

# Logical Operators: Chinese Checkers

Iteration four

April 3<sup>rd</sup>, 2014

Curtis Smith

Peter Pobjewski

Saajid Mohammed

Chris Kellendonk

James Kostiuk

Kuba Subczynski

Ben Stitt

Taylor Whited

# 1 Table of contents

1	Table of contents .....	1
2	Planning.....	4
2.1	Meeting minutes .....	4
2.11	March 20, 2014 .....	4
2.12	March 25, 2014 .....	5
2.13	April 1, 2014.....	5
2.2	Project tasks .....	7
2.3	Burn downs and ups.....	8
2.31	Iteration four progress.....	8
2.32	Iteration three versus iteration four.....	8
2.4	Presentation .....	9
3	Design.....	10
3.1	Artificial intelligence (AI) .....	10
3.11	Easy .....	10
3.12	Medium.....	10
3.13	Hard.....	11
3.2	Database.....	11
3.21	Control flow .....	11
3.22	MySQL stored procedures .....	11
3.3	Network API .....	13
3.31	User .....	13
3.32	Game .....	14
3.4	Visual .....	16
3.41	Introduction .....	16
3.42	Idea.....	16
3.43	Screenshots.....	16

4	Testing.....	17
4.1	Iteration four testing plan .....	17
4.11	Revision history.....	17
4.12	Introduction .....	17
4.13	Objectives.....	17
4.14	Testing phases.....	18
4.15	Testing feedback procedure .....	19
4.16	Features to be tested.....	19
4.17	Features not to be tested .....	19
4.18	Dependencies.....	19
4.19	Tools.....	20
4.110	Approvals .....	20
4.2	Manual testing .....	21
4.3	Regression testing.....	21
4.31	Feedback.....	21
4.32	Main tests .....	21
4.33	UI engine tests .....	22
4.34	Tests summary .....	22
4.4	Ease of use testing.....	23
4.41	Tasks.....	23
4.42	Results.....	23
4.5	Acceptance testing.....	24
4.51	Introduction .....	24
4.52	Project description.....	24
4.53	Test team personnel .....	24
4.54	Acceptance test plan.....	25
4.55	Notes.....	26
5	Reflection .....	27
5.1	Introduction.....	27

5.2	Communication .....	27
5.3	Planning.....	28
5.4	Analysis, Design, Development and Testing .....	29
5.5	Conclusion .....	29
6	Appendices.....	30
6.1	Appendix A .....	31
6.2	Appendix B .....	32
6.3	Appendix C .....	33
6.4	Appendix D .....	40
6.5	Appendix E.....	42
6.6	Appendix F.....	45
6.7	Appendix G .....	58
6.8	Appendix H .....	59
6.9	Appendix I.....	60
6.10	Appendix J.....	61
6.11	Appendix K.....	62
6.12	Appendix L .....	63
6.13	Appendix M.....	64
6.14	Appendix N .....	65
6.15	Appendix O .....	66
6.16	Appendix P .....	67
6.17	Appendix Q .....	68
6.18	Appendix R.....	69
6.19	Appendix S .....	70
6.20	Appendix T .....	71
6.21	Appendix U .....	72

## 2 Planning

### 2.1 Meeting minutes

#### 2.1.1 March 20, 2014

The meeting began with a retrospective of the third iteration. The burn down looked good, because of tasks being completed more consistently due to the use of sprints.

The group will try to complete the forth iteration early, since tasks can no longer be pushed back.

The medium AI was still too easy, and an AI glitch was still occurring near endgame.

Ten tasks remained open. The needs connectivity activity was pushed back to the forth iteration. Board rotation was determined to be low priority, and will only be done if time permits. The AI code needs to be reduced to pure Java code and made into a JAR library. The classes can then be extended in the Android code. Ben agreed to do this by the following Tuesday. The register player task was reopened as a subtask of the server API.

Further framework activities were then discussed. The database needs a few more hours of work put in. The client and server API need to be done, and depend on each other. The client side push message interface also needs to be completed.

Other tasks that need to be completed include adding the ability to modify a username, making the online list work and connecting to the server, creating a network player, and a network game activity which extends the needs network activity and polls on resume.

The most urgent tasks include the MySQL code, which should be done for Saturday the 22<sup>nd</sup>, the client and server side APIs, registering a user, and the inclusion of error codes.

Other tasks that need to be completed include the matchmaking timeout and the game inactivity timeout.

The room had been booked for the final iteration presentation.

The group then switched to the classroom to prepare for the presentation.

The group then presented to Vlad, showing a slideshow, the iteration 3 burn down, and a demonstration of the Android application. Vlad expressed concern over ending the game after the first player wins, but the group explained why this was the case. Once again, Vlad suggested changing the name of the Reset button to Undo. Vlad also expressed concern over changes to a username.

