中華大學

資訊工程學系

Android 手機應用
程式設計專題報告

指導老師:陳建宏 老師

專題組員:溫庭榮、林志嘉、王祖祥、林姿妤

專題編號: PRJ2012-CSIE-10119

執行時間:民國 101 年 3 月~102 年 6 月

摘要

本專題其目的主要是透過開放式平台 Android 作業系統在手機上開發遊戲程式，並透過藍芽傳輸介面來達到相互連線傳遞資料的可能性，再搭配手機所擁有的感應器來達到「不需碰到螢幕即可遊戲」的多種功能，讓此遊戲演變成「不用碰就能玩」的多樣化遊戲，使使用者遊玩的時候多增添了許多趣味性。

起初在「Game Maker」平台上面做了「小精靈」的遊戲，由於「小精靈」這遊戲的主角「小精靈」很可愛，且遊戲過關方式單純、簡單、好上手，原想說把這遊戲概念移植到 Android 平台上面來開發，卻發現「小精靈」的遊戲性略微不足，且能運用到的感測裝置也不多，後來就因「富甲天下系列」的遊戲性影響，決定以「大富翁」為遊戲主題來開發，並沿用其獨特的遊戲豐富性、趣味性，再運用現在的行動裝置及感測器的多樣化，來更豐富這「大富翁」。

目錄

壹.簡介

1.1 研究動機

現今的生活當中已經不再像是過往使用黑金鋼聯繫的時代,手機的使用漸漸的在我們周遭發生不一樣的變化。尤其是智慧型手機的功能上,不光是顛覆過往,更是成為日常不可或缺的一項電子產品,並且它也已經取代了傳統成為目前的主流。在目前智慧型手機市場的高佔有率的現況下,也漸漸改變人們使用手機的習慣,從原本單純聯繫的電話轉變成藉由手機與手機之間的互動,進而發展出另一種人與人之間的互動模式,成了生活裡習慣的一部分。

於近年來,隨著Android平台的推出,Android SDK套件提供完整的API,讓程式開發者可以在這塊領域上盡情發揮。各家手機廠商投入Android手機軟硬體的開發,如此普遍的情況以及趨勢下,Android APP的開發也日漸茁壯。Android原始碼在網路上以及許多書籍中都達到高度的開放,方便撰寫、研究開發各種應用程式。

目前大多數的 Android 遊戲都為單機遊戲的形式為主,手機上的遊戲與其他使用者互動的遊戲仍然不多。這也激起我們想開發互動式遊戲的想法 ,讓遊戲不在只是獨樂樂,也可以和親朋好友一起眾樂樂。

1.2 研究目的

此遊戲可讓使用者等車、搭車、無聊之時可隨手拿起手機來遊玩,其獨特的擲骰子、蓋房子、獨特事件格,皆不需「觸控」到螢幕即可完成此動作,並可透過藍芽傳輸介面來達成多人連線遊戲,使用者可以在遊玩的過程中因獨特的玩法而得到無限的樂趣;在未來的科技裡,也許這種「no-touch」的感應方式可以超越聲控感應的穩定度,並帶領科技業者們新的流行趨勢。

1.3 專題分工與時間配表

專題分工：

工作	人員
報告及 Android 程式設計開發	溫庭榮
Android 程式設計開發 程式測試與偵錯	王祖祥
蒐集資料	林志嘉
報告製作	林姿妤

時間分配表：

月份 工作內容	3月	4月	5月	6月	7月	8月	9月	10月	11月	12月	1月	2月	3月	4月	5月	6月
搜集資料	V	V	V	V	V	V	V	V	V	V						
學習相關軟體應用				V	V	V	V	V	V	V	V	V	V	V	V	V
程式開發										V	V	V	V	V	V	V
實作與整合														V	V	V
測試與偵錯															V	V
問題討論			V			V			V				V			V
影片拍攝																V
書面報告製作																V
預定進度 累計百分比	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	70%	80%	90%	100%

貳.專題背景介紹

2.1 開發環境介紹

Eclipse是著名的跨平台的自由整合適開發環境(IDE)。最初主要用來Java語言開發，但是目前亦有人透過外掛程式使其作為C++、Python、PHP等其他語言的開發工具。

Eclipse的本身只是一個框架平台，但是眾多外掛程式的支援，使得Eclipse擁有較佳的靈活性。許多軟體開發商以Eclipse為框架開發自己的IDE。

Eclipse 是開放原始碼的專案，你可以到 www.Eclipse.org 去免費下載 Eclipse 的最新版本，一般 Eclipse 提供幾個下載版本：Release，Stable Build，Integration Build 和 Nightly Build。Eclipse 本身是用 Java 語言撰寫，但下載的壓縮包中並不包含 Java 執行環境，需要用戶自己另行安裝 JRE，並且要在作業系統的環境變數中指明 JRE 中 bin 的路徑。

Eclipse 的設計思想是：一切皆外掛程式。Eclipse 核心很小，其它所有功能都以外掛程式的形式附加於 Eclipse 核心之上。Eclipse 基本核心包括：圖形 API (SWT/Jface)， Java 開發環境外掛程式（JDT），外掛程式開發環境等。

2.2 開發系統介紹

「Android」是一個以 Linux 為基礎的半開放原始碼作業系統，主要用於行動設備，由「Google」成立的 Open Handset Alliance（OHA，開放手機聯盟）持續領導與開發中。Google 透過官方網上商店平台 Google Play，提供應用程式和遊戲供用戶下載，截止至 2012 年 6 月，Google Play 商店擁有超過 60 萬個官方認證應用程式。同時用戶亦可以通過第三方網站來下載。2010 年末數據顯示，僅正式推出兩年的 Android 作業系統在市場佔有率上已經超越稱霸逾十年的諾基亞 Symbian 系統，成為全球第一大智慧型手機作業系統。

本專題是採用 Android 2.3.3 的版本，Android 2.3.3 是開發當時大眾最穩定的版本所以以此版本做開發。

以下為 2.2.1 跟 2.3.3 的比較表：

2.2/2.2.1 **(Froyo)** **基於 Linux Kernel 2.6.32**	2010 年 5 月 20 日，2.2（Froyo 冷凍優格）版本軟體開發套件發佈。主要的更新如下: ● 支援將軟體安裝至擴充功能內存 ● 整合 Adobe Flash 10.1 支援 ● 加強軟體即時編譯的速度 ● 新增軟體啟動"快速"至電話和瀏覽器 ● USB 分享器和 WiFi 熱點功能 ● 支援在瀏覽器上傳檔案 ● 更新 Market 中的批次和自動更新 ● 增加對 Microsoft Exchange 的支援（安全政策, auto-discovery, GAL look-up） ● 整合 Chrome 的 V8 JavaScript 引擎到瀏覽器 ● 加強快速搜尋小工具 ● 更多軟體能透過 Market 更新，類似 2.0/2.1 中的 Map 更新 ● 速度和效能最佳化 註：部分標稱為 2.2 的韌體仍然在使用 2.6.29 的核心。這種韌體可以實作大部分 2.2 的功能（比如 flash），但效能上與 2.6.32 有一定差距。

2.3 **(Gingerbread)** **基於 Linux Kernel 2.6.35**	2010 年 12 月 7 日，2.3（Gingerbread 薑餅）版本軟體開發套件發佈。主要的更新如下: ● 修補 UI ● 支援更大的螢幕尺寸和解像度（WXGA 及更高） ● 系統級複製貼上 ● 重新設計的多點觸控式螢幕幕鍵盤 ● 原生支援多個鏡頭（用於視訊通話等）和更多感測器（陀螺儀、氣壓計等） ● 電話簿整合 Internet Call 功能 ● 支援近場通訊（NFC） ● 強化電源、應用程式管理功能 ● 新增下載管理員 ● 最佳化遊戲開發支援 ● 多媒體音效強化 ● 從 YAFFS 轉換到 ext4 檔案系統 ● 開放了螢幕截圖功能 ● 對黑色及白色的還原更加真實 ● Google Talk 視訊功能

