，系統會收取「X 樓電燈 或 屋頂風扇」與「開/關」的關鍵字進而完成相對應動作。

而當有人進入家中時，家中的監控器就會發出訊息到手機 APP 裡，此時手機 APP 就會發出通知告訴使用者「有人闖入」。

而當使用者想查看家中狀況時可以對 APP 說「開啟攝影機」此時家中的監視器就會傳送影片到手機 APP

壹、研究動機

在現今的生活中，有許多需要控制的家電或是器具，因此發展出了許多的遙控裝置，但也產生了一些麻煩，比如無法及時的做到控制的動作，或是找不到指定遙控裝置，諸如此類的問題。

你是否有過這樣的經驗？找不到遙控器，或者是找到了遙控器卻發現它沒有電了，亦或是家中的長輩因為家裡太多遙控器而

分不清，或者是看不懂上面的指令而不會使用。這些都會造成我們生活上的困擾。

要如何解決這個問題呢？ 利用學 Xamarin 我們設計了一套聲控裝置 APP 程式，就能夠輕鬆的運用語音來操作我們家中的電器設備。基電實習課程中室內配線單元，也剛好運用在本次研究中周邊 I/O 控制電路的接線上，正所謂學以致用就是這個道理。

我們也希望這樣的設備能夠帶給人們更方便的生活環境，甚至是以後能擴展到更多的家電上。科技始終來自於人性，有這樣的需要，必然會有這樣的發明。

貳、研究目的

所以我們需要一種隨手可得，隨時用來控制的裝置，在現在的社會，可以說是人手一機的世界，而手機的應用性即可成為一支隨時可用的萬能遙控器，只要進入了無線 Wi-Fi 的收訊範圍，就能控制其中設定好的器具。

基於研究動機，我們想到了用樹莓派(Raspberry Pi)和隨身攜帶的手機，設計出一套利用口說控制家中電器的設備。我們使用樹莓派(Raspberry Pi)，連接家中無線網路，開啟手機程式 APP，用說的來控制電器。

以下是我們的研究目的：

(一)、研究如何語音控制家電。

(二)、研究樹莓派(Raspberry Pi)。

(三)、研究手機 APP 的程式設計 Xamarin 用法

參、研究設備與器材

表 1 材料

材料名稱	規格/大小	數量	備註
電源供應器	GPD3303	1	
尖嘴鉗		1	
斜口鉗		1	
樹莓派(Raspberry Pi)		1	
轉壓器	3V→5V	1	
風扇	5V~50/60Hz	1	
麵包板	2P	1	
LED	直徑 3mm	各 1	
杜邦線	公對公/公對母	共 14	

表 2 硬體設備

硬體名稱	規格型號	數量	備註
手機	HTC Desire 12+	1	

表 3 軟體設備

軟體名稱	規格型號	數量	備註
作業系統	Raspbian	1	

應用軟體	Visual Studio 2017	1	

肆、研究過程或方法

一、口說控制家電概念圖

此構想是用到手機的 APP，並且應用 Google 的語音輸入頁面來當作指令的判別，在判別所輸入的指令後上傳到指定的無線網路 IP，然後透過樹莓派(Raspberry Pi)的程式來做出開/關燈/風扇等動作。

二、樹莓派(Raspberry Pi)

樹莓派(Raspberry Pi)，是一款基於 Linux 的單晶片電腦。配備一枚博通（Broadcom）出產的 ARM 架構 700MHz BCM2835 處理器，256MB 記憶體（B 型已更新到 512MB 記憶體），使用 SD 卡當作儲存媒體，且擁有一個 Ethernet、兩個 USB 埠、以及 HDMI（支援聲音輸出）和 RCA 端子輸出支援。樹莓派面積只有一張信用卡大小，體積大概是一個火柴盒大小，可以執行像《雷神之鎚 III 競技場》的遊戲和進行 1080p 影片的播放。操作系統採用開源的 Linux 系統：Debian、ArchLinux，自帶的 Iceweasel、KOffice 等軟體，能夠滿足基本的網路瀏覽、文字處理以及電腦學習的需要。如圖 2。

圖 2 樹莓派(Raspberry Pi)

三、口說控制家電系統設計

由樹莓派(Raspberry Pi)接收來自手機 APP 程式所傳送出的指令，控制家電開關動作。系統設計流程規劃如圖 8。

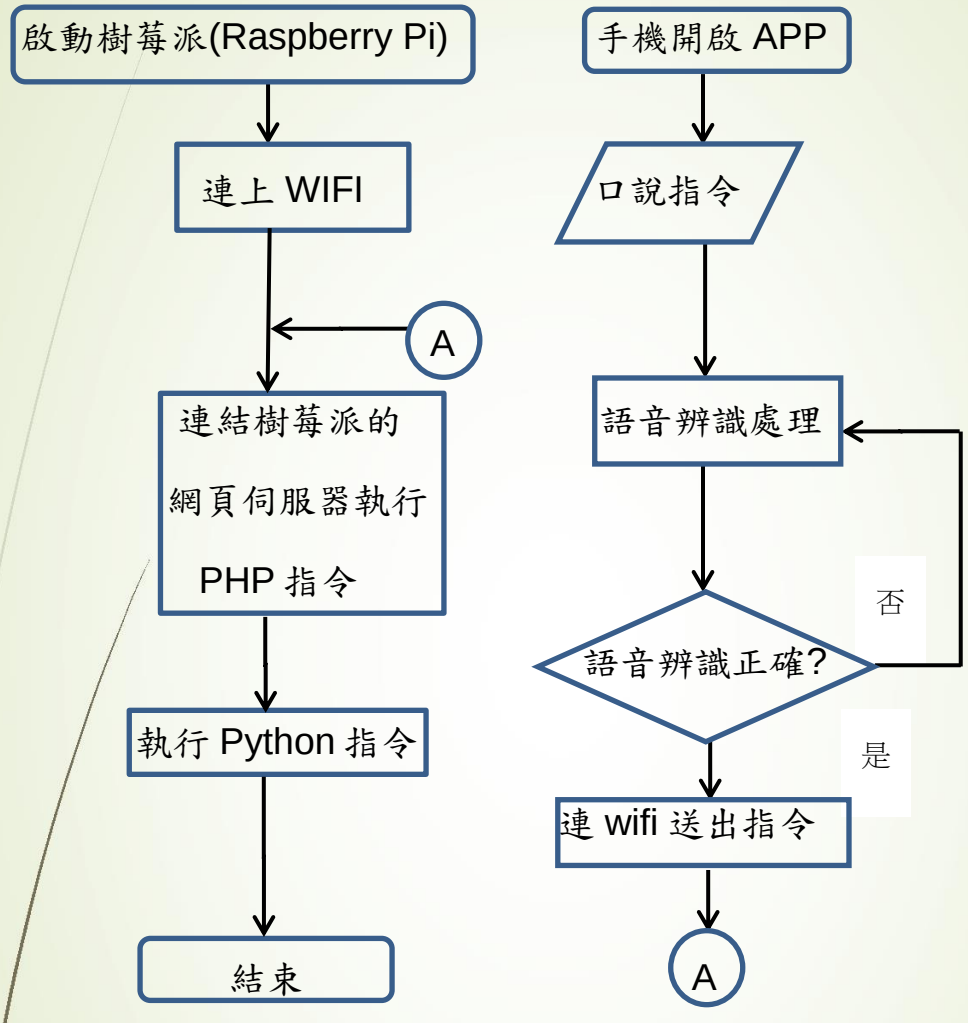
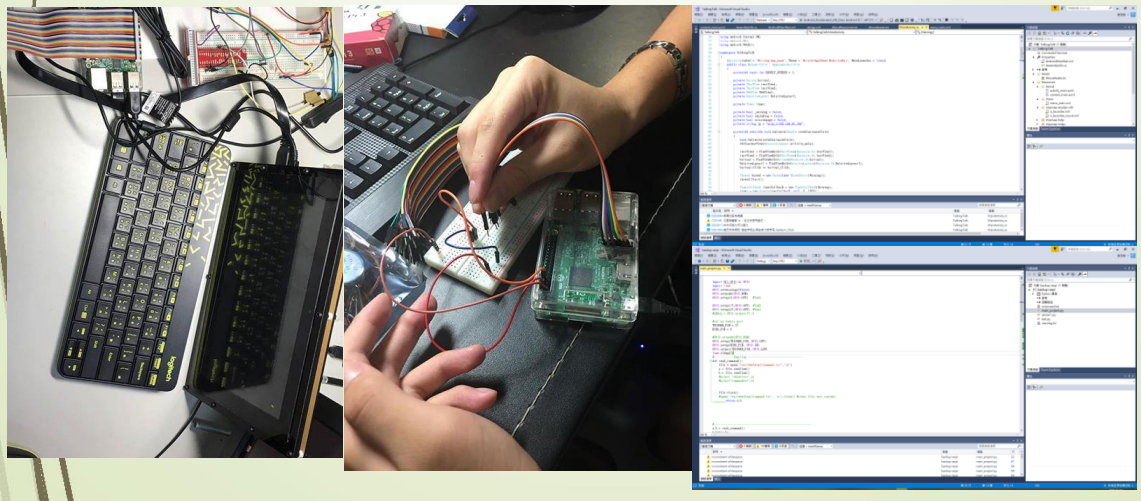


