物件導向軟體工程 –
期末小專題

Android – 中國象棋與台灣暗棋

B 組

2013/12/26

目錄

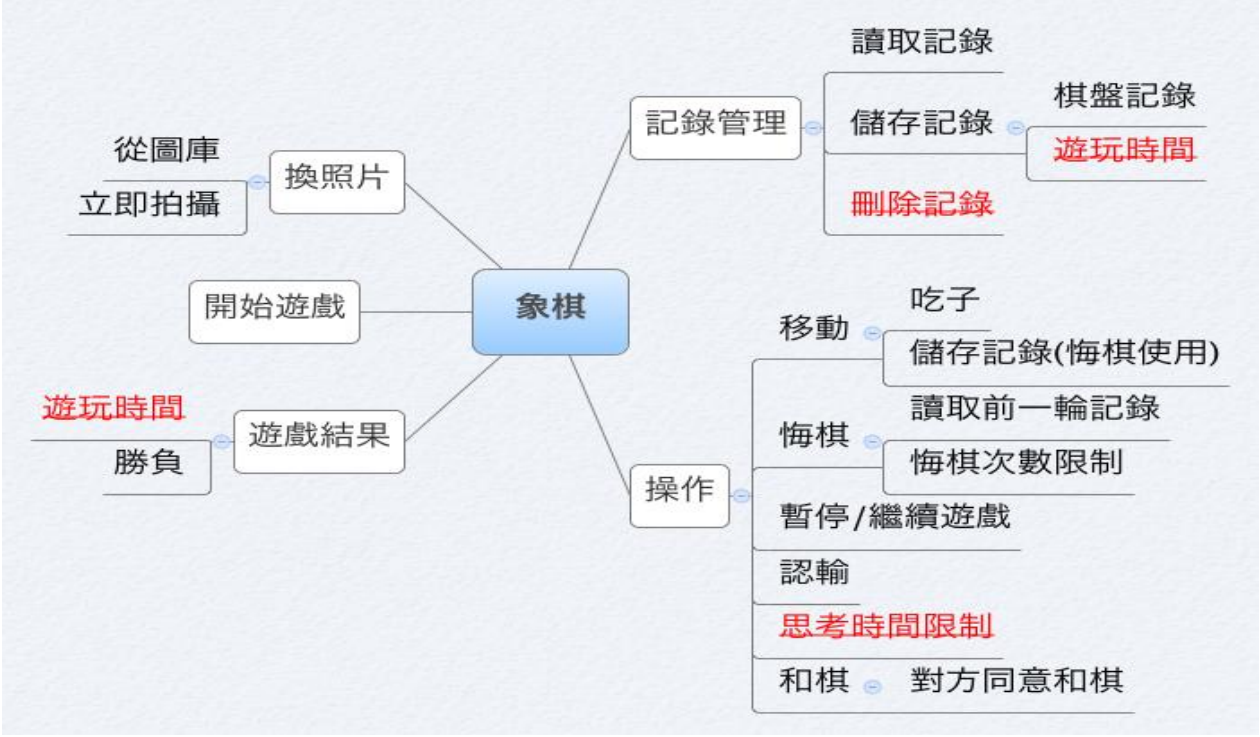
1. 工作分配說明

鄭兆廷	D0078154	Programmer
盧威辰	D0031314	Code Review
陳映如	D0071317	Refactoring, UI Design
徐郁婷	D0071303	Bug Test
洪筱惠	D0001059	Documentation