Afterward, the group discussed the suggestions made by Vlad, but decided not to alter their designs.

## 2.12 March 25, 2014

A recent commit by Peter caused an error in the application.

Saajid added sound and background music to a build on a separate branch. Volume sliders were added to the settings menu as well. The background music would play at the same time as other music playing on the device, but it was decided that other music should override music from the application. The Konami code Easter egg was also changed to alter the background music.

MySQL code for adding AI players to games was not working properly, so it will likely be handled by Java instead.

Cutting code for the AI JAR was not going well, so advice was given, and Peter was asked to look at it as well.

The trim to goal method in the ccGameBoard class was triggering an illegal move error.

The group also discussed how to handle a player placing a two-layer wall blocking a goal. It was decided a good fix would be to allow other players to swap their pieces in place of these, but only in this case.

## 2.13 April 1, 2014

The group still had thirteen topics open. These were discussed.

The hard level AI was working, but was not consistently able to beat the medium level.

Board rotation will not be completed for this iteration.

The notifications manager still needs to be done. The group also discussed the timeout for game creation and inactive players, testing, and the game state manager.

The JAR file for AI is made, but extending the objects for Android still needs to be completed.

For Thursday's demonstration, the group wishes to show Vlad two or more devices communicating over the Internet.

The binder for iteration four will include a list of API calls, music with references, and will finish with a comparison between the initial planning drawings and the finished product.

For networking, multiple tasks needed to be completed, including the client-side API, connecting the AI to the server, client-side push notifications, and connecting the online games list to the server.

The online game activity and network player still needed to be created, as well as the SQL creation script.

TH238 is booked for 6:30 on Thursday. The group should present to Vlad at 7:00.