圖 8 系統設計流程圖

四、製作過程



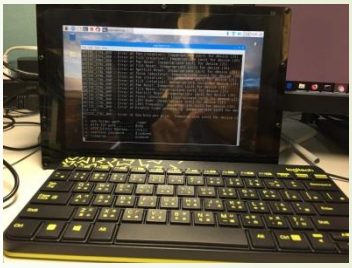
伍、研究結果

一、完成手機 APP 語音辨識機制

(一)開啟手機端 APP 程式。

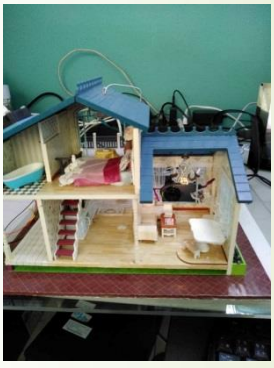
(二)按一下說話按鈕說出指令動作。

(三)說出指令 ex：開啟一樓電燈、關閉一樓電燈、開啟屋頂電扇等。



二、完成口說家電控制系統

利用樹莓派(Raspberry Pi)，當接收到手機傳來的語音指令，樹莓派(Raspberry Pi)將指令解碼後，將電力傳到家電上。整個系統完成。



陸、結論

一、特色

物聯網是現代社會中相當重要也是發展快速的一環，我們的裝置也是一種物聯網的雛形，它將來能做更廣闊的運用，諸如情境燈、電冰箱或者冷氣的溫度控制等等，都是相當方便的設計。在各種工作場合或家庭，也都能實現智慧生活的理想。

此裝置在遠處即可達成控制電器的動作，增加了它的可用性，且手機隨身帶著，指令隨時可以傳達，增加了它的高即時性，而它也相當輕便，小巧的電路即可達成。這樣的裝置將會有十分廣闊的前景與應用。

二、未來研究

(一)、運用在工廠：巡視廠房時，能及時開關所需之器具，增加安全性與方便性。

(二)、對語音輸入辨識語意能夠人性化，例如當燈太亮時，只要說出暗一點，則燈就會將亮度自動調低一點，反之亦同。

三、現有功能之改進

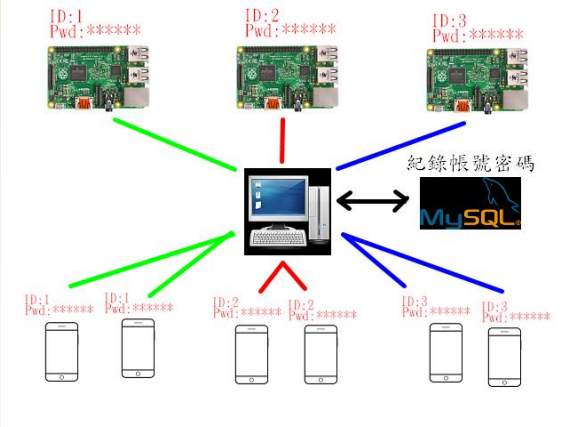
(一)、關鍵字誤差。因聲控是藉由抓關鍵字來啟動、終止動作，所以請勿用裝置無法正確辨識的話語。（ex：不要開燈、不要把風扇關掉）

(二)、連接品質。希望可以把連接的距離增加，還有讓連間的穩定度。

PS、結束專題研究後對於該程式之延伸與改良

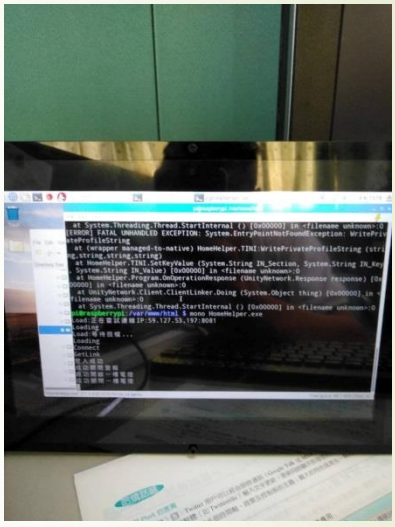
在結束該專題研究之後，我依然針對該系統進行研究與改良，並且發現了以下缺點。

因為該研究之樹梅派的家電控制的方式是以網頁伺服器並執行 PHP 指令以執行動作，因此若是要在外面進行連線必需要輸入指定 IP 位址，因此必須以寬頻連線進行連網，若要使用 DHCP 則必須要於上層 IP 分享器　　或路由器進行 port forwarding，而樹莓派的防火牆也必須開啟 80Port 才能進行連線。況且一般家庭所使用的 IP 位址皆為動態 IP 使得連入的難度又上升了，除非申請固定 IP 或者是動態域名服務。



解決方法、

運用作品一中的 UnityNetWork.dll 中的 TCP 連線系統與資料庫建立一個主從式伺服器架構，客戶可以在樹莓派中建立新帳戶並且在手機上進行登錄，如此一來我們便只需要記得自己的帳號密碼即可。



首先須在樹梅派中安裝 Mono 以執行 C#程式，安裝完成後將式沒派之客戶端打開連線之後便可以進行註冊與登入的動作，在第一次登入完成後將會輸出存放帳號密碼的檔案，以便之後可以不必再次登入。樹莓派完成登入動作後即可於手機端進行登入，之後動作便與之前相同。