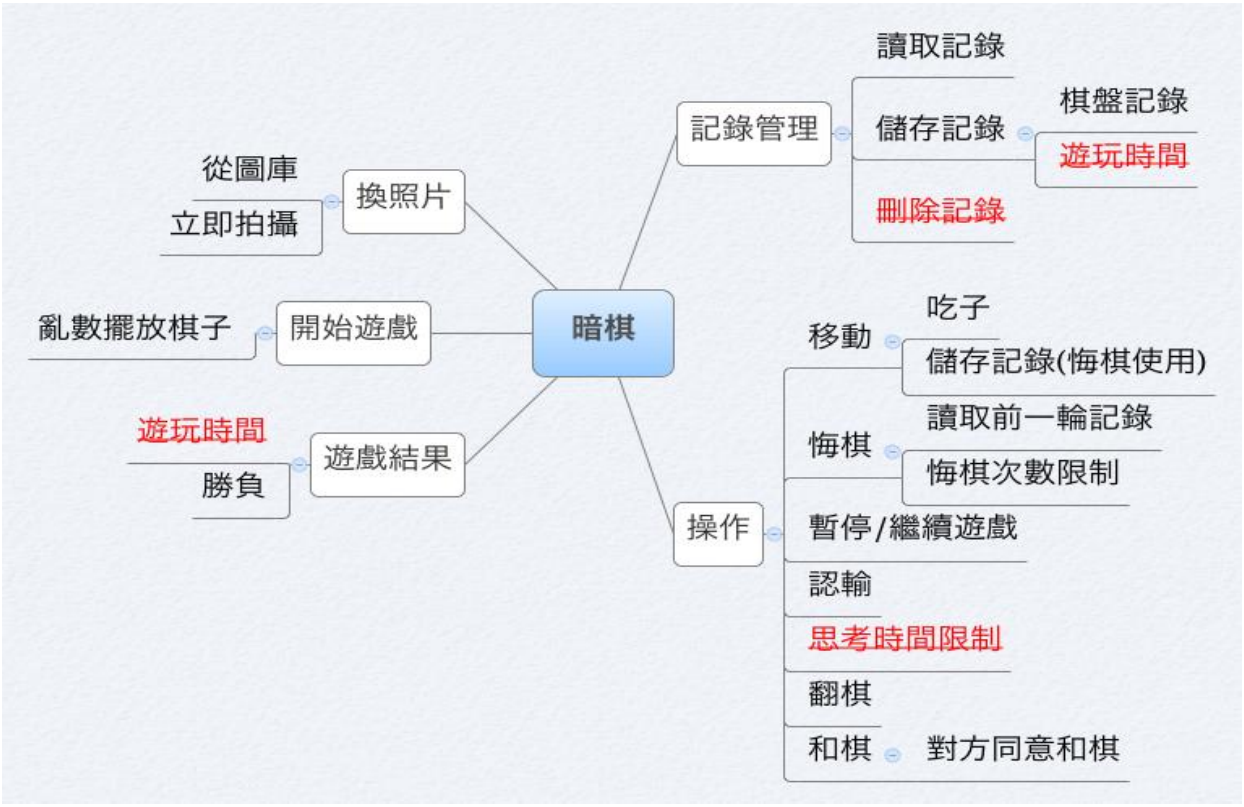
2. 功能說明與功能特色

2.1 功能樹

象棋



暗棋



2.2 品質樹

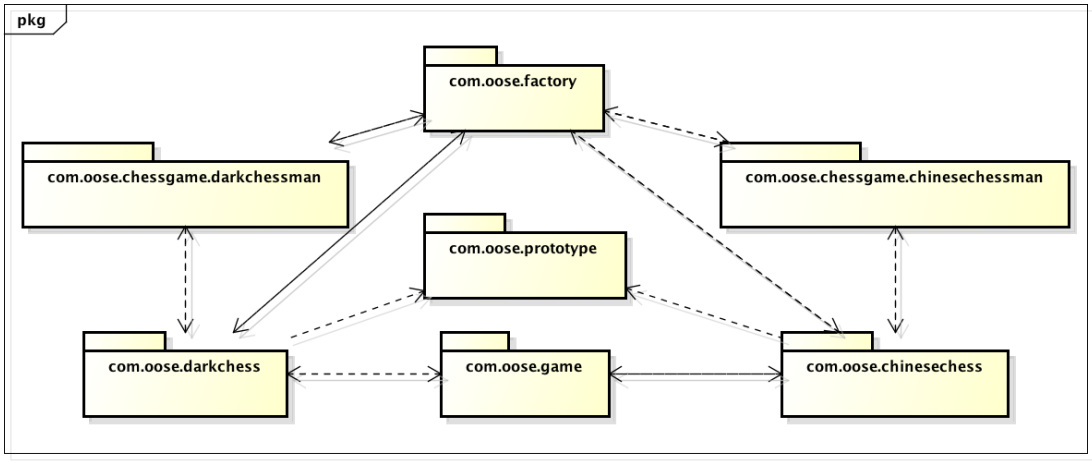
3. 設計說明

3.1 程式背景資料

使用 Android 4.0.3 版本，以 Java 搭配物件導向方式開發，並選用 HTC One V 做爲測試手機。

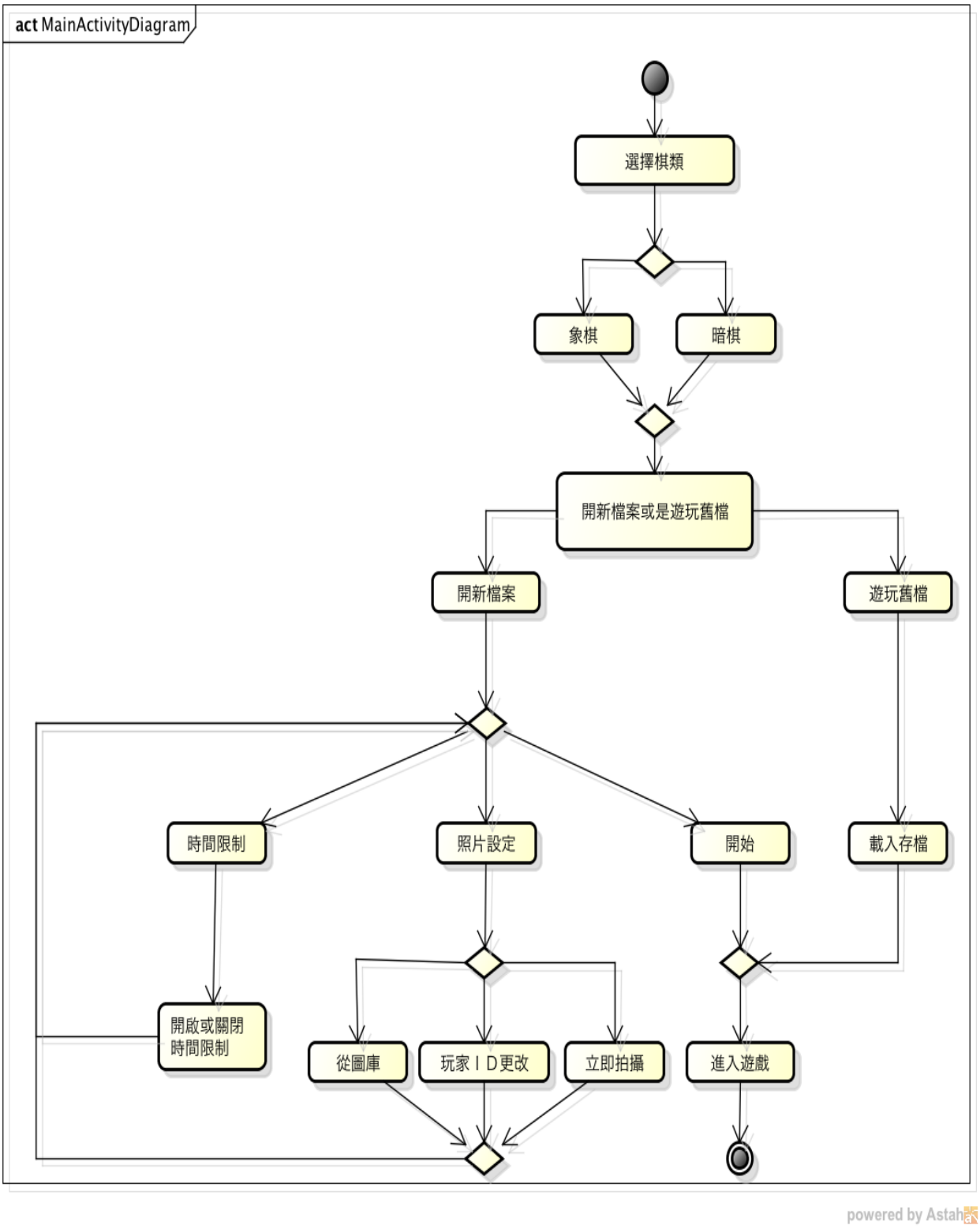
3.2 軟體架構

3.2.1 Package Diagram

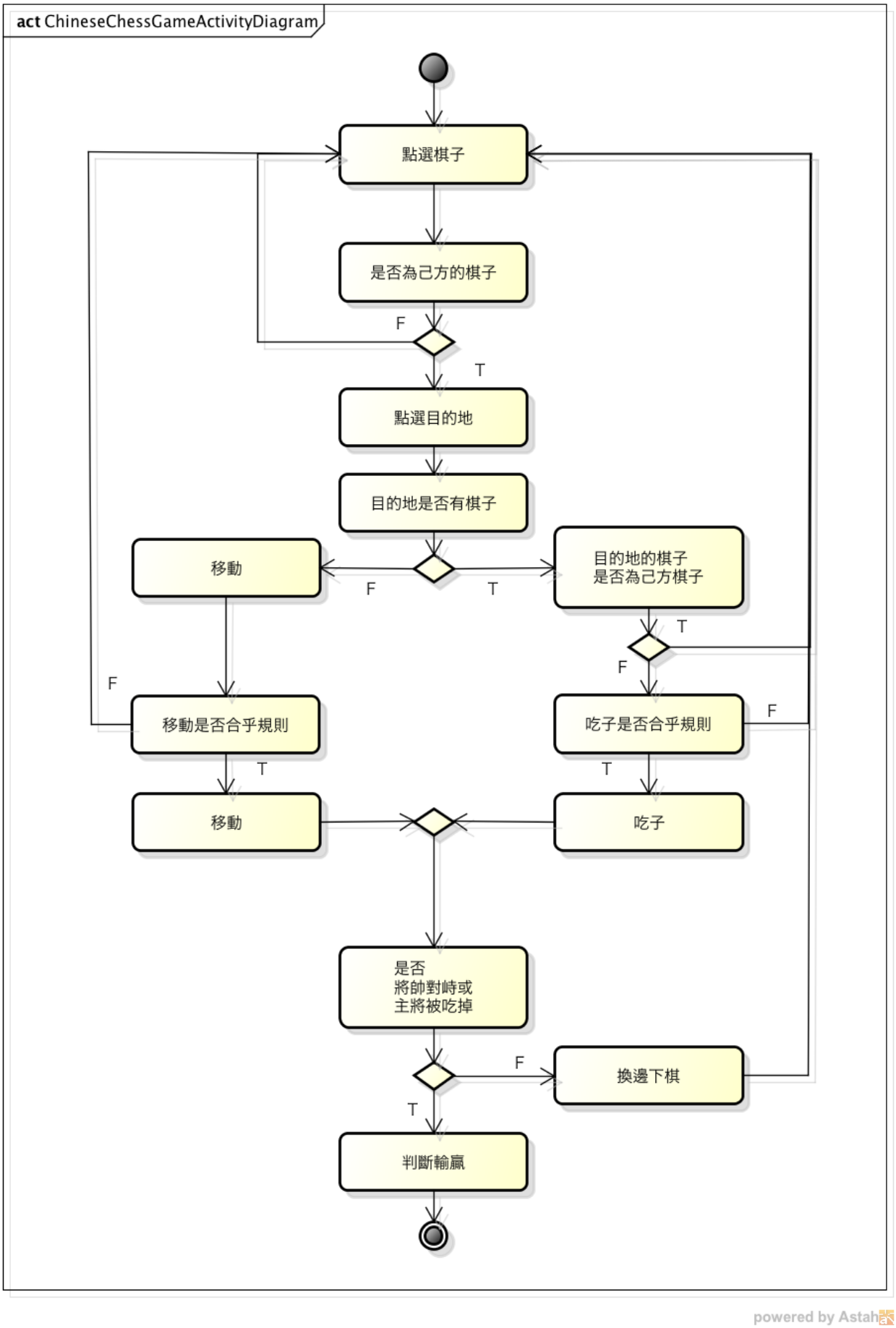