Android 2.3.3 更新包於 2011 年 2 月 9 日正式發布，僅針對上一個版本進行了 API 改進，其他無變化。

2.3 大富翁遊戲內容介紹

此遊戲內容主要以「大富翁」為主，大富翁是一種多人策略圖版遊戲。參賽者分得遊戲金錢，憑運氣（擲骰子）及交易策略，買地、建樓以賺取租金。英文原名 monopoly 意為「壟斷」，因為最後只得一個勝利者，其餘均破產收場。遊戲的設計當初旨在暴露放任資本主義的弊端，但是推出之後卻受到大眾歡迎。

美國 Parker Brothers 於 1935 年 11 月 5 日發售大富翁。英國版於 1936 年由 Waddington Games 推出。大蕭條時的大富翁在美國漸受歡迎。

大多數人認為大富翁由 Charles Darrow 發明，為的是向國會提出資本主義的弊端，財主壟斷市場，小商戶鬥不過大企業，卻想不到這遊戲會受大眾歡迎。早於 1904 年，Lizie J. Magie 已經為一種類似的遊戲大地主遊戲（The Landlord's Game）申請專利。此遊戲則發展成為不同的版本。

而現今手機遊戲上的大富翁預設多幅地圖，以擲骰點數前進，並有多種道具、卡片使用，另外觸發一些「特別事件」。主要通過購買房產，收取對方的過路費、租金，來導致對手破產。

參.專題設計方式

3.1 重要元件的應用

在元件的應用上，透過以下應用來完成遊戲每項功能的使用。

(1)Activity 的應用：

一個 Activity 就是用來處理使用者直覺的操作介面的元件，不同的 Activity 可以擁有不同的操作介面，且 Activity 跟 Activity 之間可以互相溝通。

(2)Intent 的應用：

Intent 是元件之間通訊溝通的橋樑，透過 Intent 物件將訊息傳遞，Intent 主要是執行於相同或不同的 Activity、Service 和 Broadcast Receiver 元件之間，其作用的機制也不相同。

(3)Canvas 的應用：

Canvas 都是在 graphics 類別底下，都是屬於跟圖片處理相關的類別，在遊戲製作中裡扮演很重要的腳色，Bitmap 可以將圖片的像素儲存，並利用 Canvas 來管理儲存在 Bitmap 裡的圖片，遊戲場景都是透過 Canvas 來著色上去的。

(4)Gesture 的應用：

在 Android 的系統當中，在手勢的識別上，是透過 GestureDetector.OnGestureListener 來實現的，所有的 View 都可以透過 setOnTouchListener()、setOnKeyListener()等方法來對某一件事情監聽。

在遊戲的操作上，我們一共有三種手勢由左往右滑、由右往左滑以及輕觸螢幕，由左往右和由右往左是分別抓取起點跟終點的 X 座標去計算，利用此手勢去切換選擇人物與地圖.....等等，利用輕觸螢幕去執行遊戲中的動作。

(5) SurfaceView 類：

SurfaceView 類繼承自 VIEW，但它與 VIEW 不同，VIEW 是在 UI 的主線程中更新畫面的，而 SurfaceView 是在一個新的線程更新畫面，VIEW 的特性決定了其不適合做動畫，因為如果更新畫面時間過長，那麼主 UI 線程就會被正在畫的函數中斷住，所以在 Android 中通常用 SurfaceView 顯示動畫效果，相當適合許多 Game 的製作

(6)Orientation 感應器：

Orientation 感應器可以讀取以手機為座標中心的三軸轉動角度，也就是 Direction、Pitch 和 Roll，Direction 是以手機正面向上法線(Z 軸)為軸旋轉，Pitch 是以手機中間橫向(Y 軸)為軸旋轉，Roll 是以手機中間直向(Y 軸)為軸旋轉，我們在專題只用到 Pitch 和 Roll 的量值，在程式中 Pitch 的值代表 forward 的前進向量量值，Roll 代表的 turn 是轉彎向量量值。

SensorEventListener 說明：

資料	意義	資料範圍
event.value[0]	Direction	0~359
event.value[1]	Pitch	-180~180
event.value[2]	Roll	-90~90

我們從 X、Y、Z 軸的變化量特別定義了搖晃到一定程度便會執行擲骰子的動作，其他也有判斷往前傾斜、往後傾斜、往左傾斜、往右傾斜來執行遊戲裡的動作。

(7)bluetooth 的應用：

Android 的藍芽物件定義在 Android.bluetooth 底下，BluetoothAdapter 類別主要是控制手機藍芽裝置功能和取得手機系統上的藍芽裝置。

BluetoothDevice 類別是表示遠端的藍芽裝置；BluetoothSocket 類別表示 Cluent 端傳輸介面；BluetoothServerSocket 表示 Server 端連接介面，這種運作方式和網路 TCP Socket 非常相似。

1.要使用藍芽設備，必頇在 AndroidManifest.xml 註冊使用權限

<uses-permissionandroid:name="android.permission.BLUETOOTH_ADMIN" />

<uses-permission android:name="android.permission.BLUETOOTH" />

2.利用 BluetoothAdapter.getDefaultAdapter()方法來取得本地端的藍芽裝置，再來就是要啟動藍芽裝置，可以利用 isEnabled()去檢查藍芽是否有沒有啟動，沒有的話可以利用動作 Intent:ACTION_REQUEST_ENABLE 並且搭配 startActivityForResult()來啟動本地的藍芽裝置。

3.可以呼叫 getBondedDevice()方法來找到以前曾經配對過的裝置，它會回覆一組 BluetoothDevice，從 BluetoothDevice 可以取得一個 MAC address 來建立一個連接，並且可以利用 startDiscovery 來搜尋附近的藍芽裝置。

4.實作 Server 端：呼叫 listenUsingRfcommWithServiceRecord()，取得 BluetoothServerSocket 實體物件，並且呼叫 accept()方法監聽遠端藍芽裝置的連線請求，當接受到遠端藍芽的連線請求，accept()方法會回傳一個實體物件 BluetoothServerSocket，利用該物件建立連線串流，當連線已經建立就呼叫 close()關閉端口。

5.實作 Client 端：從搜尋或已經配對的藍芽裝置的 MAC，呼叫 BluetoothAdapter 物件實體的 getRemoteDevice(String MAC)方法，傳回指定連接遠端的 BluetoothDevice(遠端的 Server)，接著呼叫遠端 BluetoothDevice 物件的 listenUsingRfcommWithServiceRecord()方法取得實體物件，再利用 BluetoothSocket 物件實體的 connect()方法與遠端 Server 建立連線，建立連線之後，

就可以開始藉由串流傳送資料。

6.當藍芽連線已經建立，就可以用 IuputStream 和 OuputStream 來處理透過 socket 的資料，讀寫資料給串流是透過 read(byte [])與 write(byte [])。