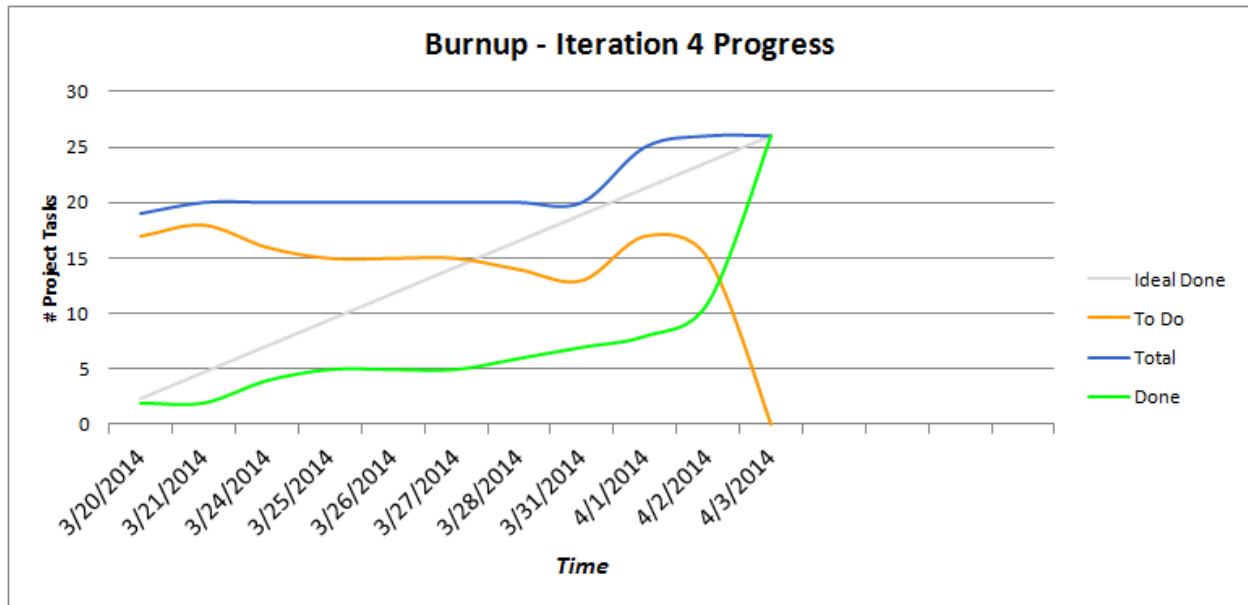
## 2.2 Project tasks

Task	Created	Status	Completed
#117 Construction - rotating the game board for offline play	3/3/2014	Cancelled	3/1/2014
#137 Needs connectivity activity	3/3/2014	Done	3/23/2014
#159 Register player	3/13/2014	Done	3/20/2014
#173 Fixing getPossibleMoves - not leaving goal	3/19/2014	Done	3/25/2014
#184 Medium AI Problem	3/20/2014	Done	3/28/2014
#191 Database - finish off remaining testing and refactoring	3/21/2014	Done	3/29/2014
#194 Construction - modify username	3/20/2014	Done	4/1/2014
#195 Construction - connect online list with the server	3/20/2014	Done	4/2/2014
#198 Resume dialog bug	3/20/2014	Done	3/22/2014
#209 Create test plan	4/2/2014	Done	4/2/2014
#210 Do Manual Testing	4/1/2014	Done	4/3/2014
#212 Create Ease of Use Tasks	4/1/2014	Done	4/2/2014
#106 Play against Hard Difficulty AI	3/3/2014	Done	4/3/2014
#156 Construction - Notifications Manager	3/13/2014	Done	4/3/2014
#158 Cut AI Code from Android	3/13/2014	Done	4/3/2014
#189 Construction - Create Server API	3/20/2014	Done	4/3/2014
#190 Construction - Client Side API	3/20/2014	Done	4/3/2014
#192 Construction - Server side hooking up AI	3/20/2014	Done	4/3/2014
#193 Construction - create client side push message interface	3/20/2014	Done	4/3/2014
#196 Construction - online game activity network calls	3/20/2014	Done	4/3/2014
#197 Construction - create network player	3/20/2014	Done	4/3/2014
#199 Construction - when players have been inactive replace them with AI	3/20/2014	Done	4/3/2014
#200 Construction - when players have been waiting for more than 1 minute fill the rest of the game with AIs	3/20/2014	Done	4/3/2014
#211 Do regression testing	4/1/2014	Done	4/3/2014
#213 Do Ease of Use Testing	4/1/2014	Done	4/3/2014
#214 Create presentation for final iteration	4/1/2014	Done	4/3/2014



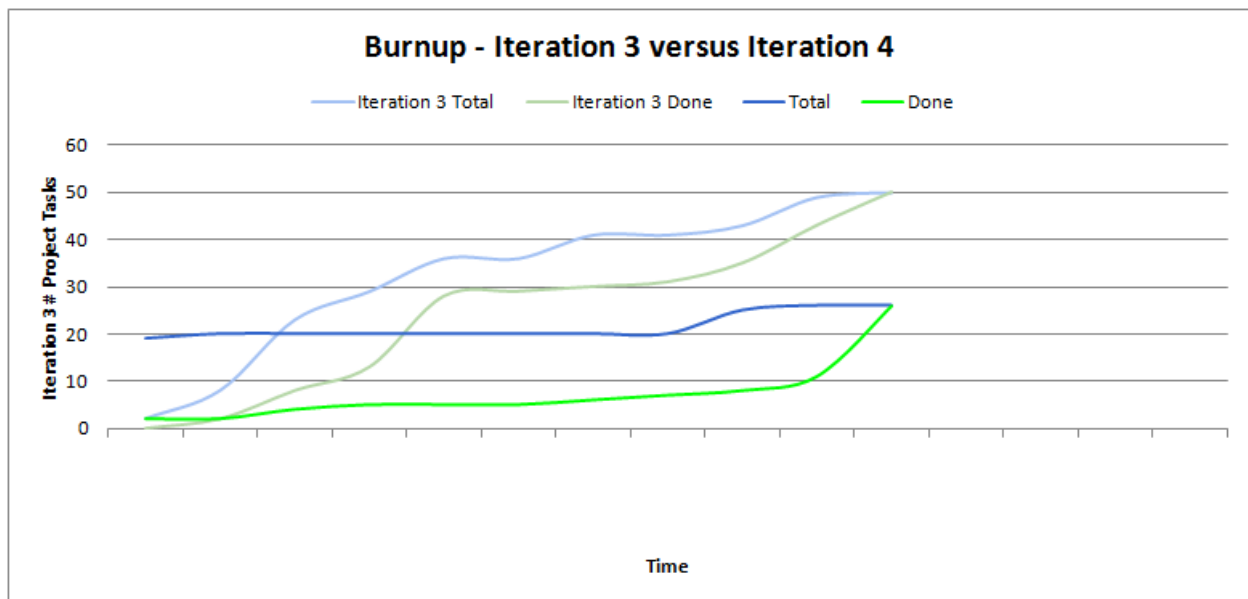
## 2.3 Burn downs and ups

### 2.31 Iteration four progress



2.1 Iteration four progress burn down and up chart


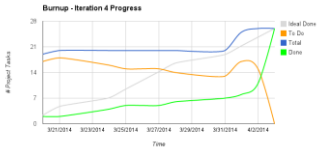
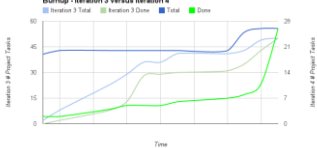
### 2.32 Iteration three versus iteration four



2.2 Iteration three versus iteration four burn up chart

## 2.4 Presentation

The following presentation will be presented during the retrospective meeting on April 3, 2014.  
Order follows: left to right, top to bottom.