3.2.2 Activity Diagram

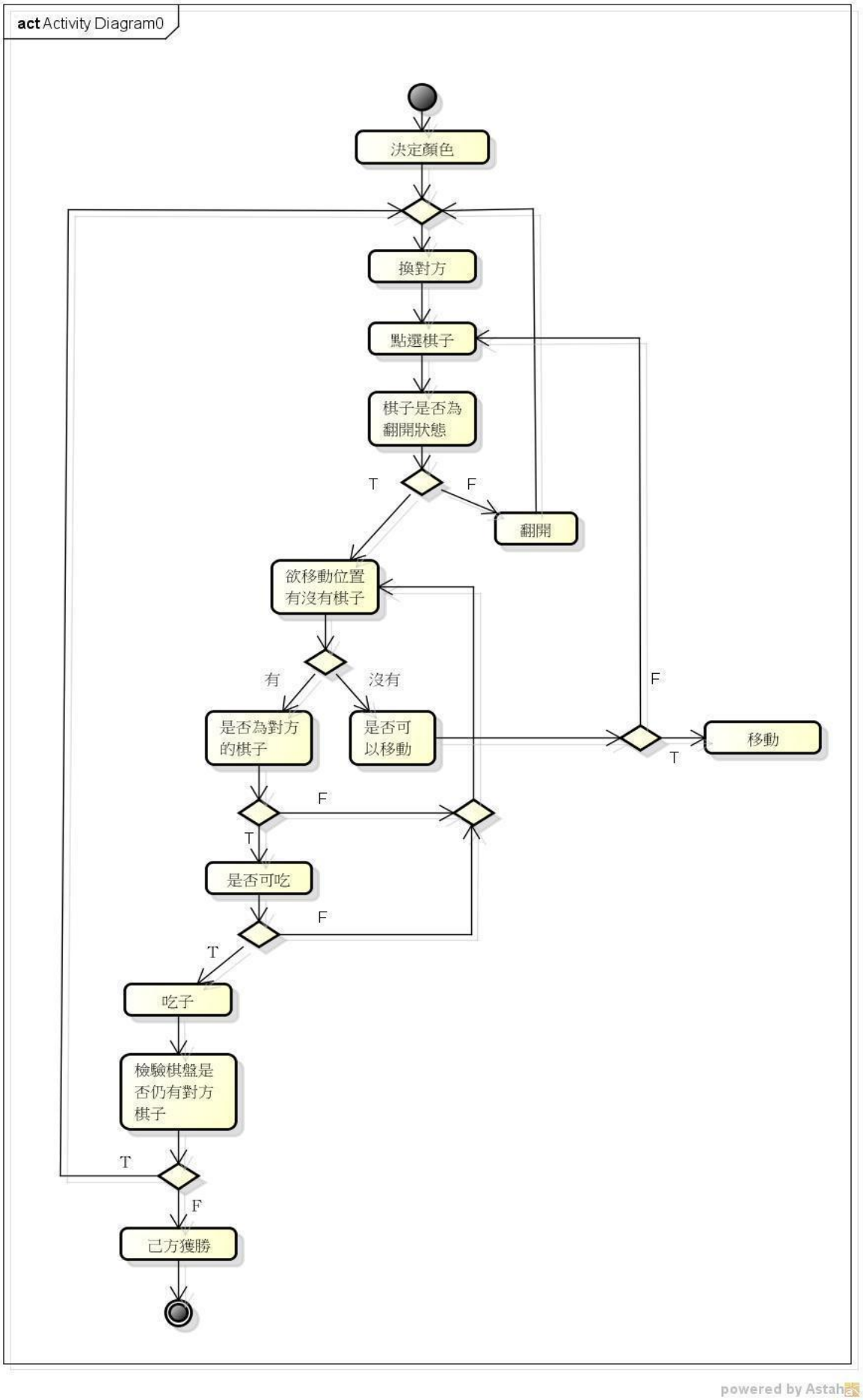
遊戲開始與參數設定流程



中國象棋遊戲流程

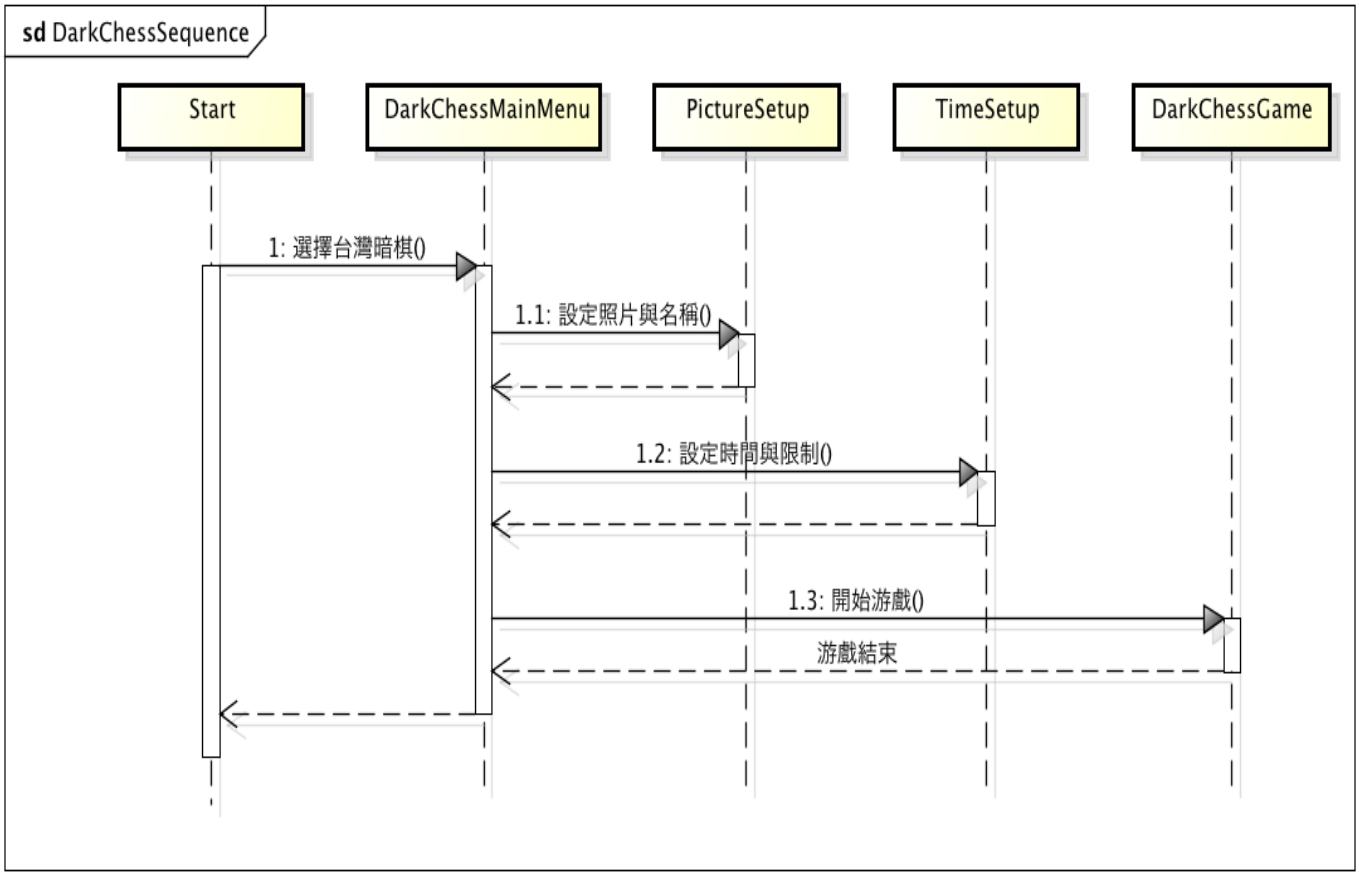

act ChineseChessGameActivityDiagram
點選棋子
是否為己方的棋子
F
T
點選目的地
目的地是否有棋子
移動
F
T
目的地的棋子
是否為己方棋子
T
F
F
移動是否合乎規則
T
吃子是否合乎規則
F
T
移動
吃子
是否
將帥對峙或
主將被吃掉
F
換邊下棋
T
判斷輸贏
powered by Astah