(8)Handler 的應用：

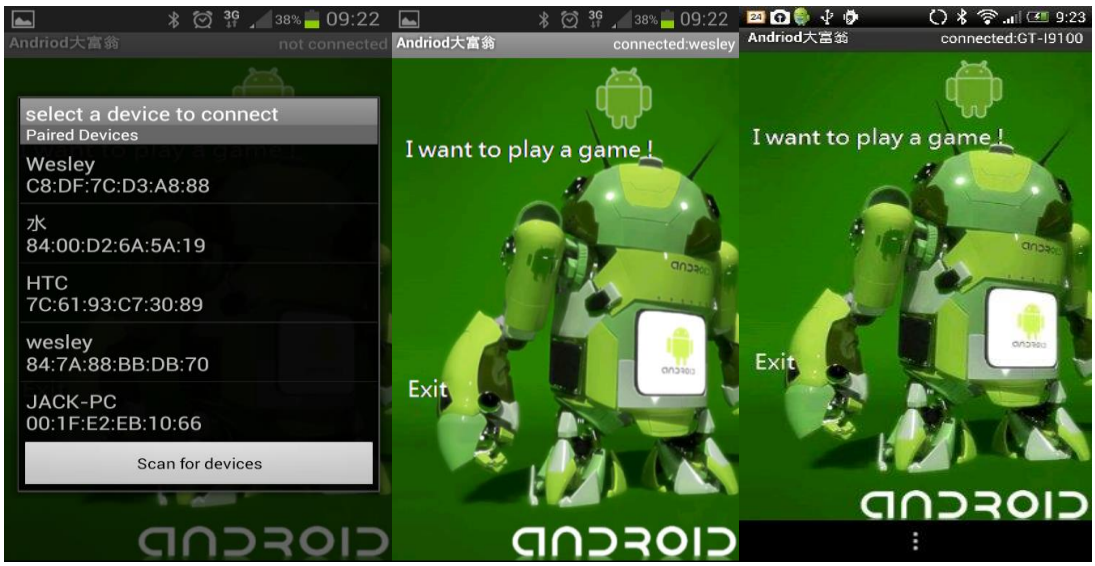
Handler 類別主要用於應用程式的主執行緒同使用者自己新建的執行緒進行通訊，應用程式在主執行緒中維護一個訊息佇列。Handler 機制使得執行緒之間的通訊透過 Message 和 Runnable 物件來傳遞和處理。Handler 在主要的用法就是安排 Message 和 Runnable 物件使其在未來的某個時刻被處理或執行。

我們利用藍芽傳遞的訊息，就是透過 Handler 利用 Message 將這些訊息傳遞出去再做處理。

3.2 程式運作流程圖

（1）在一開始讓藍芽開啟後，手機裝置偵測於本地的藍芽裝置後，按下玩家要連線的裝置。

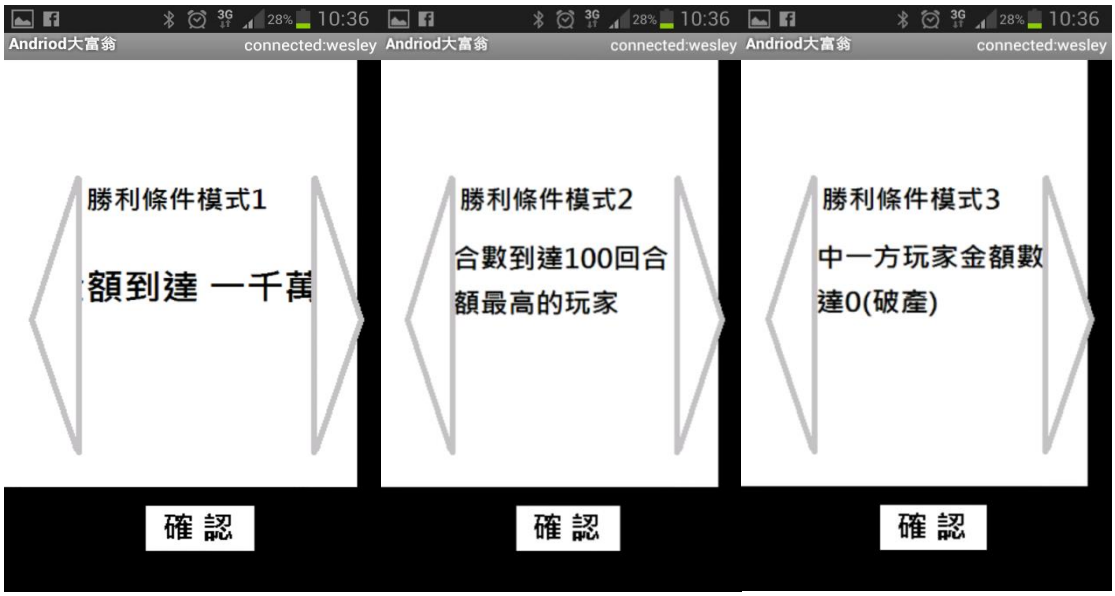
（2）按下確定連線之後就畫面右上角就會顯示 connecting，一但連線成功便會顯示 connected to 連接上的裝置名稱，由被連接的玩家為 player1，連接的玩家為 player2。



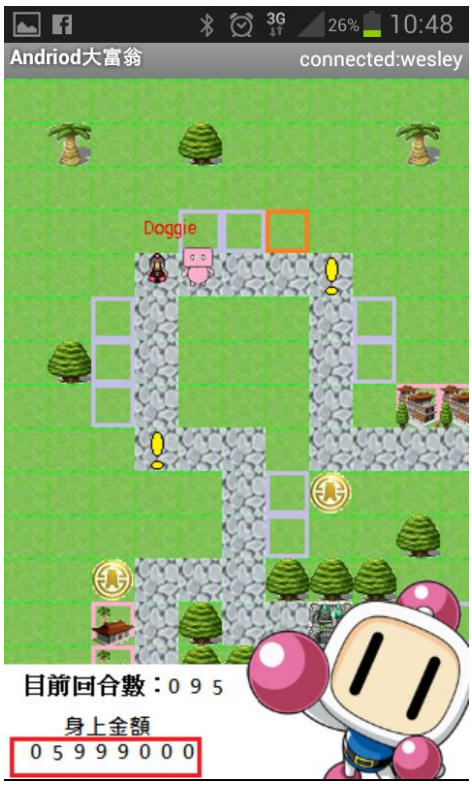
（3）點選開始遊戲後，player1 進入選擇角色畫面，一共有 4 位角色可以選擇，接著由 player2 進入選擇角色畫面。

（4）接著 player1 進入選擇地圖畫面，一共有 2 張地圖可以選擇，而這時 player2 是進入 waiting 畫面等待 player1 選擇完地圖和模式。

（**5**）player1 選擇完地圖後，接著進入選擇模式畫面，一共有三個模式，分別是：1.金錢數率先達到一千萬、2.100 回合數金額最高、3.一方玩家破產後由另一方獲勝。