<div><h3>Review</h3><p>Using Agile principles Split project into time-boxed iterations Created and prioritized the list of features Pulled highest priority features into the current iteration</p></div>	<div><h3>Project Timeline</h3></div>
<div><h3>Iteration #4 Goals</h3><p>Pulled the next set of features from the backlog Implementation of the hardest level of artificial intelligence Sound Online play</p></div>	<div><h3>Planning</h3><p>Final iteration was limited to 2 weeks All tasks are tracked in GitHub Analysis, design, construction and testing spawned many subtasks eventually leading to a total of 26 project tasks</p></div>
<div></div>	<div></div>
<div><h3>Observations</h3><p>The shorter duration was challenging Several relatively complex components (web server, database, integration with Google services) took more time than predicted</p></div>	<div><h3>Continuing Other Iteration Activities</h3><p>GitHub used to house all artifacts, source code and automated tests Analysis and design artifacts followed conventions from previous iterations Construction and testing was done using AndroidStudio travis-ci.org used for continuous integration</p></div>
<div><h3>Reflection</h3><p>Agile Principle #12: “At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.” Source: <a href="http://agilemanifesto.org/principles.html">agilemanifesto.org/principles.html</a></p></div>	<div><h3>Demonstration</h3></div>
<div><h3>Feedback?</h3></div>	<div><h3>Thank you</h3></div>

## 3 Design

### 3.1 Artificial intelligence (AI)

The original plan for the artificial players that a user may play against, or pit up against each other to help train players or learn strategies from, has been modified during the evolution of the application. The three levels of difficulty originally established were to be labeled as Easy, Medium, and Hard to help users know exactly what they were picking without the requirement of outside assistance. They were to have clearly distinct levels of difficulty to beat, and work in a time effective manner.

#### 3.11 Easy

The easy difficulty was planned to use a greedy algorithm to strictly take the largest possible move towards the goal. After implementing this it was found that not a single new user was able to beat the artificial player and during a meeting it was requested to be made easier. Since then the greedy algorithm has remained the same, but a restriction on the player has been put into place. Each turn the AI will generate a number between two and six and will make that the largest number of moves available that turn. For example if it generated three, then it may slide/hop, and then hop up to two additional times that turn during its final chosen move.

#### 3.12 Medium

The medium difficulty was to add a min/maxing algorithm to make moves that don't help the opponents more preferred. Also to add a preference to stay in the middle of the board, back piece weight, and split. After making the easy AI easier, along with our modification of the rules to make the game faster it was found that utilizing the greedy AI without the restrictions and adding a preference to staying centered on the path towards the goal was the most effective 'medium' difficulty. Many different difficulties were created and pitted against each other, and this has seemed to fit our purposes the best; being able to beat the easy AI consistently, while not being impossible for users to beat.

## 3.13 Hard

The plan surrounding the hard difficulty was originally to take the Medium AI and extend it to thinking a move in advance. This has been completed successfully and, while it still takes a moment to process even with pruning implemented, has been beating the medium difficulty computer players consistently. Creating these computer players has been quite exciting, and I look forward to experimenting with and perfecting them over the next few months.

## 3.2 Database

### 3.21 Control flow

- See Appendix G

### 3.22 MySQL stored procedures

Note that outputs capitalized represent literal column names of outputted tables.

Procedure name	Inputs	Outputs	Notes
checkWinner	USERID, GAMEID	USERNAME – filled with a name if a winner if found, null otherwise	Updates winner automatically on GAMES table if a winner is found.
createGame	NUMBEROFPLAYERS	GAMEID	Should not have to be called explicitly. Only to force a new game creation.
getCreationTimes	N/A	GAMEID, CREATED – filled with times of games created that are not ready.	DATETIME formatted as YYYY-MM-DD HH:MM:SS
fillGame	GAMEID	USERID, USERNAME, ISAI	Fills the game with AI users and returns the list of players for that game. NOTE: It will create one more AI than necessary and return a check constraint error. This is due to some issue with how LOOPS handle incoming variables.
getWinner	GAMEID	USERNAME – filled with a name if a	N/A