台灣暗棋遊戲流程


act Activity Diagram0
決定顏色
換對方
點選棋子
棋子是否為翻開狀態
T
F
翻開
欲移動位置有沒有棋子
有
沒有
是否為對方的棋子
是否可以移動
F
T
移動
F
T
是否可吃
F
T
吃子
檢驗棋盤是否仍有對方棋子
T
F
己方獲勝
powered by Astah

3.2.3 Sequence Diagram

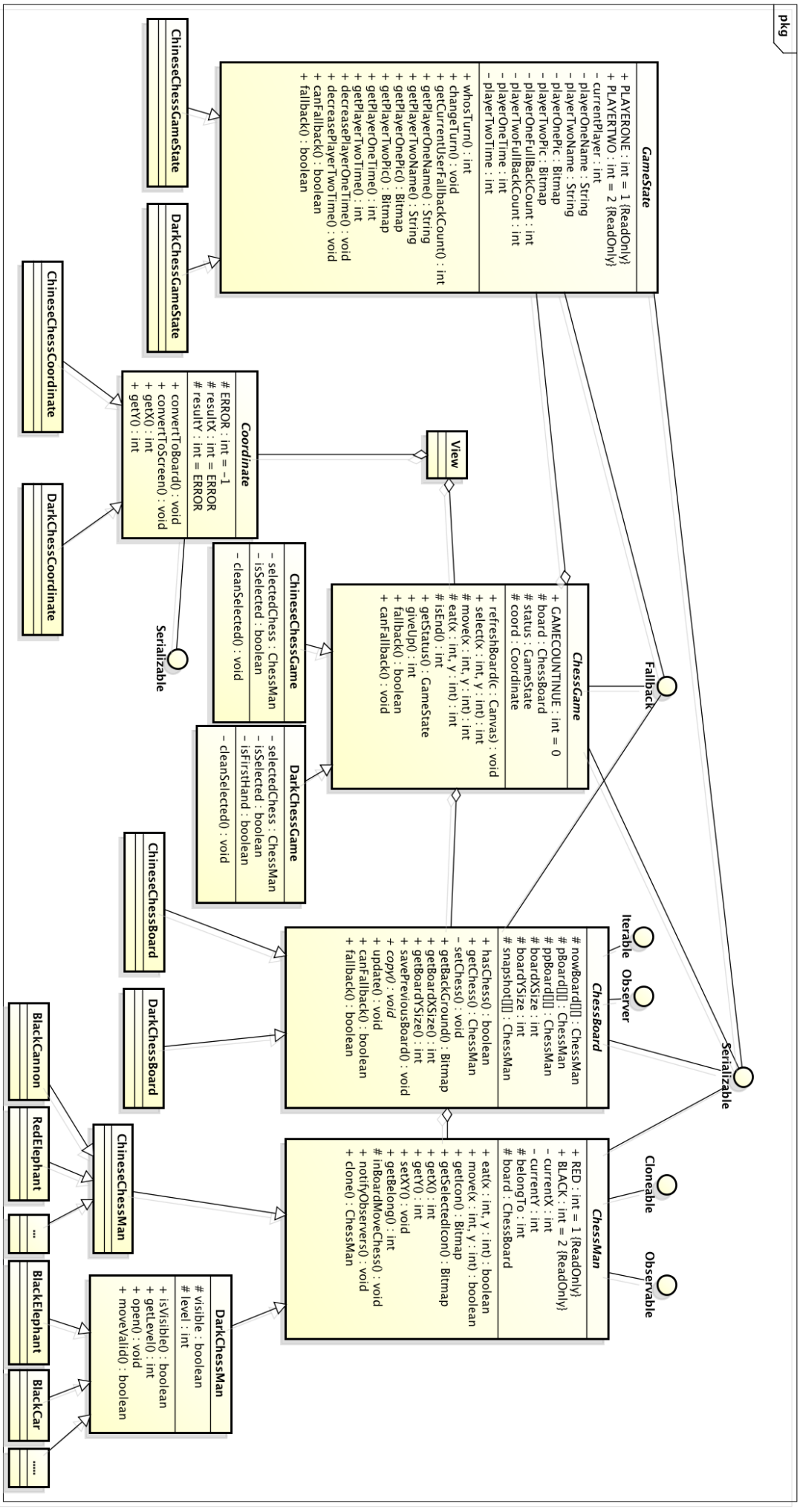
遊戲啓動至遊戲結束流程(以台灣暗棋爲例)



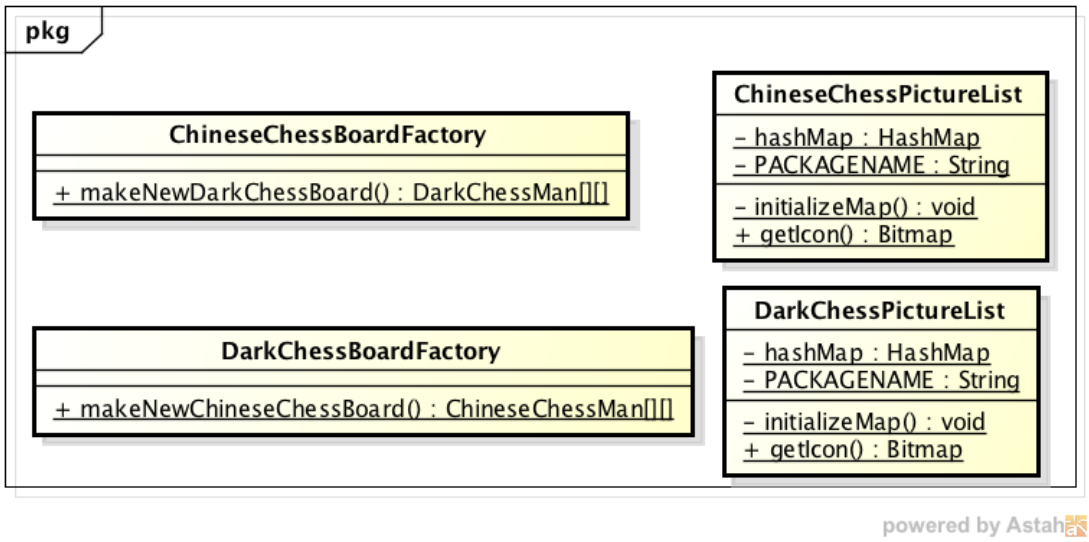
3.3 物件設計

3.3.1 Class Diagram

主要設計



各 Factory 與棋盤、棋子圖案 LookupTable



3.4 演算法設計

3.4.1 螢幕點選位置轉換程式座標

```
if (螢幕點選位置未超出棋盤範圍) {
        X 座標 = 點選位置/每格寬度
        if(X 座標 = 9) { //棋盤設計邊框問題解決
                X 座標 - 1
        }
        Y 座標 = 點選位置/每格長度
        if(Y 座標 = 10) { //棋盤設計邊框問題解決
                座標 - 1
        }
} else {
        回傳錯誤訊息
}
```

3.4.2 中國象棋 —— 炮的棋子走法

● 移動：掃描點選的棋子到目的地的線性範圍，如果直線範圍或橫線範圍內有棋子則取消移動，若沒有棋子則移動到目的地。

● 吃子：掃描點選的棋子到目的地的線性範圍，並搭配一個布林值判斷是否隔著一個棋子，如果直線範圍或橫線範圍內掃到多個棋子或沒有掃到棋子都取消移動，若只有一個棋子則成功吃子。

3.4.3 中國象棋 —— 判斷勝負

●將帥對峙：先抓黑將，再做直行的線性掃描，若同一行只有另一顆棋子且爲紅帥，判定爲將帥對峙

●主將被吃：即黑將或紅帥被吃

→ 綜合以上兩種情況，只要符合其中一種即判斷輸贏

●投降：直接結束並判定對方獲勝

●和局：詢問對方是否同意，同意則直接結束遊戲。

3.4.4 悔棋

利用三個二維陣列分別存放，現在的棋盤狀態、前一步的棋盤狀態、前兩步的棋盤狀態。先判斷是否有前兩步的棋盤狀態，如果有 就可以悔棋，並把現在的棋盤狀態改成前兩步的棋盤狀態。