（**6**）player1 選擇完地圖以及模式後便後開始遊戲，此時 player2 讀取到地圖以及模式資訊後也從 waiting 畫面進入開始遊戲。



（7）進入大富翁遊戲畫面後，由 player1 開始第一回合，可以選擇使用道具干擾對手或是擲骰子來決定玩家停留的格子，而所停留的格子種類不同有相對應的事件發生。

（8）格子種類以及道具：

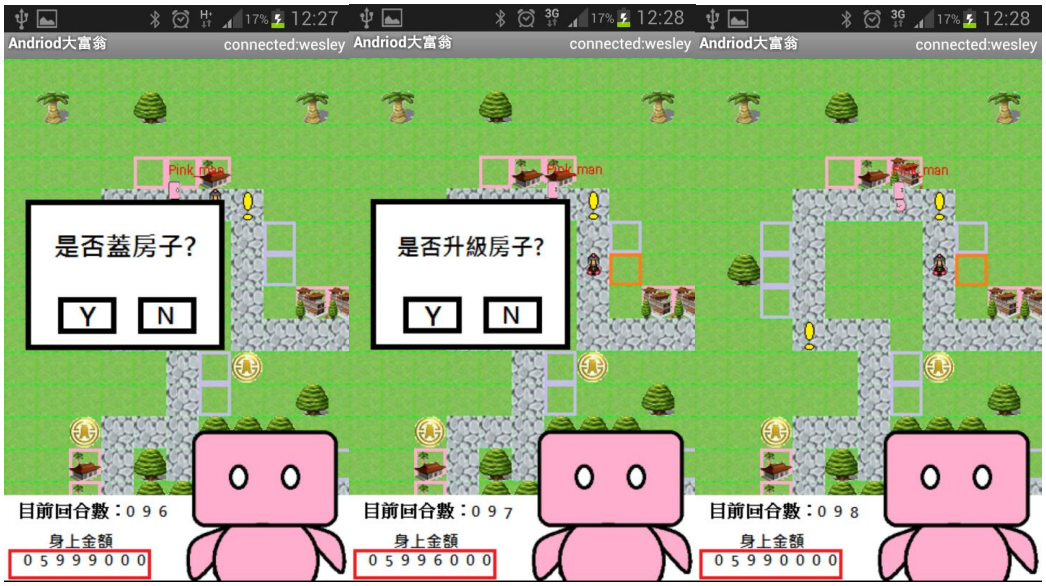
下列介紹格子種類：

可購買的空地：

當玩家走到空地時，可以自行決定是否購買空地的所有權。持有土地的所有權才可以對其他玩家收取過路費。

房屋：

一但有了該土地的所有權，當第二次以後再停留該土地上時可以選擇是否要購置房子，甚至已經建構過的房子可以在升級。



機會/命運：

走到此格子時，會隨機從眾多事件中來擇一發生會產生不可預期的結果。

下列介紹會發生的事件

1. **暫停事件:**一旦觸發此事件將會有 1~3 回合之間無法做出任何動作

2. **監獄事件:**一旦觸發此事件將會被關進監獄 1~3 回合無法做出任何動作，結束後從監獄出發

3. **旅行事件:**一旦觸發此事件將會移動到特定格子

4. **增加金錢事件:** 一旦觸發此事件會獲得金錢

5. **減少金錢事件:** 一旦觸發此事件會減少金錢

過路費:

持有此土地所有權的玩家將會收取過路費，會依照土地的狀況來收取過路費。

銀行:

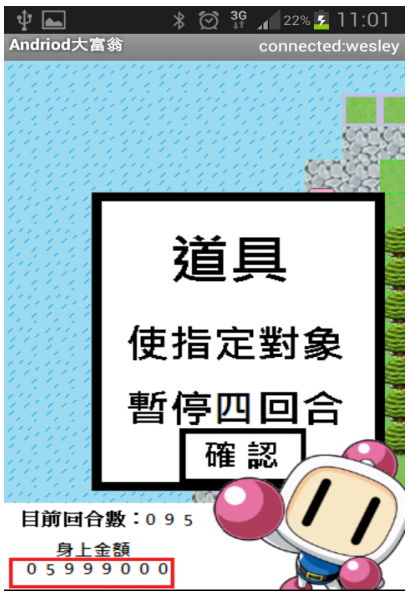
經過銀行可以進行存款提款的動作,存在銀行的存款可以藉由一定比例的利息增加。

空地:

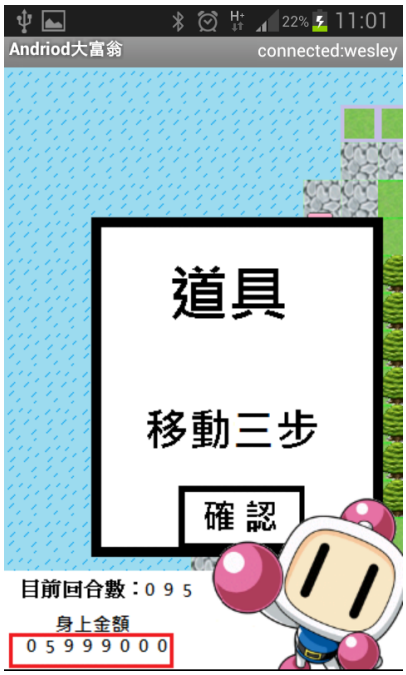
這類型的土地不能購買，也不會有任何狀態。

下列介紹可使用的道具:

道具一:可使用道具使對方暫停 1~3 個回合數。



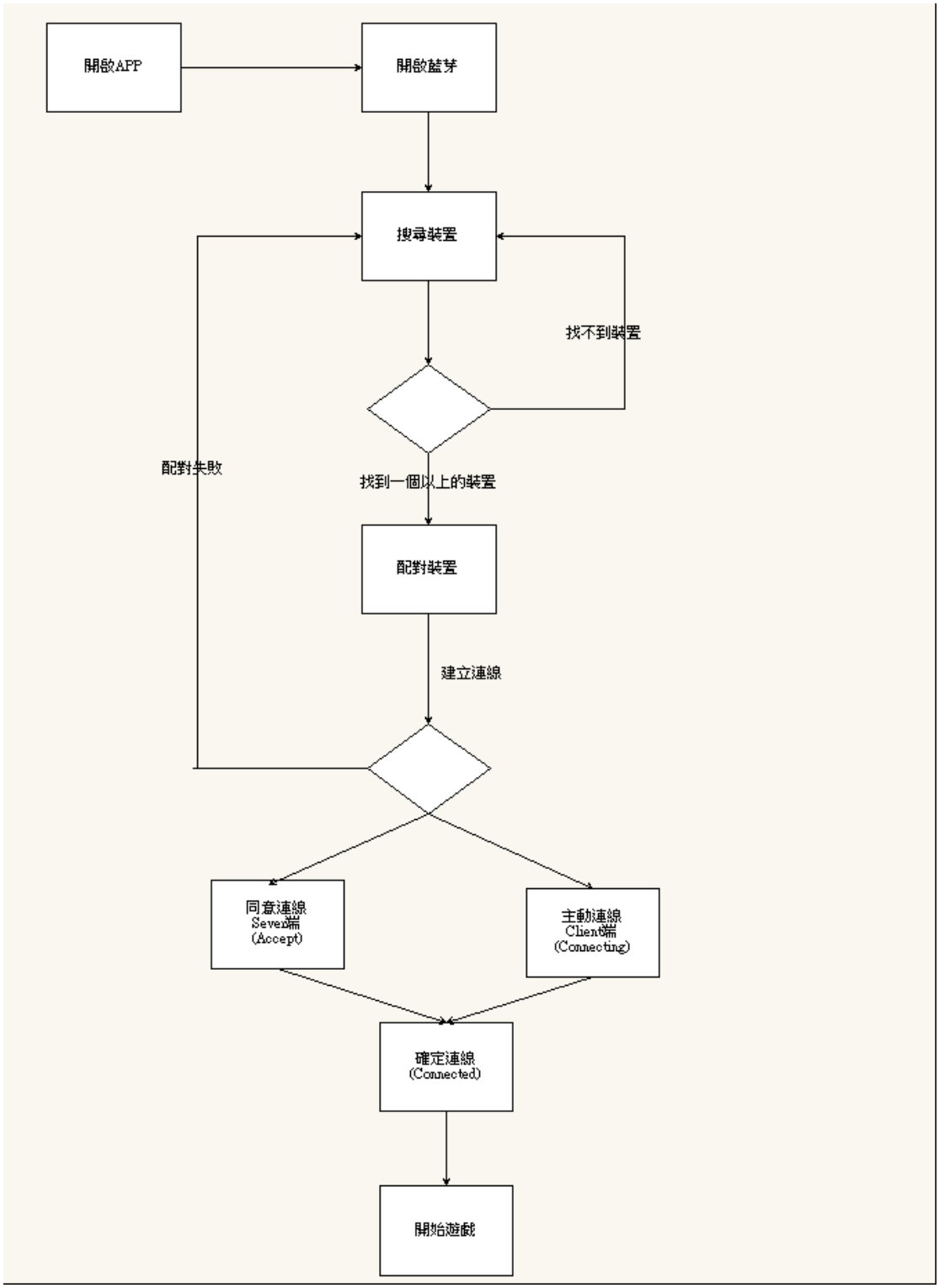
道具二: 可依照自己需求指定要移動的步數(1~6 步數)。



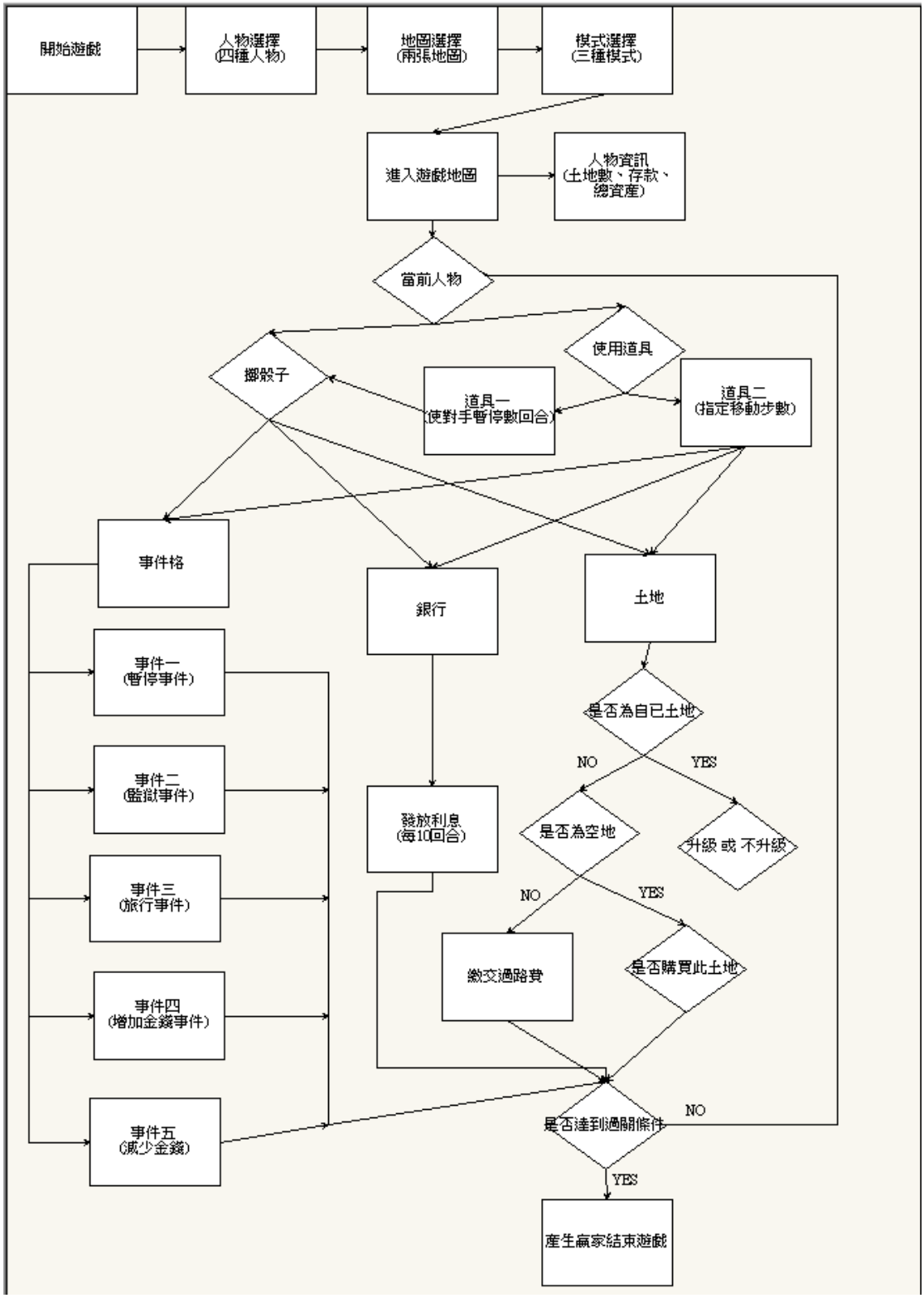
(9)以上事件擇一發生或使用道具後結束回合，將會檢查是否有達到模式選擇下結束遊戲的條件，若是沒有將會輪到 player2 玩家。

(10) Player2 玩家也可以選擇使用道具干擾對手或是擲骰子來重複以上動作，若是沒有達到結束遊戲的條件，回合將會以此模式輪流直到達到結束遊戲的條件產生贏家為止。

藍芽連線流程圖：


開啟APP
開啟藍芽
搜尋裝置
找不到裝置
配對失敗
找到一個以上的裝置
配對裝置
建立連線
同意連線
Sever端
(Accept)
主動連線
Client端
(Connecting)
確定連線
(Connected)
開始遊戲

遊戲運作流程圖：


開始遊戲
人物選擇
(四種人物)
地圖選擇
(兩張地圖)
模式選擇
(三種模式)
進入遊戲地圖
人物資訊
(土地數、存款、
總資產)
當前人物
使用道具
擲骰子
道具一
(使對手暫停數回合)
道具二
(指定移動步數)
事件格
銀行
土地
事件一
(暫停事件)
事件二
(監獄事件)
事件三
(旅行事件)
事件四
(增加金錢事件)
事件五
(減少金錢)
是否為自己土地
NO
YES
發放利息
(每10回合)
是否為空地
升級 或 不升級
NO
YES
繳交過路費
是否購買此土地
是否達到過關條件
NO
YES
產生贏家結束遊戲

3.3 主要功能演算法

3.3.1 藍芽：

1.一進入 APP 程序便會開啟藍芽功能，有藍芽權限的要求，並確定開啟藍芽應用。

2.按下"是"之後，會開啟藍芽功能，並且等待連線。

3.如果是第一次跟另外藍芽裝置做連線，可以選擇讓本地端被搜尋，讓其他裝置來搜尋取得 MAC address，以便做連線。

4.等待連線當中，本身會先處於被動的狀態下會先執行 AcceptThread 的執行緒先假設每臺裝置都有可能想要跟它做藍芽連線。若是主動按下 menu 鍵可以主動與其他裝置作連接，主動連線的一端會先關閉 AcceptThread 的執行緒接著執行 ConnectThread 的執行緒去嘗試作連接，畫面上會顯示 Connecting。

5.此時按下連接的那端我們將設定他為 Client 端，而被動同意連線的一端我們設定為 Sever 端，以方便我們區別玩家的連線狀態來傳遞資訊。

```
socket = mmServerSocket.accept(); //同意連線
player_state = player_server ; //同意連線的一方做為Sever端

mmSocket.connect(); //請求連線
player_state = player_client ; //請求連線的一方做為Client端
```

6.一但成功連線之後就會同時關閉 AcceptThread 和 ConnectThread 的執行緒，此時 Sever 端與 Client 端同時執行 ConnectedThread 的執行緒，同時在我們的畫面上便會顯示 Connected to 連線上的裝置，我們這時就可以做資料上的傳遞了。

7.兩方在這個時候都拿到這次藍芽溝通專用的 BluetootheSocket，Socket 相當於一個連接彼此的接口，彼此利用 BluetoothSocket 互相做資料傳輸，資料傳輸利用 InputStream 和 OutputStream 並透過旗下的 write(byte [])和 read(byte [])在做 2 隻藍芽裝置的溝通。

8.write(byte [])和 read(byte [])都只能讀取 byte array，我們所需要傳遞的資料都是以 integer 資料型態為主，因此我們需要將要傳遞的 integer 資料型態轉變成 string 型態再用 getbyte()的方法放進 byte array 寫入到 write(byte [])，再由 read(byte [])讀取到傳遞到的 byte array 轉回成 string,再使用 Integer.parseInt()變成我們所需要 integer 資料型態。