缺點

一、若是伺服器停止運作，使用該連線的所有樹梅派也將停止運作。

二、安全性的疑慮，若是被別人知道帳號密碼，便有權限控制其所有家電。

三、無法使用原本的網路攝影機，原本的攝影機是使用網頁的方式去運行的，但是新　的 系 統因為並非使用網頁，因此必須使用其他辦法連線網路攝影機，方法目前尚未找到。

搜索工具

手機自動追蹤器

GitHub: https://github.com/Jimmy01240397/PhonePOS

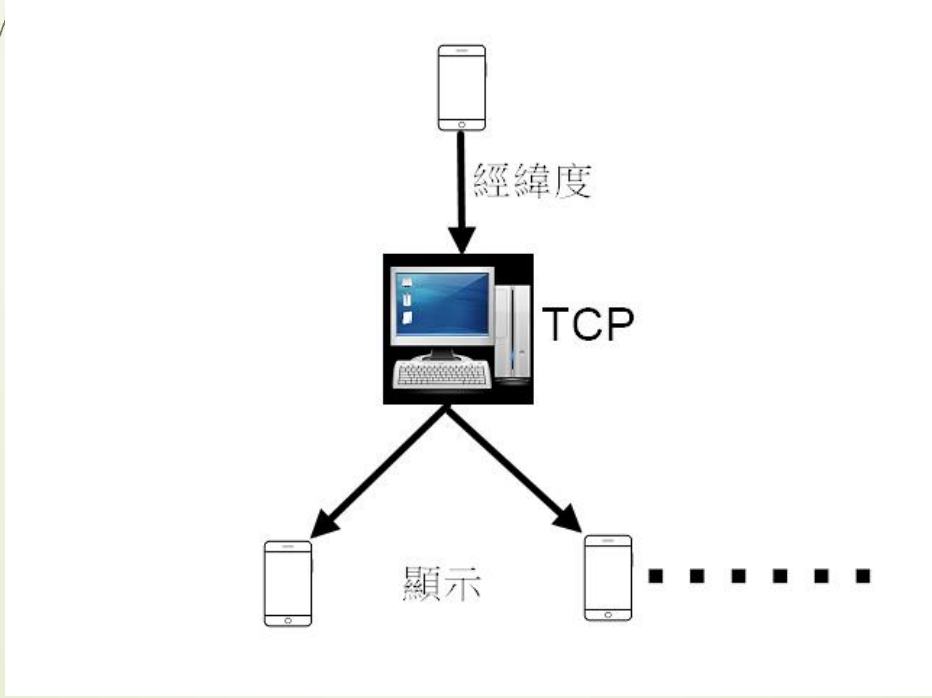
壹、製作動機

當手機不見了的話該怎麼辦？打電話？又或者是用Google的尋找手機功能呢？為了能夠方便尋找手機我希望可以製作一個能夠隨時定位手機的程式方便一時時能夠找回。

貳、製作目的

透過Android Studio製作出一個可以用開機、收到電話等事件啟動背景服務的無視窗APP，發送目前手機位置到伺服器並將該位置消息傳送至另一支手機的接收端APP並在上面的地圖顯示位置。

參、連線架構



肆、伺服器

因為這次是以Android Studio製作因此伺服器必須要從頭重新以Java再製作一次。為了降低其複雜程度，該程式僅用兩個類別實現簡易伺服器的功能。

一、Server 是代表Server的主程式主要負責接受新連線。

二、MyThread1 在Server成功建立新連線後，便會新增一個負責同步接收與寄出資料的線程，也就是說這個代表一個客戶端。

伍、客戶端

一、發送器：

廣播接收器：用接收手機事件如開機、收到電話。

服務：收到事件後所啟動的服務，每三秒一次將目前的GPS位置資訊傳送至TCP伺服器。

主畫面：其實這原本是沒有要做的，但後來試過很多次發現廣播接收器似乎無法直接啟動服務，因此我製作了一個空頁面給廣播接收器呼叫，當打開該頁面後便會立即啟動該服務並且關閉該頁面。若是該手機關閉GPS則會自動跳轉至設定讓使用者開啟GPS，若是沒開啟權限也會自動跳出啟動權限之畫面。

二、接收器：

使用GoogleMapAPI顯示地圖並將接收的位置資料顯示在地圖上，並在最上方顯示經緯度與複製經緯度的標題與按鈕。

隨身碟網路備份工具

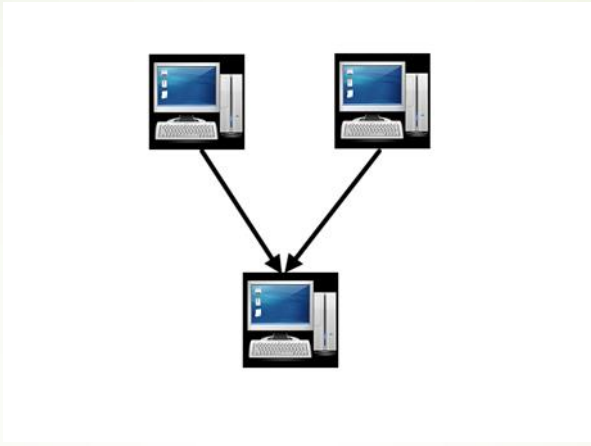
GitHub: https://github.com/Jimmy01240397/BackUp

壹、製作動機與目的

為了避免隨身碟的資料損失，希望可以設計一個自動備份程式。

能夠讓隨身碟街上時可以自動將其內部資料備份至伺服器中的資料夾中。

貳、連線架構

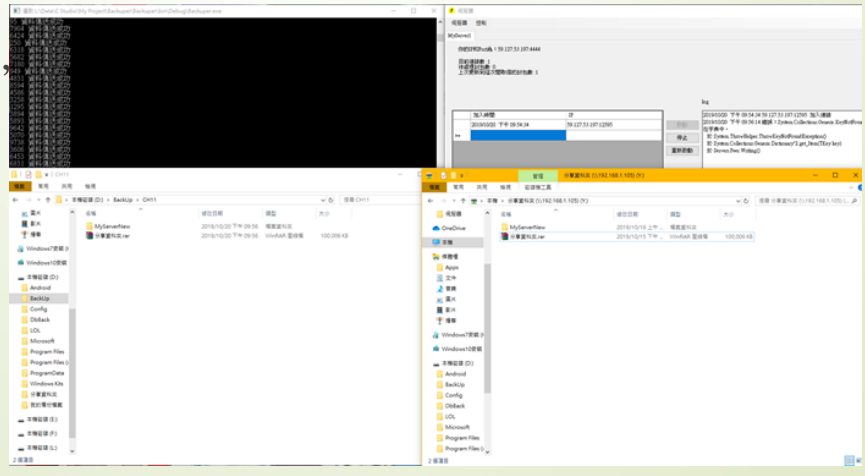


參、製作方法

運用作品一中的UnityNetWork.dll 中的 TCP連線系統，製作一個可以接收檔案的伺服器，與發送檔案的Console客戶端。在客戶端中的隨身碟插入時，該磁碟機編號便可以讀取裡面的內容，判斷指定磁碟機編號是否可以讀取以執行傳送，將內部檔案每1MB進行拆分並與資料夾目錄封包編號一起進行傳送，並在伺服器端重新組合。