3.4.5 取得圖像資源

利用 HashMap 建構一個可取得圖像資源的表，以方便圖像資源取得。

3.4.6 存檔

將其設計之 ChessGame 等物件序列化後，呼叫 Java 系統函式庫來儲存資料，當使用者要載入存檔時，Android 的 Activity 在初始化 ChessGame 等新物件時，會去讀取已儲存的 ChessGame 等物件資訊，使用其物件來當作 Reference，用此來完成存檔功能。

3.5 Design Pattern 採用說明

3.5.1 Simple Factory

使用其觀念來創造出一個新的棋盤。

採用地方：ChineseChessBoardFactory.java, DarkChessBoardFactory.java

3.5.2 Singleton

在圖像資源的 Lookup Table 中只有一個 static 的 HashMap

採用地方：ChineseChessPictureList.java, DarkChessPictureList.java

3.5.3 Observer

使用 Observer 來控制棋子和棋盤的互動。把每個棋子都當成 Observable，而棋盤是 Observer。

採用地方：ChineseChsssMan.java, ChineseChessBoard.java, DarkChessMan.java.
DarkChessBoard.java

3.5.4 MVC

使用 MVC 來將 Model、View、Controller 分離，並賦予它們各自的責任，使程式結構更有擴充性。

採用地方：ChessBoard.java (Model), Android View(View), ChessGame(Controller)

3.5.5 Iterator

使用 Iterator 來讓 View 與 Controller 可以循序取得棋子，在棋盤內實作一個 Iterator，並可知道有無下一個棋子和可循序取得下一個棋子。

採用地方：ChessBoard iterator ChessMan

3.5.6 Template Method

使用 Template Method 來將整體的骨架定義出來包成一個 package，並保留部分步驟留給子類別去實作。

採用地方：Package “com.oose.prototype”