9.接收到資料後我們先區分 Sever 端和 Client 端在不同的遊戲狀態下所接收到不同的值做出相應的動作。

程式碼如下：

```
int getMessage = Integer.parseInt(readMessage); //將string換成int
status = getMessage / 1000000000;
if(player_state == player_server) //伺服端
{
    if(gameState == change_role )
    {
        sovereignty = 1;
        role_no[sovereignty] = getMessage; //接收客戶端的人物資料
        sovereignty = 0;
        server_tomap = 1; //接收到客戶端的人物資料後伺服端進入地圖選擇
    }
    if(gameState == gaming )
    {
        if(getMessage <= 1000000006 && getMessage > 1000000000) rnd[sovereignty] = getMessage % 1000000000; //擲骰子
        if(getMessage < 2000000000 && getMessage > 1000000006) cash[sovereignty] = ((getMessage - 1000000000) /10); //金額
        if(getMessage >= 2000000000) deposit[sovereignty] = ((getMessage - 2000000000) /10); //存款金額
        if(getMessage == 1200) nomoney = true; //購買土地金額不足
        if(getMessage == 1020) no = true; //不蓋房子
        else if(getMessage == 1010) yes = true; //買土地
        else if(getMessage == 1030) yes = true; //蓋房子一
        else if(getMessage == 1040) yes = true; //蓋房子二
        if(getMessage == 1050) check_bank[sovereignty] = 0; //進入銀行旗標初始化
        if(getMessage == 1060) rnd_event[sovereignty] = 1; //暫停事件
        else if(getMessage == 1070) rnd_event[sovereignty] = 2;  //監獄事件
        else if(getMessage == 1080) rnd_event[sovereignty] = 3;  //旅行事件
        else if(getMessage == 1090) rnd_event[sovereignty] = 4;  //增加金錢事件
        else if(getMessage == 1100) rnd_event[sovereignty] = 5;  //減少金錢事件
        if(getMessage >= 1061 && getMessage <= 1063) rnd_stop[1] = getMessage % 1060; //接收客戶端人物暫停回合
        if(getMessage >= 2010 && getMessage <= 2014)
        {
            use_props_No1 = true; //客戶端使用道具一
            props_01_stop[0] = getMessage % 2010; //暫停的回合數
        }
        if(getMessage >= 2020 && getMessage <= 2026)
        {
            use_props_No2 = true; //伺服端使用道具二
            props_02_walk = getMessage % 2020; //行走的步數
        }

    }
}
```

```
else if(player_state == player_client) //客戶端
{
    if(gameState == menu )
    {
        role_no[sovereignty] = getMessage; //接收伺服端的人物資料
        sovereignty = 1;
        client_tochangerole = 1; //收到伺服端人物資料後進入人物選擇
    }
    if(gameState == waiting )
    {
        if(getMessage <= 3) map_no = getMessage; //接收地圖資訊
        if(getMessage > 3)
        {
            mode_flag = getMessage; //接收模式資訊
            client_togaming = 1; //收到地圖與模式開始遊戲
        }
    }
    if(gameState == gaming )
    {
        if(getMessage <= 1000000006 && getMessage > 1000000000) rnd[sovereignty] = getMessage % 1000000000; //擲骰子
        if(getMessage < 2000000000 && getMessage > 1000000006) cash[sovereignty] = ((getMessage - 1000000000) /10); //金額
        if(getMessage >= 2000000000) deposit[sovereignty] = ((getMessage - 2000000000) /10); //存款金額
        if(getMessage == 1200) nomoney = true; //購買土地金額不足
        if(getMessage == 1020) no = true; //不蓋房子
        else if(getMessage == 1010) yes = true; //買土地
        else if(getMessage == 1030) yes = true; //蓋房子一
        else if(getMessage == 1040) yes = true; //蓋房子二
        if(getMessage == 1050) check_bank[sovereignty] = 0; //進入銀行旗標初始化
        if(getMessage == 1060) rnd_event[sovereignty] = 1; //暫停事件
        else if(getMessage == 1070) rnd_event[sovereignty] = 2;  //監獄事件
        else if(getMessage == 1080) rnd_event[sovereignty] = 3;  //旅行事件
        else if(getMessage == 1090) rnd_event[sovereignty] = 4;  //增加金錢事件
        else if(getMessage == 1100) rnd_event[sovereignty] = 5;  //減少金錢事件
        if(getMessage >= 1061 && getMessage <= 1063) rnd_stop[0] = getMessage % 1060; //接收伺服端人物暫停回合
        if(getMessage >= 2010 && getMessage <= 2014)
        {
            use_props_No1 = true; //客戶端使用道具一
            props_01_stop[1] = getMessage % 2010; //暫停的回合數
        }
        if(getMessage >= 2020 && getMessage <= 2026)
        {
            use_props_No2 = true; //伺服端使用道具二
            props_02_walk = getMessage % 2020; //行走的步數
        }

    }
}
break;
```

3.3.2 畫面顯示

與 SurfaceHolder.Callback 相關聯的三個程序 ：

public void surfaceCreated(...)　　當畫面建立時使用

public void surfaceChanged(...)　　當畫面改變時使用

public void surfaceDestroyed(...)　當畫面結束時使用

場景畫面的呈現，利用 bitmap 將讀進來的顏色圖片存成 bitmap 型態，在利用 canvas 類別當中的 drawBitmap 這個方法將整個遊戲場景著色。

如何讓一個動畫產生?

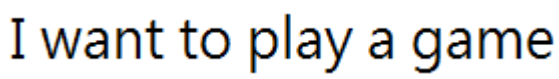
就是在短時間內不斷的更新圖片和座標，來產生一個連續的畫面，即為動畫。而我們在產生畫面時，我們寫了一個 draw()來產生畫面。

其中裡面分別產生的畫面有：

1.選單介面

a.背景圖。

b.按鈕圖(I want play a game、exit)。



2.選人物

a.背景圖。

b.左右箭頭圖(可切換角色)。

c.確認圖。

3.選地圖(只有在伺服端才會顯示)

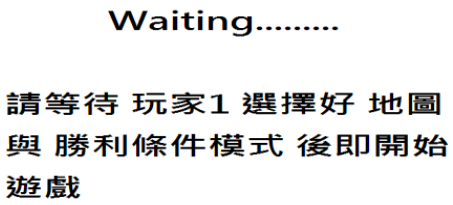
a.背景圖。

b.左右箭頭圖(可切換地圖)。

C.確認圖。

4.等待畫面(只有在客戶端才會顯示)