肆、運作結果

使用網路磁碟機模擬隨身碟，成功運作。



簡易C、C++ IDE

GitHub: https://github.com/Jimmy01240397/CSharpMaker

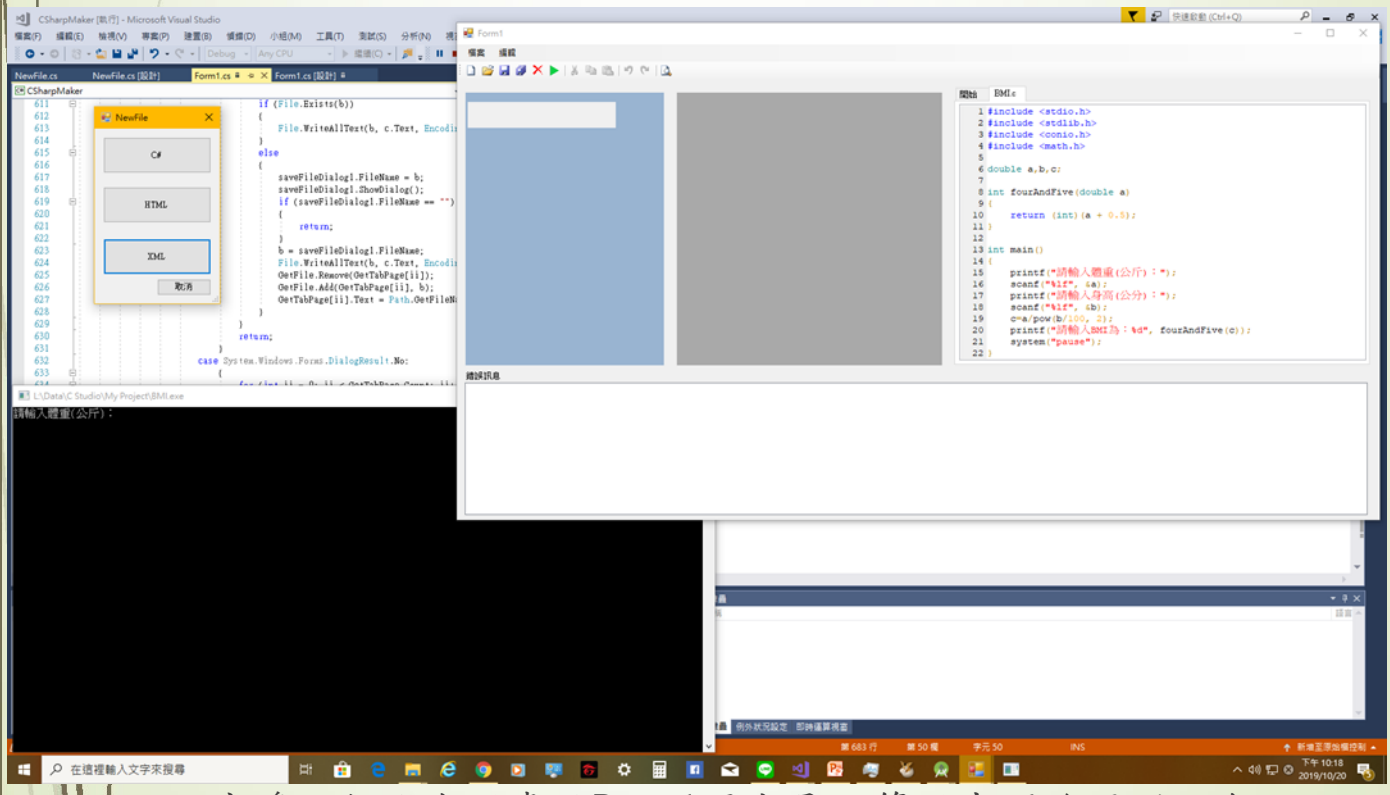
壹、製作動機與目的

希望可以製作屬於自己的C、C++開發介面。

貳、製作方法

使用ScintillaNet.dll製作一個程式編輯介面並運用gcc編譯器時線編譯功能。

參、運作結果



至於旁邊兩個沒有用處的Panel原本是打算用來開發圖形化介面，但後來發現要開發這個太花時間和成本因次放棄該開發。

PictureBox簡易模擬碰撞器

GitHub: https://github.com/Jimmy01240397/PictureMove

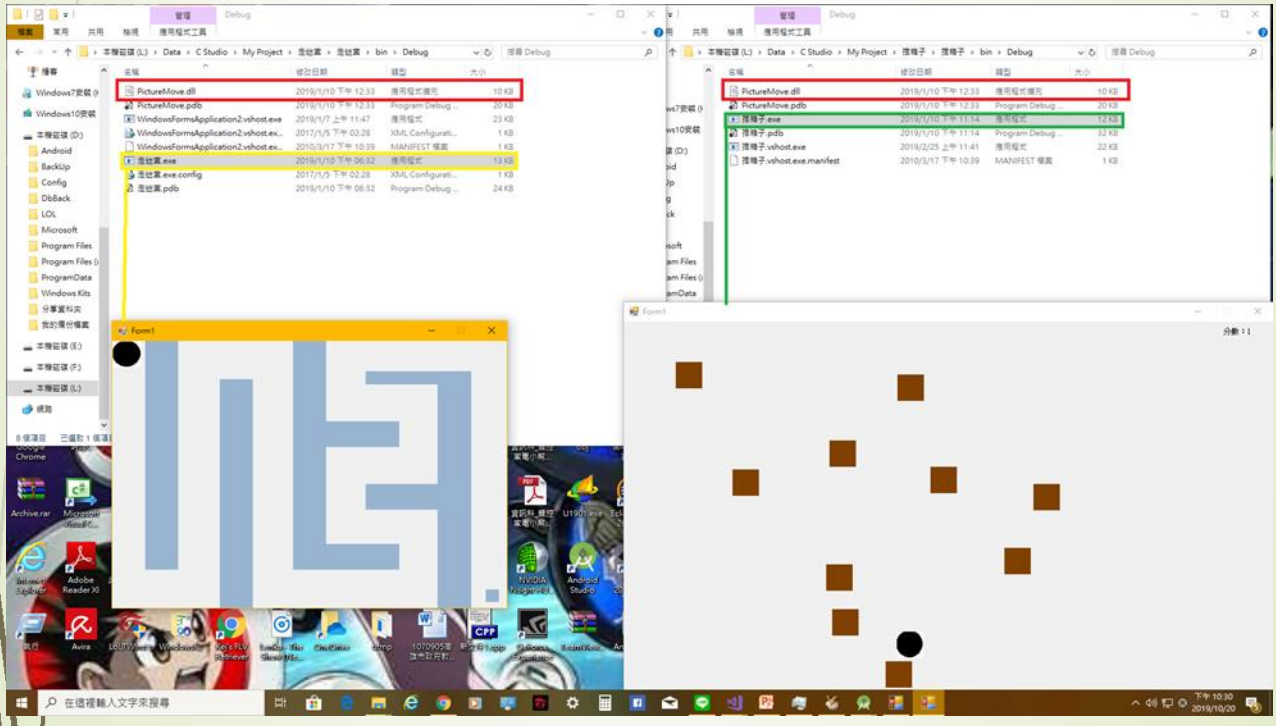
壹、製作動機與目的

希望可以製作一個類似於Unity碰撞器的簡易版。

貳、製作方法

製作一個新類別繼承至PictureBox，在其中寫方法用來運算該PictureBox之長寬與位置作為碰撞區並運用Timer進行測量該PictureBox之父容器中是否有與其他PictureBox碰撞。