3.5.7 Flyweight

使用 HashMap 來共用重複使用到物件，避免每次用到都去新建一個物件。

採用地方：ChineseChessPictureList.java, DarkChessPictureList.java

4. 成果

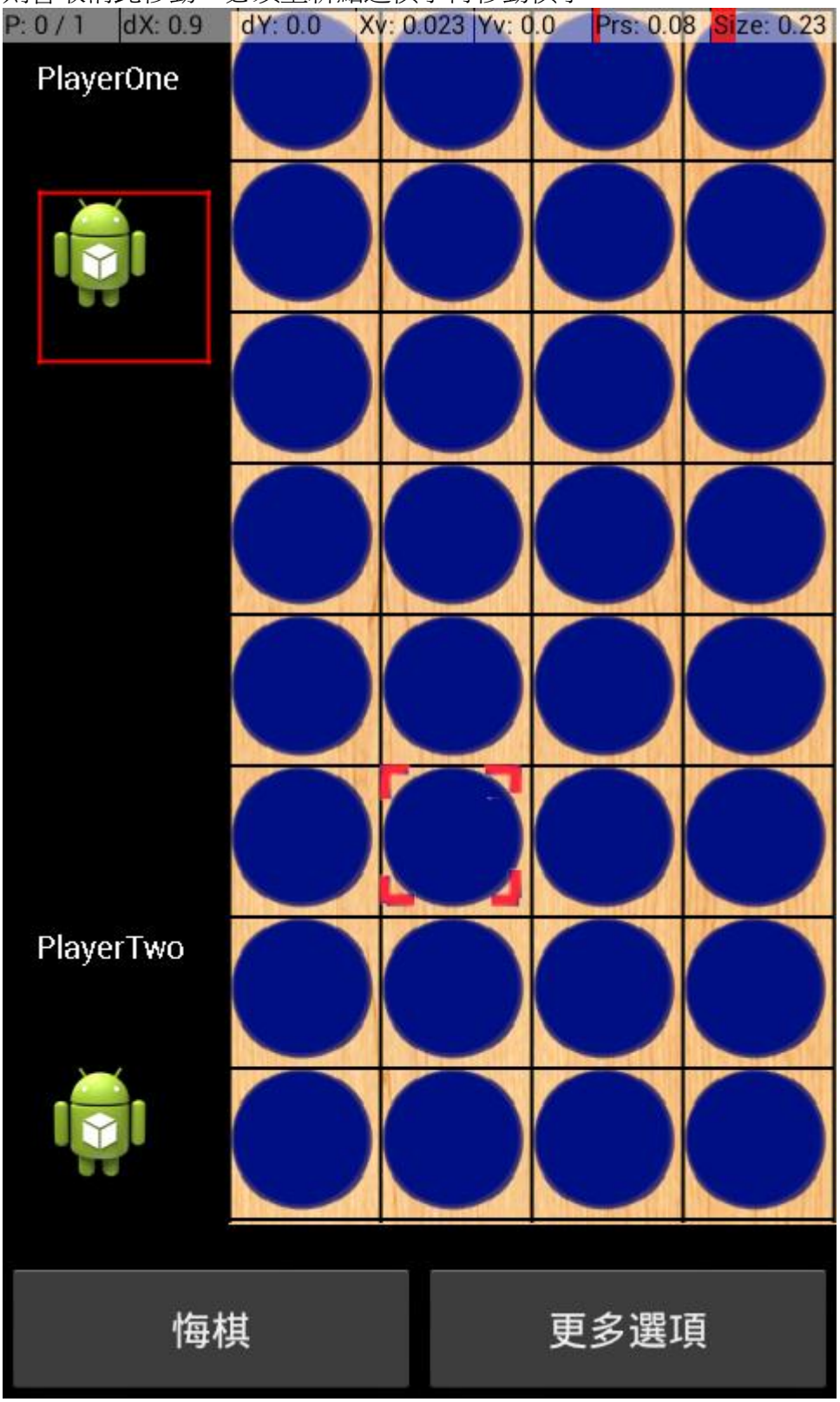
4.1 已完成功能

4.1.1 照片設定

玩家名稱預設爲「PlayerOne」和「PlayerTwo」，可讓玩家自行命名玩家名稱。玩家照片可從手機圖　　庫中選取或相機立即拍攝照片

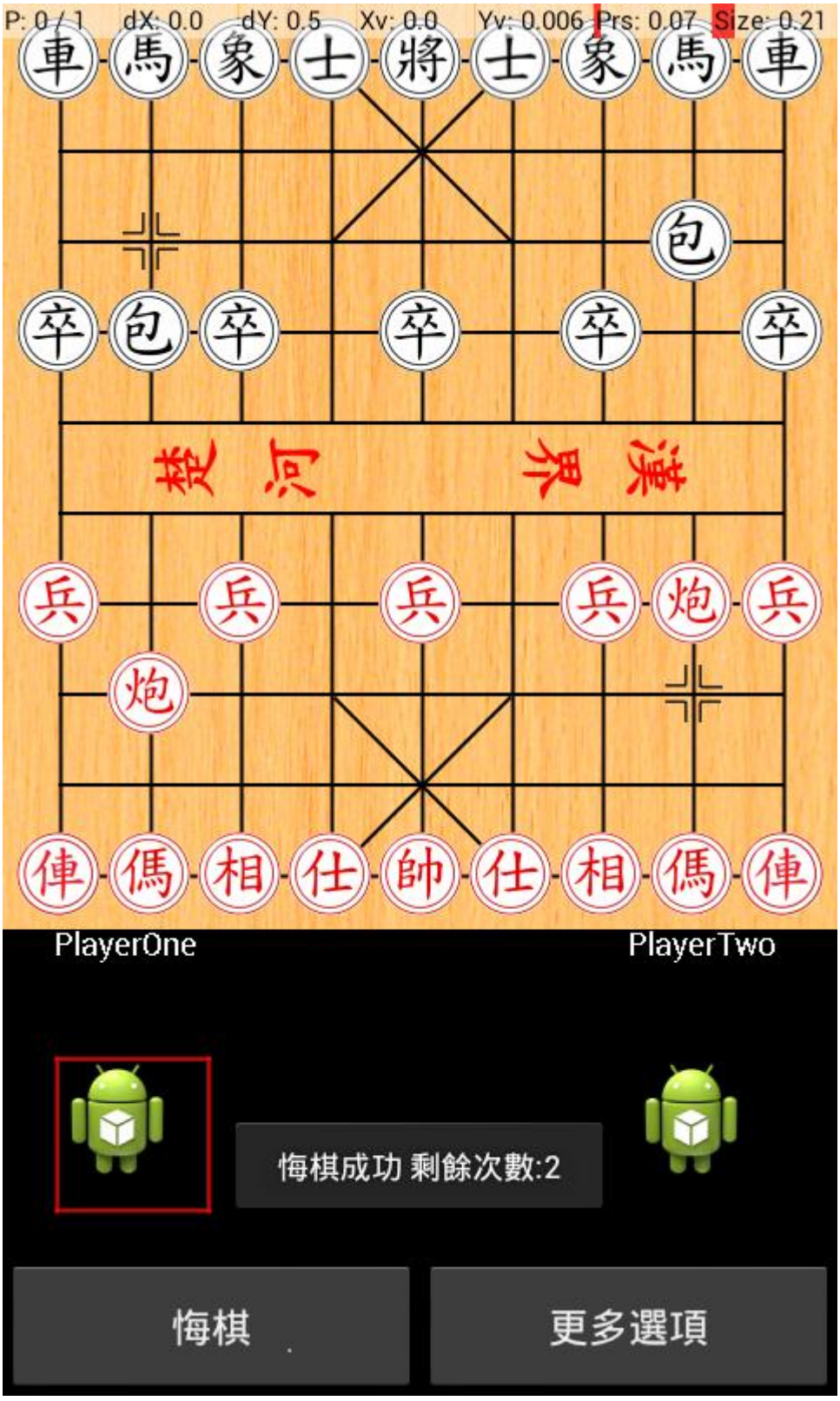
4.1.2 移動棋子

點選棋子後，會將棋子框起，之後點選要移動到的目的地即可移動棋子。若此移動違反移動規則，則會取消此移動，必須重新點選棋子再移動棋子