		winner has been declared, null otherwise	
matchMake	USERID, NUMBEROFPLAYERS	GAMEID	Will create a game if no suitable game is found.
nextTurn	GAMEID	CURRENTTURN – returns the next iterated turn for that game	N/A
getGameState	GAMEID	USERID, USERNAME, PLAYERNUMBER, ISAI, CURRENTTURN, WINNERID of users within a specific game	Last two rows will be repeated for each player row. Just a small side effect.
getUserGames	USERID	GAMEID, NUMPLAYER, ISREADY, CURRENTTURN, PLAYERNUMBER, WINNERID	Just note that numPlayer is the number of players and playerNumber is the number of the user within that game. <i>Note: a subsequent call will need to be made to get usernames of winnerIDs.</i>
userJoinsGame	USERID, GAMEID	N/A	Used by matchMake, shouldn't have to be used unless for some reason you need to force a user into a game.
userLeavesGame	USERID, GAMEID	N/A	Removes the user from the game, replaces them with an AI user. Username of player within the game is the same. Will also check if all users are AIs, if so it will delete the game and by extension the pieces and all associated AI users.
getGamePieces	GAMEID	PLAYERNUMBER, ONROW, ONINDEX	N/A
populateBoard	GAMEID, PLAYERNUMBER	N/A	Called by matchMake, should not have to be used explicitly
updatePiece	GAMEID, OLDROW, OLDINDEX, NEWROW, NEWINDEX	N/A	Updates a piece in the database.
createUser	USERNAME (max 50 characters)	USERID	Creates a new user.
createAI	NONE	USERID as aid	Used by userLeavesGame, deleteUser, should not have to be called explicitly.
deleteUser	USERID	N/A	Replaces user with one AI, updates relation table. If called

			on an AI user it will pass an error.
getUsername	USERID	USERNAME	N/A
setUsername	USERID, NEWUSERNAME (max 50 characters)	N/A	Will return an error if that username is already taken because of unique restriction on username.

## 3.3 Network API

### 3.31 User

#### Register a user

- URL
  - POST /users/register/:username
- Request Object
  - Name: RegisterUserRequest
  - Fields: username
- Response Object
  - Name: UserRegistrationReceivable
  - Fields: username, userId

#### Change a username

- URL
  - POST /users/change/:userId/:username
- Requested Object
  - Name: ChangeUsernameRequest
  - Fields: userId, username
- Response Object
  - Name: SuccessReceivable
  - Fields: code, message

## 3.32 Game

### Join a game

- URL
  - POST /games/join/:playerCount/:userId
- Requested Object
  - Name: JoinGameRequest
  - Fields: playerCount, userId
- Response Object
  - Name: JoinGameReceivable
  - Fields: gameListItem (gameId, isReady, currentPlayerNumber, playerNumber, winnerUsername, winnerNumber, numberOfPlayers)

### Delete a game

- URL
  - POST /games/delete/:gameId/:userId
- Requested Object
  - Name: DeleteGameRequest
  - Fields: gameId, userId
- Response Object
  - Name: SuccessReceivable
  - Fields: code, message

### Get a game state

- URL
  - GET /games/gamestate/:gameId
- Requested Object
  - Name: GameStateRequest
  - Fields: gameId
- Response Object
  - Name: GameStateReceivable
  - Fields: gameId, currentPlayerNumber, winnerUserId, isReady, numPlayers, pieces[] (playerNumber, row, index), players[] (userId, username, playerNumber), gameState (currentPlayerNumber, winnerUserId, isReady, numPlayers)

## Get a list of games

- URL
  - GET /games/list/:userId
- Requested Object
  - Name: GameListRequest
  - Fields: userId
- Response Object
  - Name: GameListReceivable
  - Fields: gameListItems[] (gameId, isReady, currentPlayerNumber, playerNumber, winnerUsername, winnerNumber, numberOfPlayers)



## 3.4 Visual

### 3.41 Introduction

Iteration four is the last planned iteration for this application and, as such, the visual design of the application in its current state is the final design implementation until future development. The images in this section are screenshots from the application taken on mobile device running Android. Insignificant variations will be observed in different devices due to manufacture modifications. Design concepts have been utilized in an attempt to minimize display variations.

### 3.42 Idea

Before any digital mock ups were created, the following wireframe sketches were produced. These sketches jumpstarted the design process and all future mock ups stem from them. As a team, we feel that it is important to include these initial drawings in the documentation to show how far we have come together: from pieces of paper to a full-blown game.