參、運作結果

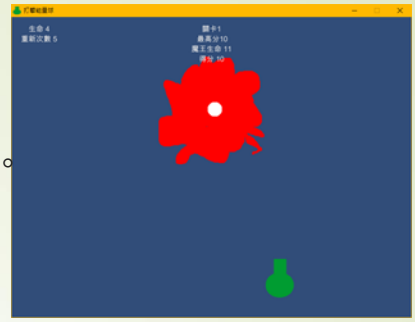


以上為兩個以該物件為基礎所做的小遊戲，左為走迷宮，右為推箱子。

Unity 簡易平面射擊遊戲

GitHub: https://github.com/Jimmy01240397/JimmyShoot

壹、製作動機與目的



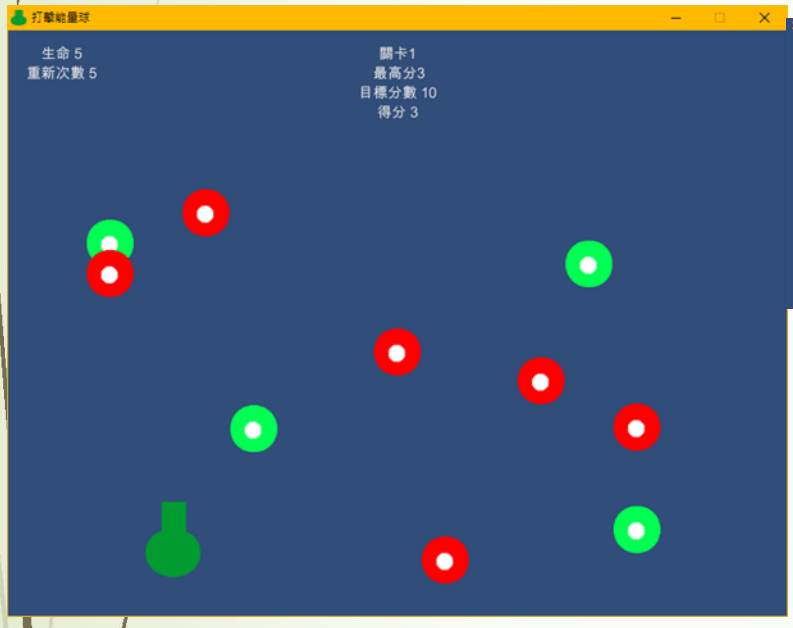
希望能藉由此遊戲來熟悉Unity的方式。

貳、製作方法

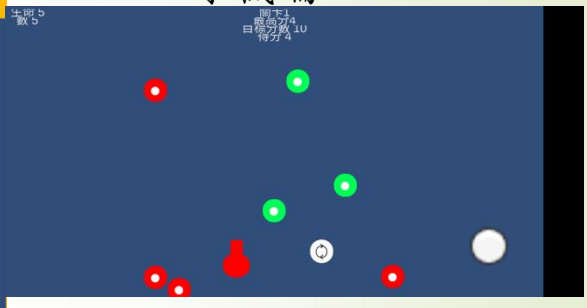
參考網路與書本製作第一個Unity遊戲。

參、運作結果

PC端



手機端



按下左右方向鍵或滑動螢幕進行移動，並且按下空白建或螢幕上的按鈕發射子彈。必用正確顏色的子彈擊中正確顏色的敵人否則敵人會瞬間降落，達一定分數後會進入魔王模式，打倒魔王則可進入下一關敵人顏色會隨關卡越來越多種。