a.Waiting 圖。



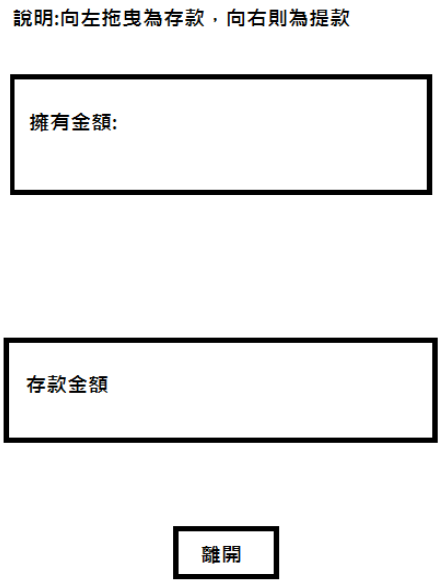
5.遊戲進行

a.背景(草地、道路、蓋房子格、房子)。

b.角色。

c.視窗(事件、確認蓋房子、跳回選單介面)。

d.銀行。



地圖顯示方法：

利用二維陣列將地圖存取，

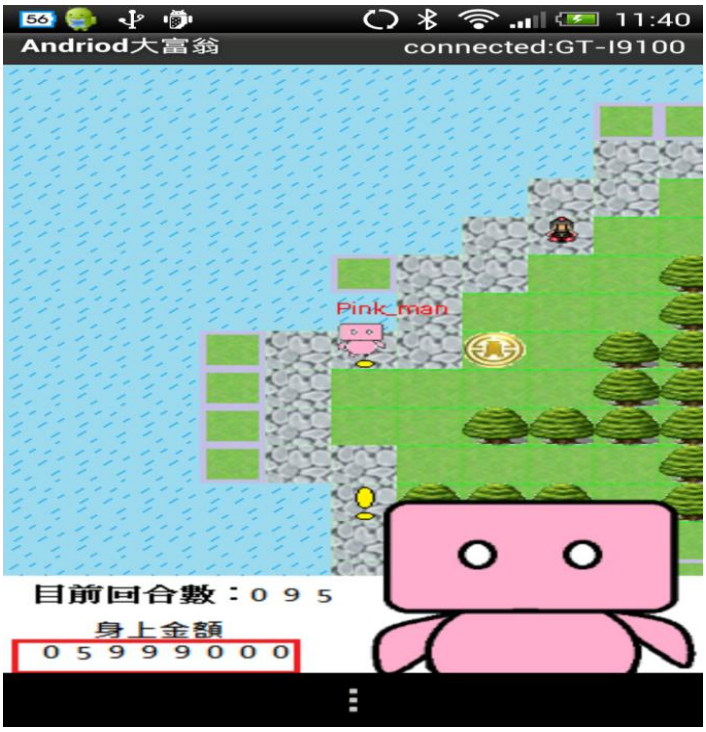
再利用 for 迴圈在螢幕上繪製出地圖。

地圖為必須先繪製的，

然後再覆蓋人物圖、房子圖、視窗...等圖片

```
for(int i=0+overmap_x[sovereignty]; i<16+overmap_x[sovereignty]; i++) {//繪出地圖
    for(int j=0+overmap[sovereignty]; j<11+overmap[sovereignty]; j++) {
        if(src==330)
        {
            src=0;
            dst+=30;
            if(dst==480)dst=0;
        }
        canvas.drawBitmap(bmp2, src, dst, null);//繪出草地
        if(i<20 && j<20){
            if(map[i][j]==1) canvas.drawBitmap(bmp3, src, dst, null); //道路
            if(map[i][j]==6) canvas.drawBitmap(event_Lattice, src, dst, null); //命運機會事件格
            if(map[i][j]==7) canvas.drawBitmap(bank_lattice, src, dst, null); //銀行
            if(map[i][j]==8) canvas.drawBitmap(tree, src, dst, null); //樹
            if(map[i][j]==9) canvas.drawBitmap(senlin, src, dst, null); //樹
            if(map[i][j]==10) canvas.drawBitmap(chengshi, src, dst, null); //監獄
            if(map[i][j]==11) canvas.drawBitmap(Ocean, src, dst, null); //海
            if(map[i][j]>1&&map[i][j]<6) canvas.drawBitmap(buildhouse_bg, src, dst, null);//灰色框框
            if(map2[i][j]==3)canvas.drawBitmap(buildhouse_bg_1, src, dst, null);//pink 玩家1土地
            if(map2[i][j]==13)canvas.drawBitmap(buildhouse_bg_2, src, dst, null);//橘色框框 玩家2土地
            if(map[i][j]==4) canvas.drawBitmap(house1, src, dst, null); //一階房屋
            if(map[i][j]==5) canvas.drawBitmap(house2, src, dst, null); //二階房屋
```

```
public static int[][] taiwan_map= {
{11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11},
        {11,11,11,11,11,11,11,11,11,11,2,2,2,2,11,11,11,11,11,11,11},
        {11,11,11,11,11,11,11,11,11,11,1,1,1,1,11,11,11,11,11,11,11},
        {11,11,11,11,11,11,11,11,11,1,1,0,0,1,1,2,11,11,11,11,11},
        {11,11,11,11,11,11,11,11,1,1,0,0,0,0,1,11,11,11,11,11,11},
        {11,11,11,11,11,11,2,1,1,0,0,9,0,2,1,1,2,11,11,11,11},
        {11,11,11,11,11,11,11,1,0,0,0,9,0,0,0,1,2,11,11,11,11},
        {11,11,11,2,1,6,1,7,0,9,9,0,8,8,1,11,11,11,11,11,11},
        {11,11,11,2,1,0,0,0,0,9,9,0,8,8,1,2,11,11,11,11},
        {11,11,11,2,1,0,0,9,9,9,9,8,8,8,1,2,11,11,11,11},
        {11,11,11,2,1,1,0,0,0,0,9,8,2,1,1,2,11,11,11,11},
        {11,11,11,11,11,6,9,9,9,0,9,8,8,6,11,11,11,11,11,11},
        {11,11,11,11,11,1,1,10,9,0,0,0,0,1,7,11,11,11,11,11},
        {11,11,11,11,11,1,9,9,9,9,2,0,1,1,2,11,11,11,11,11},
        {11,11,11,11,7,1,1,1,0,8,1,1,1,11,11,11,11,11,11,11},
        {11,11,11,11,11,11,11,11,1,1,0,1,11,2,11,11,11,11,11,11,11},
        {11,11,11,11,11,11,11,11,11,1,1,1,2,11,11,11,11,11,11,11,11},
    {11,11,11,11,11,11,11,11,11,2,11,11,11,11,11,11,11,11,11,11},
{11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11},
{11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11},
};
```



3.3.3 感應器的運用

取得 SensorManager 物件：透過 SensorManager 物件方能取得各種感應器的資訊，而要取得該物件必須呼叫 Context 的 getSystemService()，並指定欲取得的系統服務名稱。

```
public ShakeListener(Context context) {
    // 獲得監聽對象
    mContext = context;
    start();
}
```

實作 SensorEventListener：實作 SensorEventListener 的 onSensorChanged()，當感應器的值改變時會自動呼叫此方法，並傳入 SensorEvent 物件，透過該物件可以取得產生事件的感應器。

```
// 重力感應器感應獲得變化數據
public void onSensorChanged(SensorEvent event) {
    // 現在檢測時間
    long currentUpdateTime = System.currentTimeMillis();
    // 兩次檢測的時間間隔
    long timeInterval = currentUpdateTime - lastUpdateTime;
    // 判斷是否達到了檢測時間間隔
    if (timeInterval < UPTATE_INTERVAL_TIME)
        return;
    // 現在的時間變成last時間
    lastUpdateTime = currentUpdateTime;

    // 獲得x,y,z坐標
    mGX = event.values[0];  //X軸
    mGY = event.values[1];  //Y軸
    mGZ = event.values[2];  //Z軸

    // 獲得x,y,z的變化值
    float deltaX = mGX - lastX;
    float deltaY = mGY - lastY;
    float deltaZ = mGZ - lastZ;

    // 將現在的坐標變成last坐標
    lastX = mGX;
    lastY = mGY ;
    lastZ = mGZ;
```

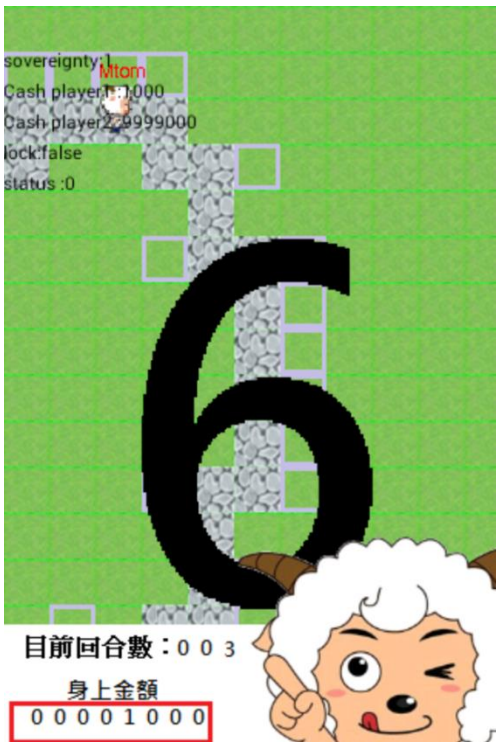
為指定的感應器註冊 SensorEventListener：呼叫 SensorManager 的 registerListener() 替指定的感應器註冊 SensorEventListener，當感應器的值變化時，SensorEventListener 的 onSensorChanged() 才會自動被呼叫。

```
// 注冊
if (sensor != null) {
    sensorManager.registerListener(this, sensor,SensorManager.SENSOR_DELAY_GAME);
}
```