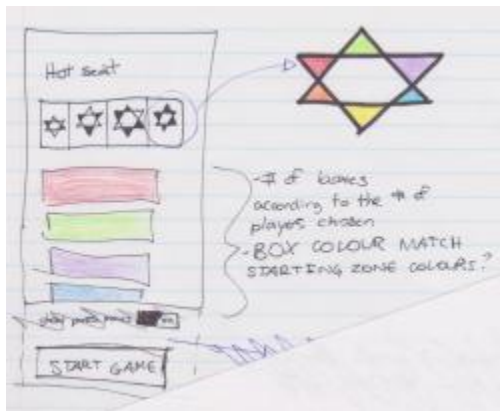


Figure 3.1 "Hot seat" configuration activity

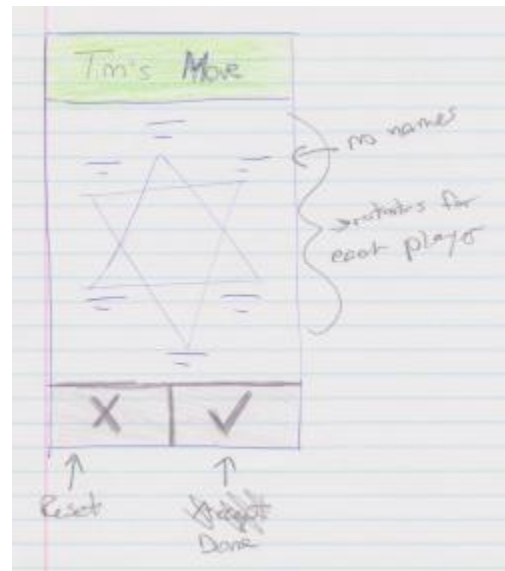


Figure 3.2 Game board activity

### 3.43 Screenshots

All activities of the application have been captured and can be viewed in Appendix F.

## 4 Testing

### 4.1 Iteration four testing plan

#### 4.11 Revision history

Date	Author	Description of change
2014/03/31	Saajid Mohammed	Authored testing plan
2014/03/01	Taylor Whited	General revision and editing
2014/03/02	Saajid Mohammed	Minor corrections

#### 4.12 Introduction

The program to be tested is a full functioning offline and online version of a Chinese Checkers game for Android, with the capability of playing against artificial intelligence. It contains six activities. The first activity has a buttons which lead to either the offline configuration, online configuration, settings or help. The offline configuration activity is a configuration screen for the game you wish to play. It contains the options to set the number of players, their names and whether or not a certain players are computer opponent. It then launches the game activity that contains a canvas with the game board and buttons to confirm the player's move and undo a move. The online list activity contains any current games you are involved in; it also allows you to create new online games. The other activities are a help activity and settings activity. The help menu contains FAQ's and a basic tutorial for setting up a game. The settings activity contains a toggle for showing hints; it also contains the user information and setup to allow the user to play online. Apart from the activities the application will also have the ability to notify users, when the application is no longer in focus, that that there is has been changes in their current online games.

#### 4.13 Objectives

Verify the current phase programming is functional and works consistently.

## 4.14 Testing phases

### UI Automated Unit testing

N/A

### UI Automated Integration testing

N/A

### Manual testing

- OnlineListActivity
  - Verify game list is populated appropriately
  - Verify games can be joined, deleted and forfeited
- Verify online game play is complaint with the rules of Chinese Checkers as stated in Game rules (reference iteration three)
- Verify online matchmaking
- Verify push notifications are displayed appropriately
- Verify push notifications take player to proper game
- Verify AI players are obeying game rules and functions with noticeable difference in skill levels.

### Performance and stress testing

N/A

### Regression testing

- Verify previous functionality is still intact
- Verify previous phase bugs are no longer present

### Ease of use testing

A focus group shall be assembled and given access to the functioning application they will also be given a set of tasks to accomplish. Each member of the focus group shall then fill out an Ease of Use Testing Form (Appendix B). The forms will then be tallied and actions shall be taken in accordance with the results at the retrospective meetings.

## Acceptance testing

Criteria and this phase testing to be determined by team lead.

### 4.15 Testing feedback procedure

At the end of each test phase or immediately following a failure of an in phase test, a Test Feedback Form (Appendix A) shall be generated. The feedback form will be generated as an issue on GitHub with the contents of the (Appendix A). Further action shall be determined on case bases.

### 4.16 Features to be tested

- OnlineListActivity
  - Button functionality
  - Player lobby tracking
  - Transition into proper OnlineGameActivity with updated state
  - Transition to HelpActivity
  - Transition to SettingsActivity
- OnlineGameActivity
  - Button functionality
  - Game playability
  - Popup Dialogs
  - Transition to HelpActivity
  - Transition to SettingsActivity
- Game adheres to rules as stated in Game rules (reference iteration three)
- Push Notifications
- Online Matchmaking
- AI Opponents

### 4.17 Features not to be tested

N/A

### 4.18 Dependencies

Availability of classes and modules for current phase.

## 4.19 Tools

- Eclipse
- JUnit API
- Android Testing API

## 4.110 Approvals

Name	Project Role	Signature	Date
Curtis Smith	Project Lead	_____	_____
Peter Pobjewski	Deputy Lead	_____	_____
Ben Stitt	Documentation Lead	_____	_____
Saajid Mohammed	Test Lead	_____	_____