運用路徑長度限制所製作的簡易資料夾封鎖

GitHub: https://github.com/Jimmy01240397/Folder-Locker

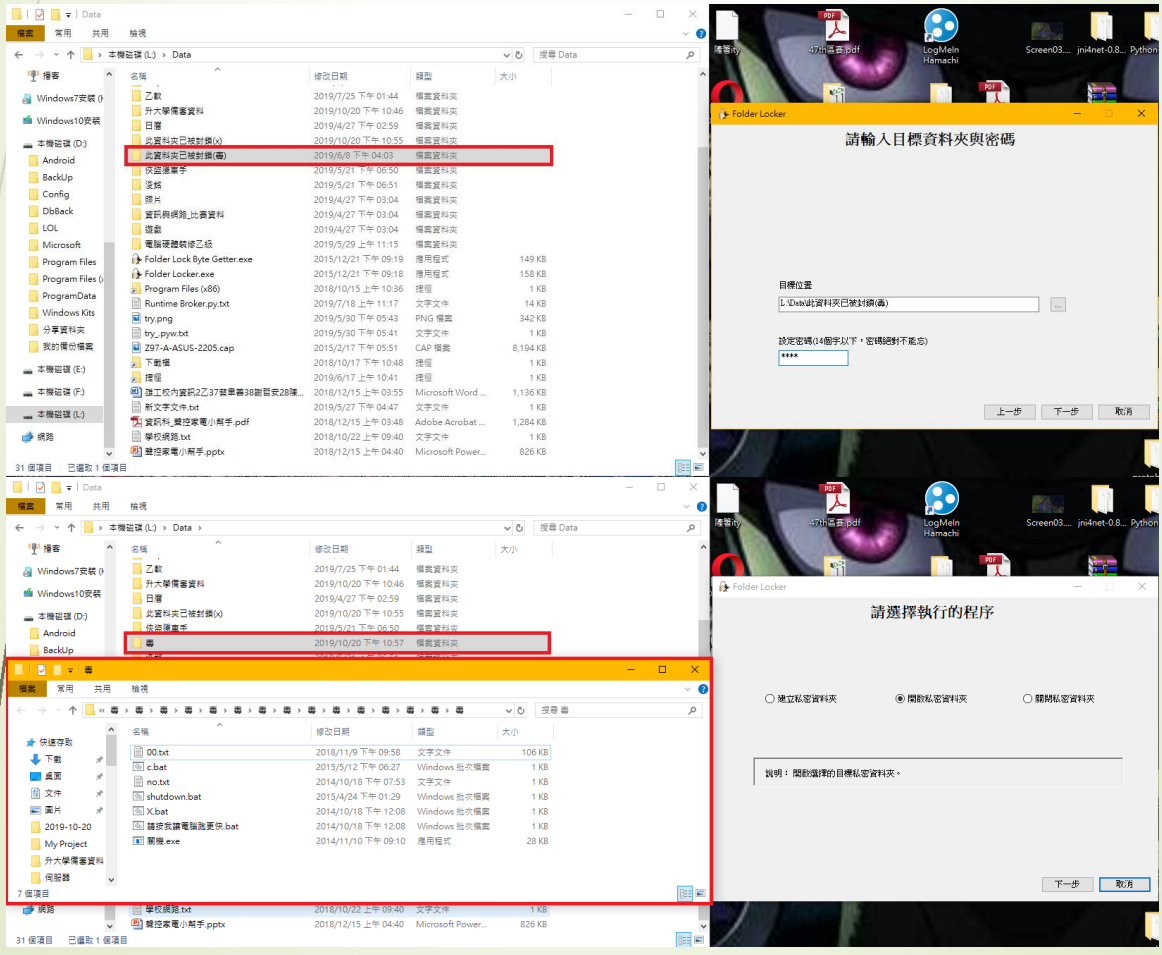
壹、製作動機與目的

在網路上發現運用此原理所寫出的bat檔，希望可以用C#改良該bat檔，並且透過這個研究熟悉C#的語法。

貳、製作方法

參考網路與書本學習C#的IO操作。

參、運作結果



簡易計時器

GitHub: https://github.com/Jimmy01240397/Time-Counter

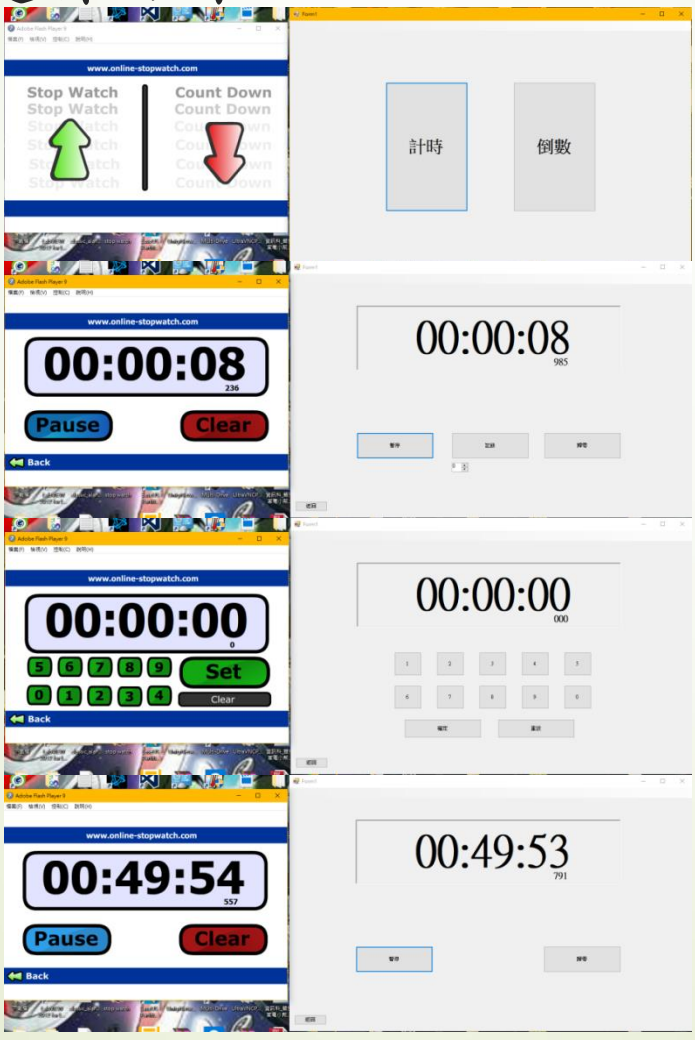
壹、製作動機與目的

希望可以藉由重現網路上的計時程式stopwatch.exe來熟悉C#的基本語法。

貳、製作方法

參考網路與書本學習C#的基本操作。

參、運作結果

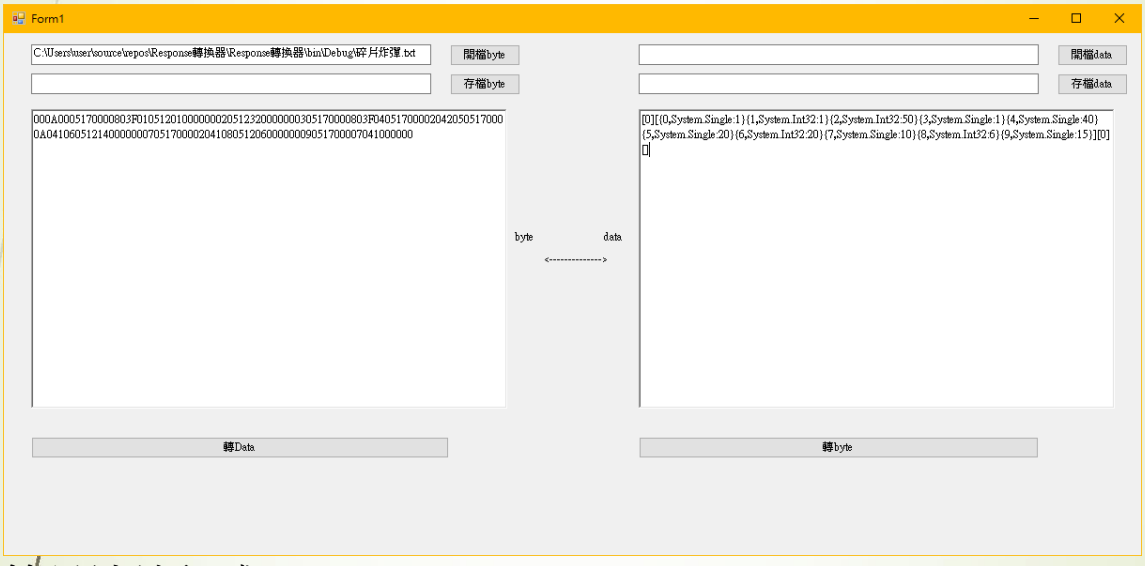


附錄

以下為幾個比較重要的作為工具用的作品

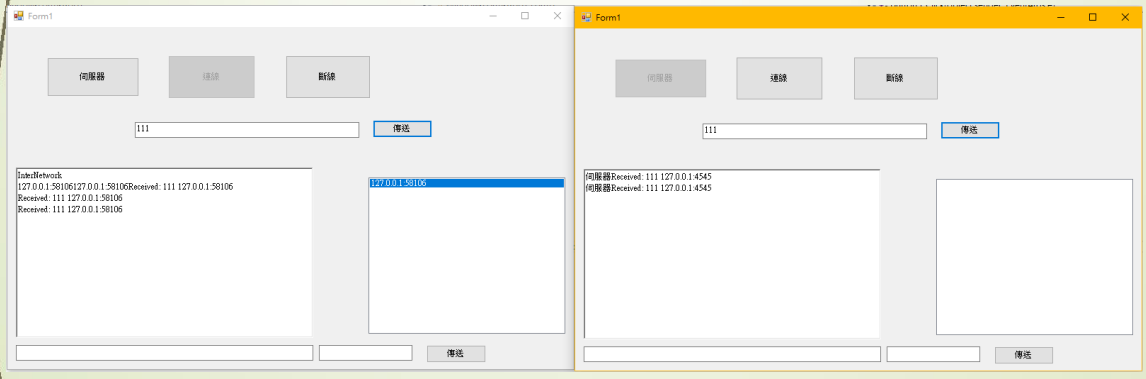
Response 轉換器

將該類別所要序列化之資料以自訂的格式字串顯示出來，若要修改其序列化的內容只需要將原本的 Bytes 輸進去做轉換後在該介面做修改即可。



簡易連線程式

用來測試 C#Socket 功能。



最近研究

Unity ML-Agent 遊戲人工智慧

簡介：

Unity Machine Learning Agents (ML-Agents)是一款開源的 Unity 插件，使得我們得以在遊戲環境和模擬環境中訓練智慧 agent。我們可以使用 reinforcement learning（強化學習）、imitation learning（模仿學習）、neuroevolution（神經進化）或其他機器學習方法，通過簡單易用的 Python API 進行控制，對 Agent 進行訓練。我們還提供最先進算法的實現方式（基於 TensorFlow），讓遊戲開發者和業餘愛好者能夠輕鬆地訓練用於 2D、3D 和 VR/AR 遊戲的智慧 agent。這些經過訓練的 agent 可用於多種目的，包括控制 NPC 行為（採用各種設置，例如多個 agent 和對抗）、對遊戲內部版本進行自動化測試、以及評估不同遊戲設計決策的預發布版本。ML-Agents 對於遊戲開發者和 AI 研究人員雙方都有利，因為它提供了一個集中的平台，使得我們得以在 Unity 的豐富環境中測試 AI 的最新進展，並使結果為更多的研究者和遊戲開發者所用。

其源碼中有一個用來繼承用的 Academy、Agent 與 Brain 類別用來提供訓練的開發環境，其中

Agent -它可以被附加到一個 Unity 遊戲對像上（場景中的任何角色），負責生成它的觀測結果、執行它接收的動作並適時分配獎勵（正/負）。每個 Agent 只與一個 Brain 相關聯。

Academy -它指揮 agent 的觀測和決策過程。在 Academy 內，可以指定若干環境參數，例如渲染質量和環境運行速度參數。Communicator 位於 Academy 內。

Brain -它封裝了 Agent 的決策邏輯。實質上，Brain 中保存著每個 Agent 的 policy，決定了 Agent 在每種情況下應採取的動作。更具體地說，它是從 Agent 接收觀測結果和獎勵並返回動作的組件。