4.1.3 悔棋

紅黑雙方均有 3 次悔棋機會，除了棋局剛開始雙方玩家的第一手棋之外都可使用悔棋功能。其中一方點選悔棋後，棋盤狀態將會回溯到該玩家的前一回合

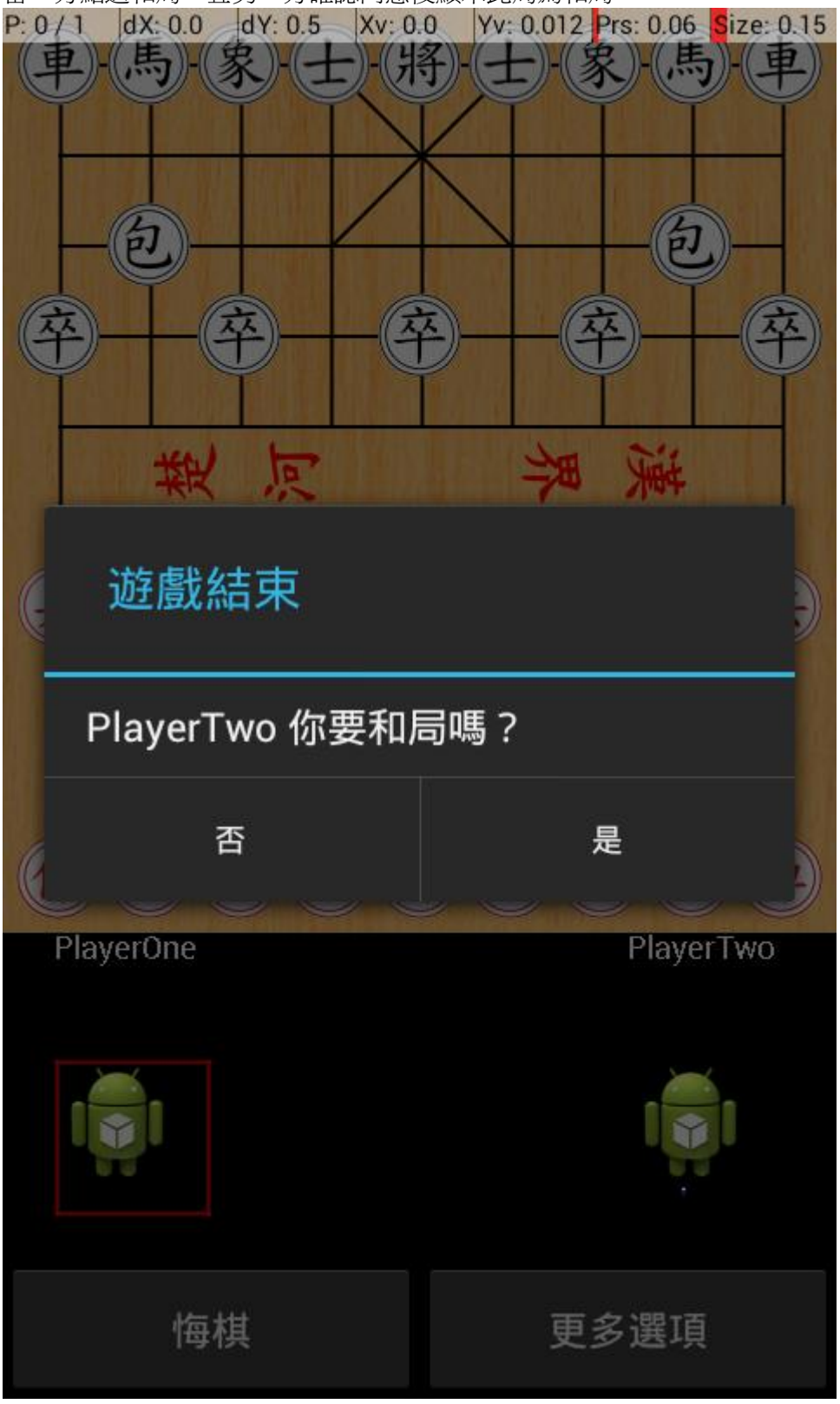


4.1.4 投降

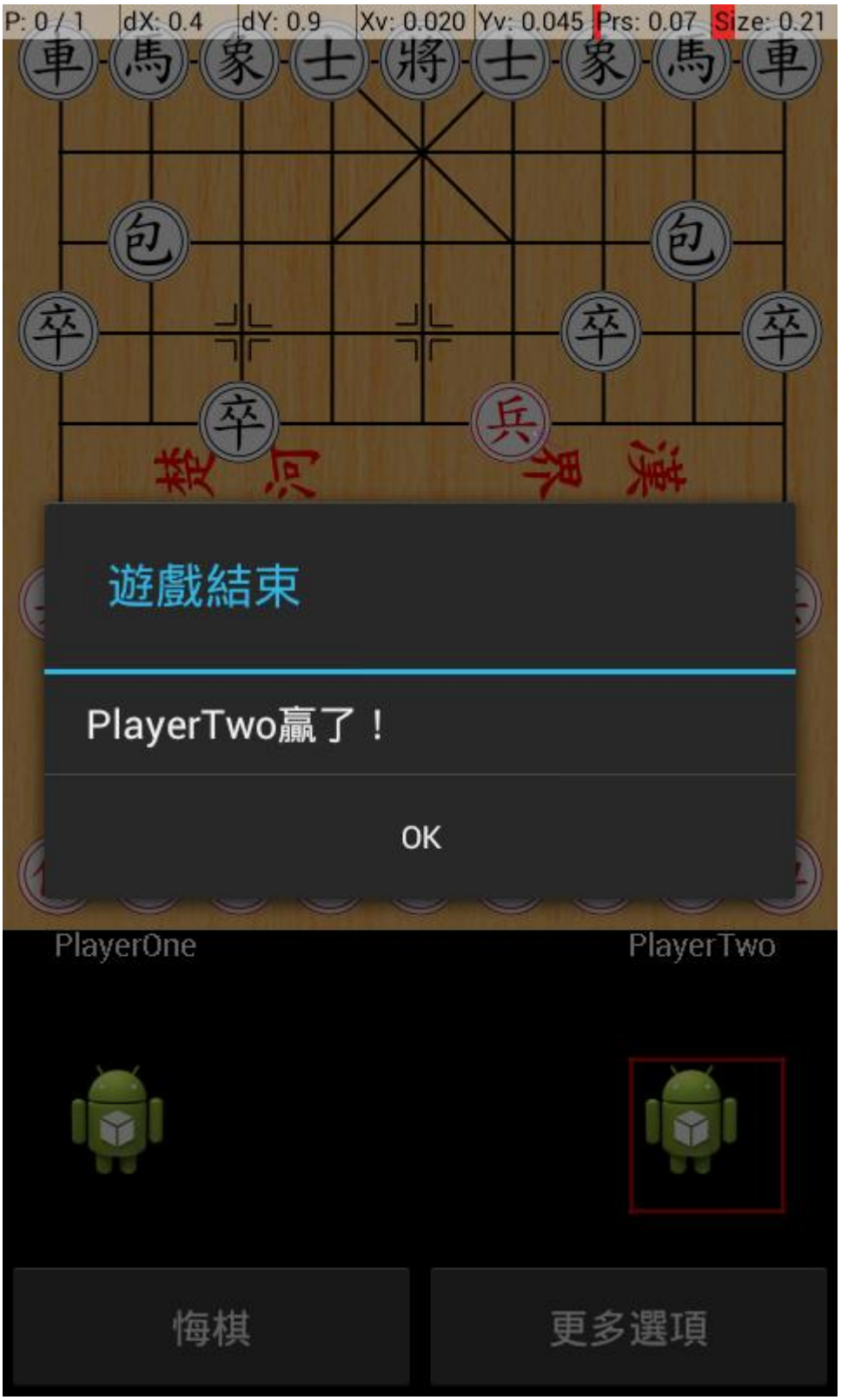
當玩家在自己的回合點選認輸，則顯示另一方獲勝，並結束遊戲

4.1.5 和局

當一方點選和局，且另一方確認同意後顯示此局爲和局



4.1.6 遊戲結果


P: 0 / 1
dX: 0.4
dY: 0.9
Xv: 0.020
Yv: 0.045
Prs: 0.07
Size: 0.21
遊戲結束
PlayerTwo贏了！
OK
PlayerOne
PlayerTwo
悔棋
更多選項