## 4.2 Manual testing

- See Appendix C

## 4.3 Regression testing

### 4.31 Feedback

- See Appendix D

### 4.32 Main tests

- See Appendix H

#### GridUnitTest

- See Appendix I

#### ActivityAtTest

- See Appendix J

#### MainActivityUnitTest

- See Appendix K

#### MainAndConfigIntegrationTest

- See Appendix L

#### MainAndConfigAndGameIntegrationTest

- See Appendix M

## OfflineGameActivityUnitTest

- See Appendix N

## OfflineConfigurationActivityUnitTest

- See Appendix O

## 4.33 UI engine tests

- See Appendix P

## DimensionsTest

- See Appendix Q

## PixelPositionTest

- See Appendix R

## PiecePositionSystemTest

- See Appendix S

## PieceDrawingDetailsTest

- See Appendix T

## 4.34 Tests summary

- See Appendix U

## 4.4 Ease of use testing

### 4.41 Tasks

- Task 1: Start a 2 player online game.
- Task 2: Start a 4 player online game.
- Task 3: Enter the 2 player game, play a couple of turns.
- Task 4: Exit the 2 player game and delete it from your list.
- Task 5: Start a 6 player game and remove it from your list before it starts.
- Task 6: Rate your overall enjoyment of the app and included comments of what you would like to see in future versions of the app.
- Task 7: Crank the volume way up and enjoy the music. On behalf of the team I would like to thank everyone for the input and support you have provided.

### 4.42 Results

- See Appendix E



## 4.5 Acceptance testing

### 4.51 Introduction

This document outlines the methods and testing procedures for all acceptance testing for the Chinese Checkers application designed and developed by the COSC 3F00 team known as Logical Operators. Each test has two possible outcomes: pass or fail. Any failed test is reason for the entire test sequence to be failed.

### 4.52 Project description

We are creating a version of Chinese Checkers. Our target audience will be users that are 6 years of age and older. The game will be played with both human and computer players. Our target platform will be Android phones (original launch on OS version 4.4, API version level 19).

We will be developing the solution using Android Studio using GitHub as our source code repository.

Graphics will be created using the Android canvas.

Networking will be accomplished using the internet via HTTP and centralized on a web server using the Play framework. Heavy processing for the AI will be executed on the web server so as to limit the processing power needed on the user's device. JSON will be the preferred method to exchange data between the Play server and Android clients.

The database solution will be MySQL.

We will implement the game as a "push game" -- where users are notified when it is their turn. This will add the option of playing a game over multiple sittings, while maintaining the ability to a play fast-paced game.

All technical documents and project tracking will be available from the GitHub wiki and service hooks available from this tool.

### 4.53 Test team personnel

The test team consists of one tester and one primary customer witness who have the authority to sign off tests. Optionally, a small, agreed number of additional customer observers can observe the tests and input their observations to the primary witnesses.

Name	Role	Team
Curtis Smith	Team Leader/Tester	COSC 3F00 Logical Operators
Saajid Mohammed	Test Lead	COSC 3F00 Logical Operators

## 4.54 Acceptance test plan

### Structure

Each test process involves COSC 3F00 Logical Operators personnel running the test procedures described in this document to the satisfaction of the team.

### Order of tests

The tests in the following section are listed in the same order as they should be performed during acceptance testing. This is to help give a logical flow of work on the system. The order of the tests is designed to minimize the use of the same elements of the system at the same time.

### Software testing

Each software component provided by Logical Operators must be tested and perform to the expected standards outlined in the table below. It is expected that several bugs or unexpected behaviours may occur during testing. Any bug, unexpected behaviour, or missing functionality will be documented as a deficiency. Each software deficiency must be resolved and tested prior to sign-off unless agreed upon in writing by both parties.

#	Test Description	Expected Result	Pass	Fail
1	Tap the Settings button, and then tap the username text entry box. Delete the existing text and enter a new username. Tap the back button, and then tap the Settings button again.	The username entered appears in the username text entry box.	X	
2	Tap the Play Online button.	A list of online games in which the user is participating appears.	X	
3	After completing task 2, tap the New Game button, tap the desired number of players, and then tap the Join button. Repeat for each number of players.	The newly created game appears on the list of online games.	X	
4	After completing task 3, press and hold one of the newly created games on the online games list. Tap the forfeit	The game is removed from the list of active	X	

	button on the prompt window.	games.		
5	After completing task 3, tap one of the games.	The current state of the board is displayed.	X	
6	After completing task 5, tap the name display at the top of the screen.	A list of players is displayed on an overlay over the screen.	X	
7	After completing task 3, tap one of the games in which it is not your turn.	Making a move is not allowed.	X	
8	After completing task 3, tap one of the games in which it is your turn.	Making a move is allowed.	X	
9	Repeat task 5 until the game is completed, and a winner is announced. Return to the online games list. Press and hold the game which has been completed. Tap the Delete button on the prompt window.	The game is removed from the list of active games.	X	

## 4.55 Notes

None

# 5 Reflection

## 5.1 Introduction

The goal of the project was to use agile software engineering practices to implement a mobile application version of Chinese Checkers that could be deployed to and used on an Android phone. Success can be measured by the artifacts created that support project planning, analysis, design and testing as well as by the final working software product. The main purpose of the COSC 3F00 course was to learn software engineering practices and the software product of this project was actually secondary to the lessons that were learned. This project summary will focus on some key lessons discovered in the areas of communication, planning, analysis, design, development and testing.