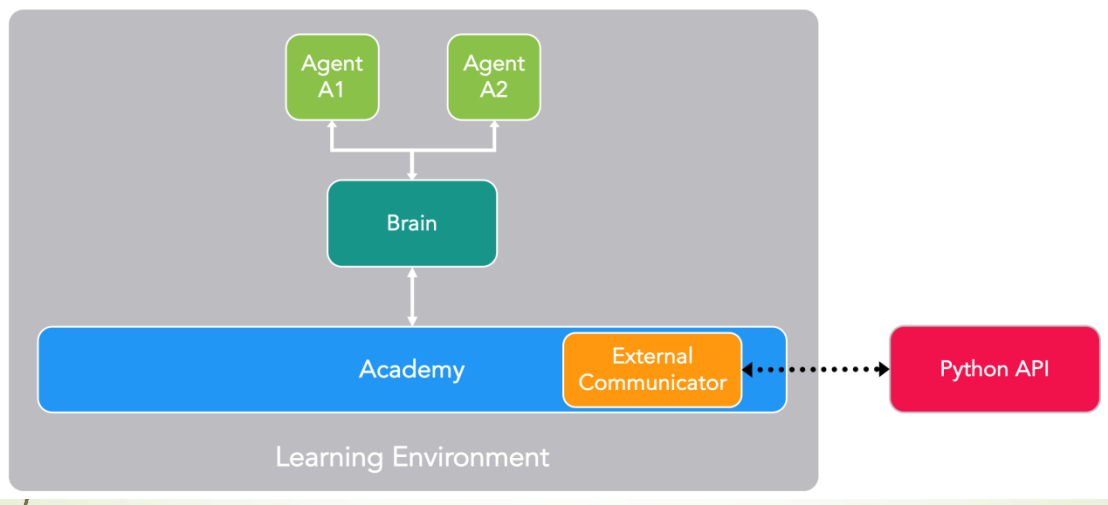
每個學習環境都會有一個全局的 Academy ，與每一個遊戲角色一一對應的多個 Agent。雖然每個 Agent 必須與一個 Brain 相連， 但具有相似觀測和動作的多個 Agent 可關聯到同一個 Brain。

而源碼中亦有提供繼承了 Brain 的類別，分別是 PlayerBrain、LearningBrain 與 HeuristicBrain。

PlayerBrain -使用鍵盤或控制器的實際輸入進行決策。這種情況下，人類玩家負責控制 Agent，由 Brain 收集的觀測結果和獎勵不用於控制 Agent。

LearningBrain -使用 Python API 進行決策。這種情況下， Brain 收集的觀測結果和獎勵通過 Communicator 轉發給 Python API。Python API 隨後返回 Agent 需要採取的相應動作。亦可選擇將訓練完成之嵌入式 TensorFlow 模型進行決策。嵌入式 TensorFlow 模型包含了學到的 policy，Brain 直接使用此模型來確定每個 Agent 的動作。

HeuristicBrain -使用寫死的邏輯行為進行決策，目前市面上大多數遊戲角色行為都是這麼定義的。這種類型有助於調用具有寫死邏輯行為的 Agent。也有助於把這種由寫死邏輯指揮的 Agent 與訓練好的 Agent 進行比較。

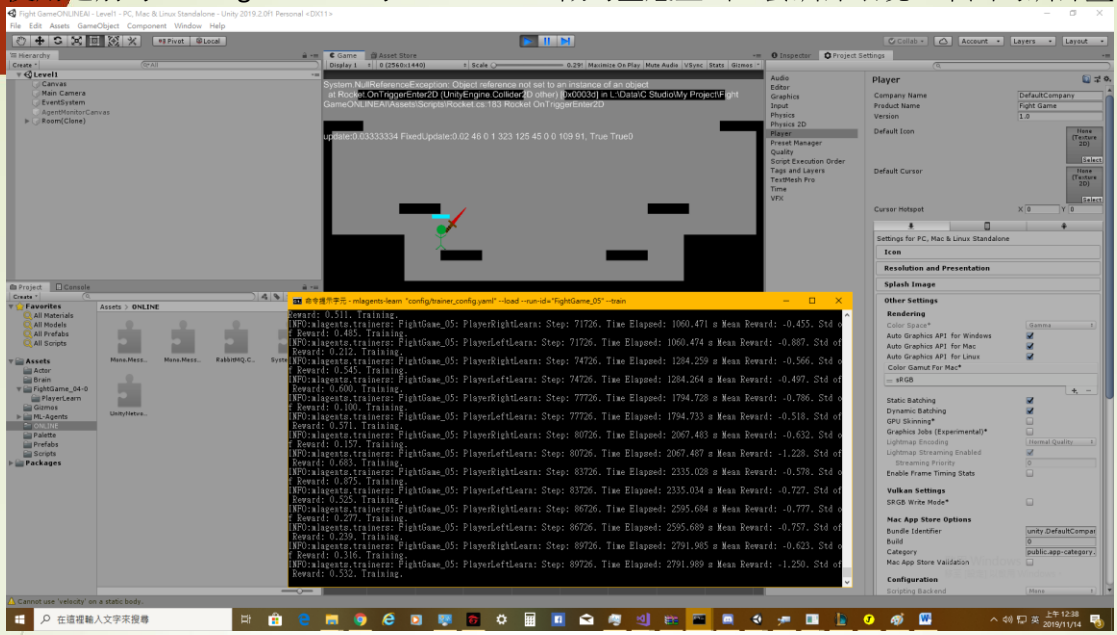


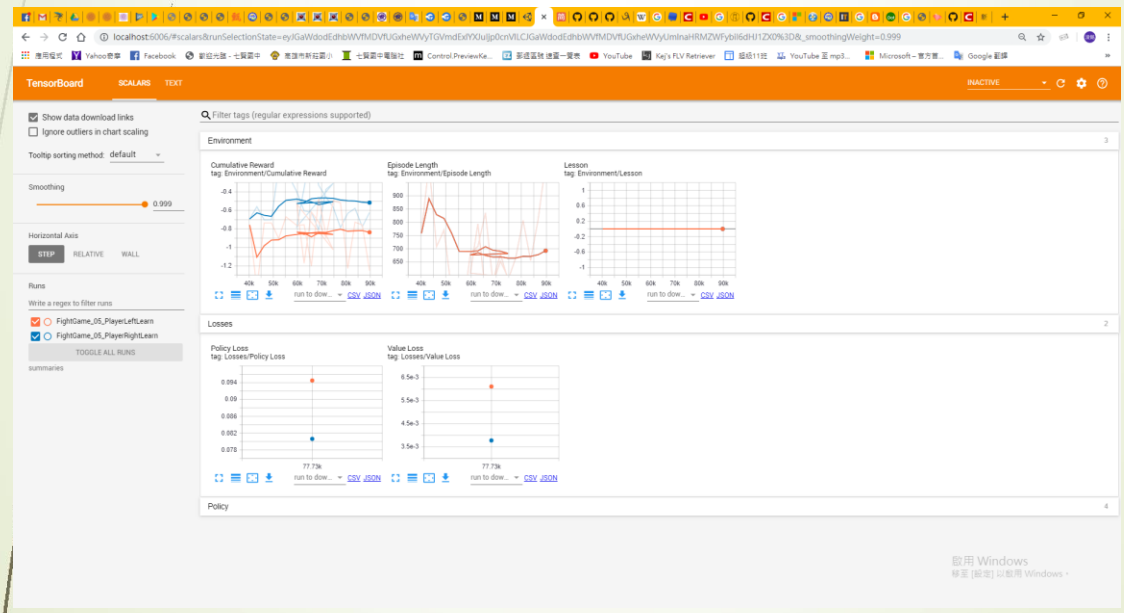
其中 Communicator 又有分為兩種，一種是以 Socket 作為溝通管道的 SocketCommunicator 另一種為以 GRPC 作為溝通管道的 RpcCommunicator。其中 ML-Agent 主要是使用 RpcCommunicator 與 Python API 進行溝通，而使用者也可以修改 C#與 Python 的源碼藉此讓他們用 SocketCommunicator 溝通。

進入決定參數階段時，Academy 會呼叫 AgentSendState 事件用來做取得與整理每個 Agent 所獲取的環境數據。之後呼叫 BrainDecideAction 事件用來讓所有 Brain 將的資料整合參數傳送至 Batcher 類別進行整合並透過 Communicator 類別將所有資料寄給 Python API，而 Python API 再用 tensorflow 進行深度學習，並將資料回傳至 Unity。