4.1.7 讀取紀錄

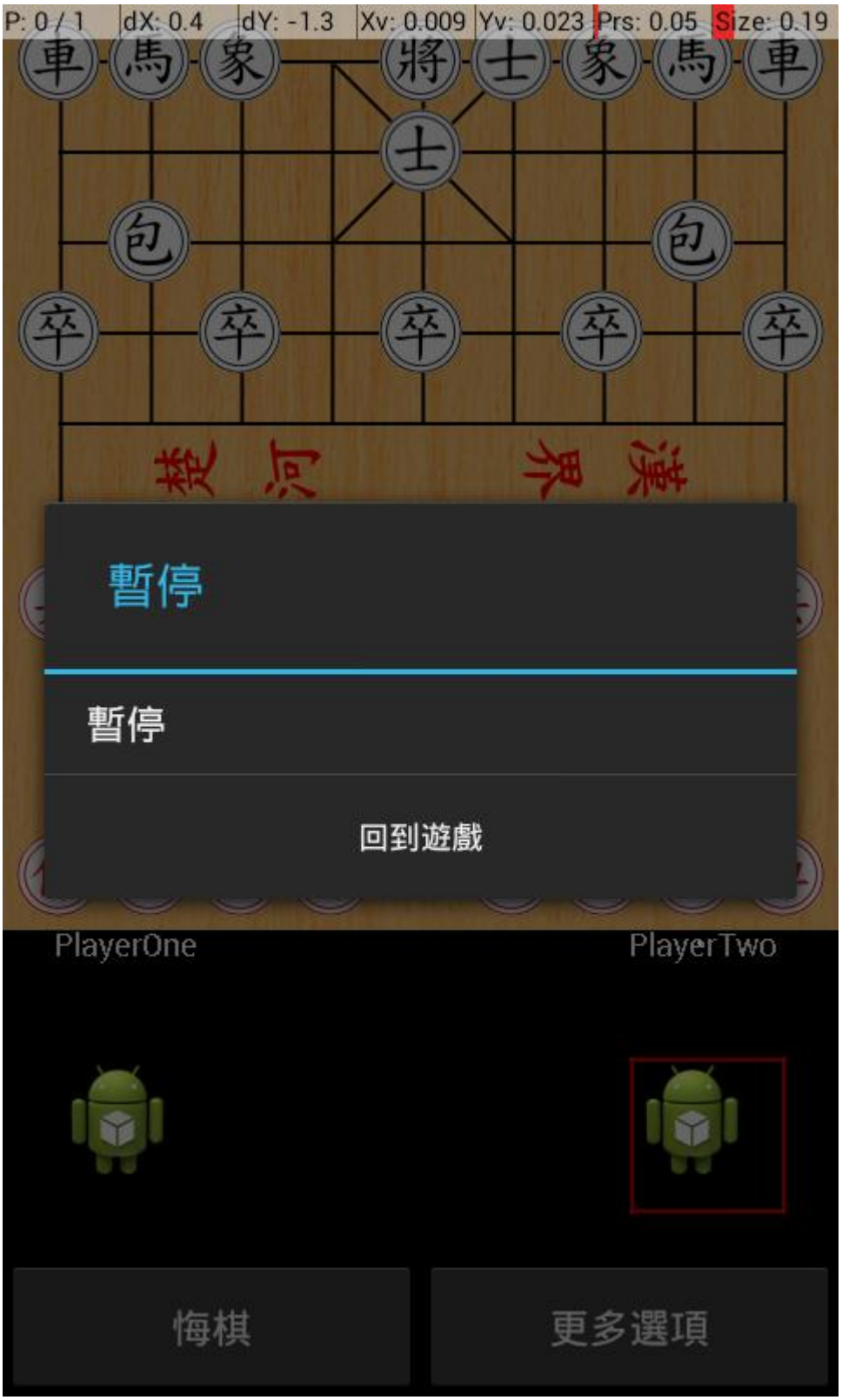
玩家可從記錄檔讀取記錄，並開始遊戲

4.1.8 亂數擺放棋子

設計暗棋時，使用亂數擺放棋子，使得每局暗棋的棋子擺放位置都不相同。

4.1.9 暫停 / 繼續遊戲

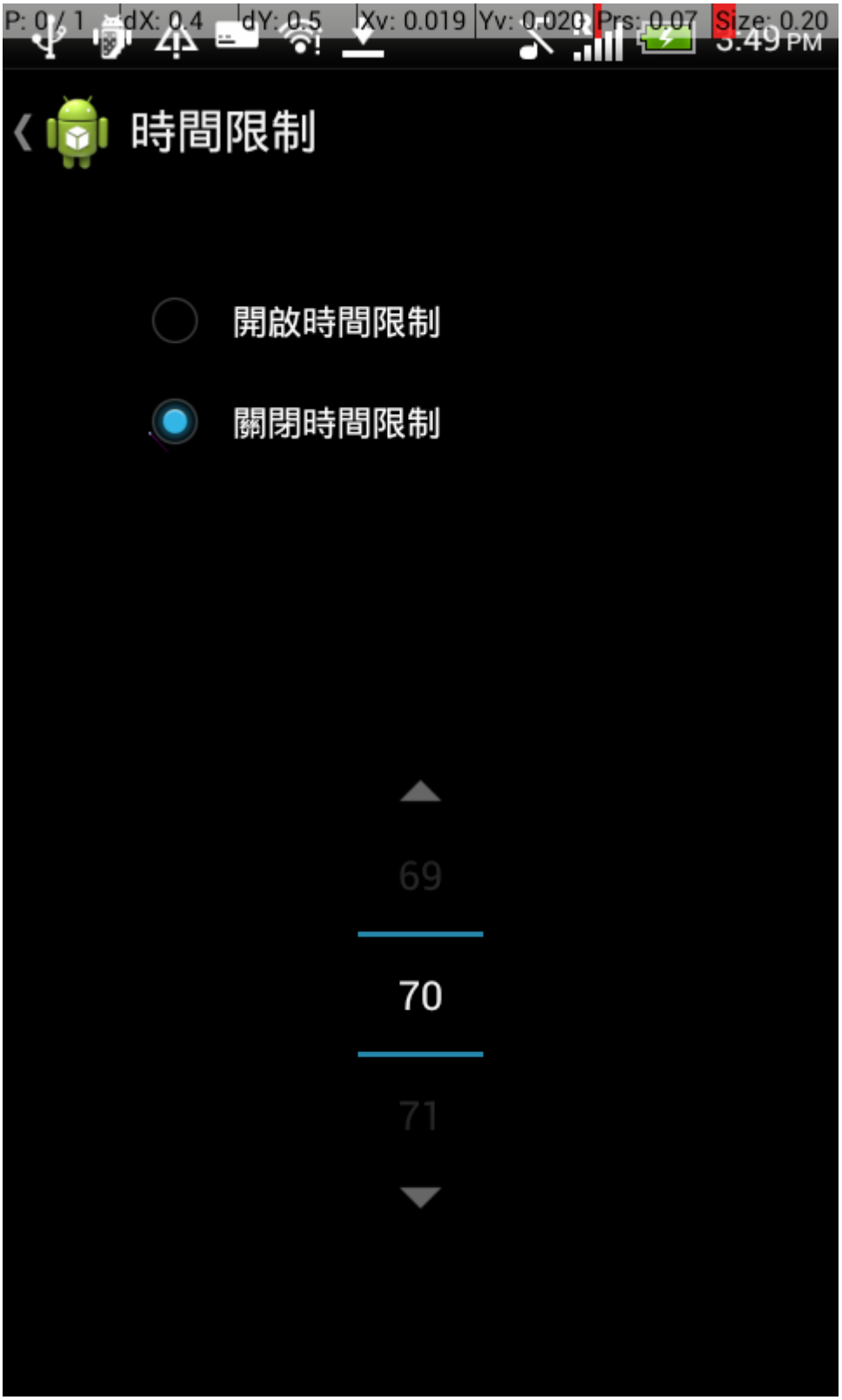
選取後，將會跳出一視窗提示遊戲暫停，按下按鈕後可繼續遊戲



4.2 未完成功能

4.2.1 限時設定(半完成)

完成時間設定的介面，在遊戲進行中的限時未完成



4.2.2 思考時間限制

因遊戲進行中的限時未完成，所以未完成「思考時間限制」的功能

4.3 軟體相關計量資訊

程式碼行數: 3931 行
Method 數量: 344 個
Class 數量: 74 個
平均每個 Class 所包含的 Method 數量: 4.649 個

4.4 R2 口頭報告後的修改/新增說明

R2 時只有實作完成中國象棋，並且棋盤爲橫向擺放。在 R3 時，中國象棋的棋盤改爲直向擺放，並且重新設計顯示介面，以更方便遊玩

5. 心得

5.1 系統重用性是否良好

系統重用性包括了設計 Template Method, Don't Repeat Yourself 等等物件導向設計原則，我們在設計遊戲時，先分析象棋與暗棋的相同地方，如都有棋盤、棋子、遊玩人數皆爲兩人等等來撰寫 Template Method，這樣就可以達到兩種棋類遊戲，使用了同一個模板的重用性要求，並且使用 DRY 原則，進一步分析，例如兩種遊戲都有棋盤，其棋子的移動 Method 應該是可以互通的，所以就將其詳細協作寫入了模板，以提高系統的重用性。

5.2 系統擴充性是否良好

爲了驗證系統的擴充性，我們做了兩種嘗試，第一爲附加功能的 interface 化，我們把悔棋功能分析並設計成 interface，其包含了兩個 Method，canFallback()與 fallback()，分別代表了判斷目前狀態可否悔棋與執行悔棋動作，之後我們將其實作於各模組中，來測試我們設計之模組是否具備可擴充性，最後實作後試驗是沒有問題的，證明我們的模組是具有擴充性的，第二爲 Templete Method 的設計後實作，我們可使用其設計好的 abstract ChessGame 等 class，來繼承與實做中國象棋與台灣暗棋，此兩個實驗應可證明此次設計的系統擴充性應具備一定程度。

5.3 系統模組切割是否良好？分工與整合是否容易

此次的系統模組切割成了五個大部分，棋盤、遊戲狀態、坐標轉換、遊戲主流程和遊戲棋子，之後即可將各部分分工後整合，在開始分工之前，我們有先詳細討論過其各模組的相互關係與界接方法，所以在模組分工實作與相互整合時並沒有出現太大的問題。

5.4 物件設計原則是否協助我系統的開發？

有的，此次設計大致上有遵照兩個開發原則，分別爲 Don't Repeat Yourself 與 The Open-Closed Principle，DRY 爲在設計兩個棋類遊戲時，如果實作時的演算法有相似之處，我們會將其寫入到 abstract class 中，遵循 DRY 原則。OCP 原則的遵守則是在 DarkChessMan(暗棋棋子裡)，暗棋棋子繼承了 abatract ChessMan 後，其內所定義的 Field 與 Method 不足以完成暗棋棋子的功能，所以我們在擴充暗棋棋子的功能時，是將暗棋所需要之功能新增於繼承後的類別中，而不是將其寫入上層的 abstract ChessMan 中，因爲其新增的功能並不一定其他的實作類別會使用到，在此部分，我們遵守了 OCP 原則。

5.5 設計樣式是否協助我系統的開發？

Design Pattern 有著大大的幫助，在我們分析與細分模組之後，我們在沒有將程式依照 Design Pattern 設計時，雖然其系統是可以正常運作，但是模組間的擴充性與接合程度實際上非常的差，在 R2 之後，我們就先了解上課時所學與一些參考資料上的 Design Pattern，嘗試將我們的系統部分使用其 Pattern 來做整理與各模組重新界接，最後發現遵照了 Design Pattern 後，其程式的可讀性、擴充性與彈性大幅的增加了，而最佳的一個例子爲，我們的暗棋遊戲部分，是在最終報告前五天才開始撰寫，如果沒有使用 Design Pattern 來設計系統，那麼最終我們暗棋實作的部分一定是無法完成的！

5.6 版本管理使用心得

在專案中導入的版本控制是一件很有趣的事情，我們這次的開發使用了 Android Developer Tools (base on Eclipse)、EGit 外掛與 GitHub 網站(以 Git 版本控制系統爲基礎的版本控制平

台)，來使我們的小專題導入版本控制，導入版本控制的優點在於當組員之間要分享程式碼時，只需將修改完後的程式 Push 至 GitHub 後，其他人再 Pull 後即可完成其同步的動作，還有其分支的功能也幫助非常的大，我們可以使用分支，各組員自行開啓屬於自己的分支，當負責的部分程式撰寫完，Push 至 GitHub 後，再由一人來分析與 Merge 各組員的成果，來達到開發時，總是能有一個可執行的程式版本，與實現多人撰寫各部分模組時，不會因爲各自的模組新增修改而干擾到了其他組員的工作，最後還有可復原式的設計，當程式運作不正常時，我們可以回復到其上次的修改狀態，來復原或是比較程式碼，來發現是什麼因素造成了程式不正常，在除錯時也方便不少，最後，導入版本控制，真的對我們的小專題製作效率提升非常的多！

6. 小專題程式版本控制 Repository

https://github.com/gnehcmit/oosegame/releases

目前程式是以 Android 4.0.4 系統與解析度 480 x 800 爲準。
當需使用 Android Virtual Device 啓動時，
請使用 480 x 800 與系統版本超過 4.0 以上系列之模擬器來使用
(例如: 3.7” WVGA (480 x 800: hdpi) with Android 4.3 – API Level 18 搭配 512 Mb Ram)
不然程式可能會有不相容的情況發生！