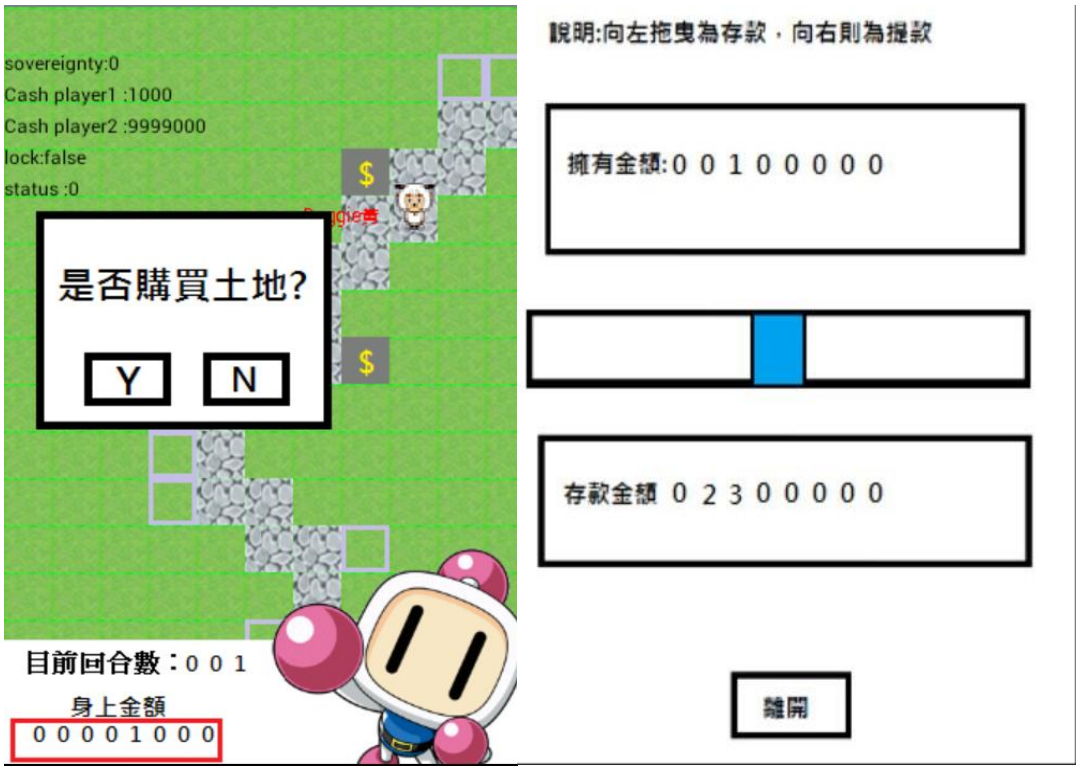
有了以上的條件之後就產生了我們需要的感應器，可以藉由實體手機感測到不同的 X,Y,Z 軸的值做出相對應的遊戲內的動作。

```
double speed = Math.sqrt(deltaX * deltaX + deltaY * deltaY + deltaZ* deltaZ)/ timeInterval * 10000;
// 達到速度閥值，發出提示
if (speed >= SPEED_SHRESHOLD) {
    onShakeListener.onShake();
}
else if(mGX>7) {  //往左傾斜
    onShakeListener.onShake_yes();
}
else if(mGX<-7) { //往右傾斜
    onShakeListener.onShake_no();
}
else if(mGY<-1.5) { //往上傾斜
    onShakeListener.onShake_check();
}
else if(mGY>9.5) { //往下傾斜
    onShakeListener.onShake_leave();
}
switch (gameState)
{
case bank:  //銀行模式下
    if(mGX>3){ //往左傾斜
        onShakeListener.onShake_yes();
    }
    else if(mGX<-3){ //往右傾斜
        onShakeListener.onShake_no();
    }
break;
}
```

當 X,Y,Z 軸到達一定程度的變化量時就會產生擲骰子的動作效果。



當我們需要選擇是或否的抉擇動作時，我們設定成「往左傾斜」為 YES 動作，「往右傾斜」為 NO 動作，或是我們進入了銀行需要做出存款提款的動作時設定時「往左傾斜」為存款動作，「往右傾斜」為提款動作。



當我們需要離開當前這個畫面的時候可以使用「往下傾斜」的動作來離開當前畫面。

肆.主要成果

影片 demo

使用到的手機型號：Butterfly(X920d)

Sony Ericsson Ray(ST18i)

Samsung Galasy S2(GT-I9100)

伍.問題與解決方法

Q：如何產生繪圖？

A：起初是用 view 來產生繪圖，後來面臨到畫面更新問題，而找到 SurfaceView 有 thread 可以刷新畫面。

Q：藍芽連線資料傳輸問題 ？

A：由於起初資料型態是 int 的,藍芽傳輸的資料型態只能用 byte 去做傳輸,但只有字串才可以使用 getbyte(),所以我們就先把資料型態 int 的資料轉成字串再使用 getbyte()來去達到資料傳輸出,而資料輸入時 getbyte()先轉成字串,再轉回 int。

Q：蓋房子時如何判斷該格子是否可以蓋房子？

A：當人物每次移動時，需要判斷人物移動格的上下左右是否有可以蓋的房子，而我們利用代碼的不一樣來區分土地等級，這樣有可以判斷該格子是否可以蓋房子。

Q：如何增加遊戲的趣味性？

A：利用事件發生的機率來使玩家對隨機的事件產生意外的衝擊，進而增加趣味性。

陸.專題製作心得

經過這次的專題製作發現不僅能培養學生發現問題、解決問題的能力，也能培養溝通協調與團隊合作的能力，例如尋找研究問題、瞭解研究方法、收集資料的方法、資料整理與分析、工作分配、時間管理、專題製作格式的認識，簡報的技巧等。

因為我們是初次作專題製作研究，起初是先觀摩學長們的歷屆作品，讓我們有了初步具體的概念以及有關研究法的書面資料，較能掌握專題的架構與格式，以及專題參考資料的寫法。另外在製作的過程中，學生需要具備基本的電腦文書處理能力，例如使用 WORD 排版、表格的呈現與繪製、PPT 的設計等，這些都是專題製作的研究之一。

在製作專題過程中，雖然辛苦且投入了很多時間，尤其在這之中修改了不少次，雖然覺得很挫折，但大家還是非常努力地完成。藉由這次的學習經驗，讓我們體會到團結、訓練口才及表達能力的重要性，相信這對畢業後的我們會是一大幫助，很感謝能有這次機會讓我們得知如何去做好一個報告及上台發表，使我們受益良多。

柒.結論與未來展望

一開始從無到有，任何資訊都不大了解，後來冷靜下來跑去詢問了去年專題製作的學長們，在他們給予的資訊與老師的教導有方之下，終於有了些許頭緒，清楚的知道我們這次該做的主題與方向，進而再去摸索需要增加的項目和功能，在某次老師的建議之下「no-touch」的概念就此衍生出來了。

只要有平衡的感應裝置，「no-touch」就可應用在此裝置上面，而在「no-touch」的概念之下，未來很多科技也許可以利用此概念，來創造出勝於聲控的控制指令，例如：開車不須碰觸方向盤，也許可利用３Ｄ座椅來控制車子前進的方向與速度，進而減少肢殘人士開車方面的困擾與負擔。

相信在未來的科技進步之下，「no-touch」一定可以帶給人們更多的便利。

捌.致謝

本專題報告得以順利完成，首先要感謝陳建宏指導教授一路細心耐心的協助幫助我們，且定時撥空出時間和我們討論本專題的製作，給予意見，讓本專題得以順利完成。

其次，要感謝學校的學長提供我們關於專題上所需的相關資料及資源，使我們的專題能順利進行。

也感謝組員們的分工合作及資料收集，在製作專題期間的陪伴及互相加油打氣。

最後也要感謝學校所提供的書籍與設備，在專題製作過程中才能更加完整，也能更順利的完成，在此本組特別的感謝，致上最高的敬意與謝意。

玖.參考文獻

Android 參考書：

新觀念 Android SDK 程式設計範例教本

Google ! Android 3 手機應用程式設計入門 第四版

參考網頁：

維基百科 Eclipse http://zh.wikipedia.org/wiki/Eclipse

維基百科 Android http://zh.wikipedia.org/wiki/Android

維基百科 Monopoly http://zh.wikipedia.org/wiki/Monopoly

台大批踢踢 AndroidDev 版：http://www.ptt.cc/bbs/AndroidDev/index.html

android developer：http://developer.android.com/reference/java/lang/System.html