設計：

使用之前的 StickFightOnline 的 GameServer 做為基底些出一套訓練環境。下圖為訓練畫面。





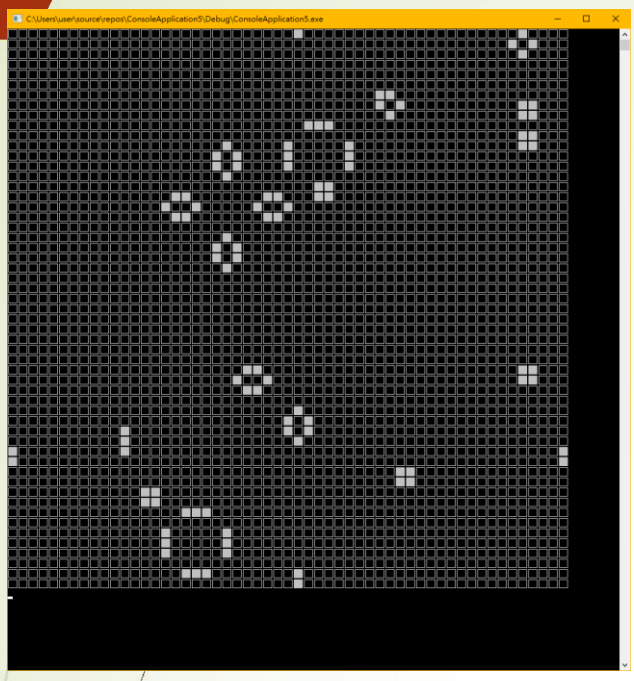
其中環境參數總共為 1822 個，輸出的動作採用連續動作參數共為 21 個。並且神經網路中的隱藏層有 6 層，其中每個隱藏層共有 2048 個節點。

而為了使測試方便，我新增了一個新的 Brain：NetBrain 作為手機客戶端的控制器。

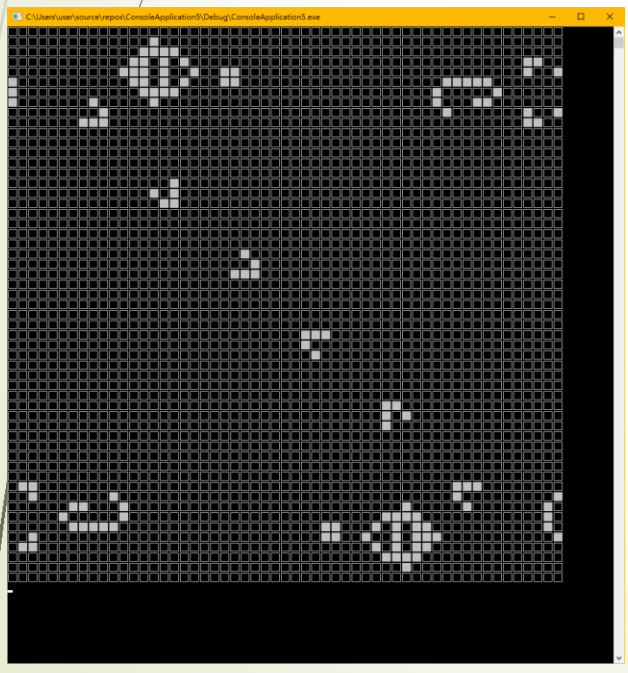
目前該專案依然在訓練與修改中。

C++康威生命遊戲

GitHub: https://github.com/Jimmy01240397/LifeGame



圖一為在執行一陣子後變成有規律的穩定狀態



圖二為以自訂初始結構的方式製造兩組方向相反的「高斯帕機槍」，而滑翔機在互撞之後便會消失。

康威生命遊戲是英國數學家約翰·何頓·康威在 1970 年發明的細胞自動機。它包括一個二維矩形世界，這個世界中的每個方格居住著一個活著的或死了的細胞。一個細胞在下一個時刻生死取決於相鄰八個方格中活著的或死了的細胞的數量。如果相鄰方格活著的細胞數量過多，這個細胞會因為資源匱乏而在下一個時刻死去；相反，如果周圍活細胞過少，這個細胞會因太孤單而死去。

其規則為：

- 每個細胞有兩種狀態 - 存活或死亡，每個細胞與以自身為中心的周圍八格細胞產生互動（如圖，黑色為存活，白色為死亡）
- 當前細胞為存活狀態時，當周圍的存活細胞低於 2 個時（不包含 2 個），該細胞變成死亡狀態。（模擬生命數量稀少）
- 當前細胞為存活狀態時，當周圍有 2 個或 3 個存活細胞時，該細胞保持原樣。
- 當前細胞為存活狀態時，當周圍有超過 3 個存活細胞時，該細胞變成死亡狀態。（模擬生命數量過多）
- 當前細胞為死亡狀態時，當周圍有 3 個存活細胞時，該細胞變成存活狀態。（模擬繁殖）

其基礎規則的實現方法非常簡單，只需要製作一個字串陣列作為 2 維世界的地圖，

```
int main()
{
    string a[100];
    //unsigned int x = time(NULL);
    srand(time(NULL));
    for (int i = 0; i < 55; i++)
    {
        a[i] = "";
        for (int ii = 0; ii < 55; ii++)
        {
            //srand(rand() % rand());
            if (Rand == 0)
            {
                a[i] += "□";
            }
            else
            {
                int xx = rand() % 2;
                a[i] += (xx == 0) ? "□" : "■";
            }
            //x = rand();
        }
    }
```

並且製作一個用來表示座標的結構(struct)儲存該次循環依造規則所需要改變的座標，並在程式最後更新陣列即可

```
12
13  #define upx -17
14  #define upy 1
15  #define dox 16
16  #define doy -1
17  #define Rand 0
18  using namespace std;
19
20  struct Pos
21  {
22      int x;
23      int y;
24  };
25
26  void Set(string data[], int x, int y, string value)
27  {
28      data[y][x * 2] = value[0];
29      data[y][x * 2 + 1] = value[1];
30  }
31
32  bool Check(string data[], int x, int y, string value)
33  {
34      return data[y][x * 2] == value[0] && data[y][x * 2 + 1] == value[1];
35  }
36
```

```
    list<Pos> changeLife;
    list<Pos> changeDeth;
    for (int i = 0; i < 55; i++){
        for (int ii = 0; ii < 55; ii++){
            int q = 0;
            for (int iii = -1; iii < 2; iii++){
                for (int iiii = -1; iiii < 2; iiii++){
                    if (iii == 0 && iiii == 0){
                        continue;
                    }
                    /*if (i + iii < 0 || i + iii >= 55)
                    {
                        continue;
                    }
                    if (ii + iiii < 0 || ii + iiii >= 55)
                    {
                        continue;
                    }*/
                    if (Check(a, (55 + ii + iiii) % 55, (55 + i + iii) % 55, "■")){
                        q++;
                    }
                }
            }
            if (Check(a, (55 + ii) % 55, (55 + i) % 55, "■")){
                if (q < 2 || q > 3){
                    Pos aa;
                    aa.x = ii;
                    aa.y = i;
                    changeDeth.push_back(aa);
                }
            }
            else{
                if (q == 3){
                    Pos aa;
                    aa.x = ii;
                    aa.y = i;
                    changeLife.push_back(aa);
                }
            }
        }
    }
    for (list<Pos>::iterator i = changeDeth.begin(); i != changeDeth.end(); ++i){
        Set(a, (*i).x, (*i).y, "□");
    }

    for (list<Pos>::iterator i = changeLife.begin(); i != changeLife.end(); ++i){
        Set(a, (*i).x, (*i).y, "■");
    }
    //Sleep(1);
}
```