## 5.2 Communication

Face to face communication is the most effective method of communication. The team worked hard to meet twice each week for a period of two hours and sometimes more. However, the first problem the team faced was the fact that there was only limited time that everyone could meet together as a group to work on the project. In order to keep information flowing between group meetings the team used several mechanisms. Email alone had drawbacks. Fortunately other social media solutions were available such as Facebook and GitHub. The team took advantage of both of these options. GitHub also had the added benefit of some lightweight project management capabilities such as project task creation, task assignment, bug tracking, discussions and milestones. It also had configuration management capabilities and was used for the team's software version control. A combination of social media and face to face communication was used with great success and were equally essential in making the progress that was observed.

## 5.3 Planning

An Agile software development approach was chosen because of its flexibility and responsiveness to change. During the initial planning phase of the project it was determined that there were many unknowns and risks so the scope needed to be flexible and there needed to be focus on quick, iterative delivery and feedback. Agile was the natural choice.

An estimated work effort capacity was calculated based upon team members' availability for the project effort. This was used as a guide when it came to making decisions about how much work was to be accomplished with each iteration. Based on this, decisions were made during each planning meeting about what could be done with the next iteration.

After each iteration a retrospective conversation took place where suggestions were raised and decisions were made about how to modify the process.

The project work was assigned and tracked using project tasks that were loaded to GitHub. Agile "Burn down" charts were created to visualize the progress of the work through the iteration.

The project effort consisted of four iterations of varying duration. This was done intentionally during the initial planning stage. The first iteration was four weeks and was the longest of the iterations. This was done to accommodate for the expected ramp up time needed to get the tools the process foundations in place. The next two iterations were three weeks in length. These iterations were intended to contain the bulk of the development. The final iteration was two weeks in length and was intended to be used to implement any final missing features and eliminate any outstanding defects.

As was previously mentioned during the initial planning stage of the project a work effort capacity was calculated and was to be 528 total hours. The expectation was the each person would commit 6 hours per week to the project, multiplied by 8 people over the course of 11 weeks. However, the observations of the team over the past 11 weeks show that this was drastically underestimated. A more conservative estimate would be that the effort was at least twice the original prediction. Less time could have been spent but the results would have been a product with less functionality.

## 5.4 Analysis, Design, Development and Testing

The use of conventional artifacts such as UI mock-ups, data flow diagrams, class diagrams, use cases and test plans all helped to guide the development and testing of the product. Other Agile concepts such as test automation and continuous integration were used to augment the construction process.

The Android Studio tool used for development included a framework for test automation. This was leveraged to build out a suite of functional and unit tests that could be run at the click of a button or by a script.

The use of an online continuous integration service called Travis ([travis-ci.org](https://travis-ci.org)) was used to automatically checkout the latest changes introduced into the GitHub source code repository, build the software executables and run the suite of automated tests whenever a developer on the team committed a change to the repository. If the Travis process failed due to a compilation error or a failed test then an email would be sent to the individual that committed the last change which introduced the problem and it would need to be fixed immediately. This gave confidence to the team that there was always a working build in the repository.

As previously mentioned, Android Studio was the development tool of choice. It afforded the team with the ability to seamlessly develop on multiple platforms. Both Windows OS and Mac OS devices were used to develop and integrate this product.

For the testing activities, both manual and automated testing was done successfully on multiple Android devices (phones, tablets, emulators).

The combination of conventional and agile development tools and activities were essential in accomplishing the goals of the project.

## 5.5 Conclusion

In order to create reliable, quality software proven software engineering practices are required. Complexity increases exponentially when working in teams and success with teams requires special commitment to planning and communication in addition to the basics of analysis, design, development and testing. Teams with individuals of varying skill sets are necessary for large software development efforts and there are many lessons to be learned and applied from this experience in COSC 3F00.

## 6 Appendices

## 6.1 Appendix A

### Test feedback form

Project: \_\_\_\_\_

Project Phase: \_\_\_\_\_ Test Phase: \_\_\_\_\_ Date: \_\_\_\_\_

Tester: \_\_\_\_\_

Pass\Fail\Extra Consideration Required: \_\_\_\_\_

Remarks (If fail what caused failure): \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Test Output (If Applicable): \_\_\_\_\_

Action Taken: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Tester Signature: \_\_\_\_\_ Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_ Project Lead Name: \_\_\_\_\_

Test Lead Name: \_\_\_\_\_



## 6.2 Appendix B

### Ease of use testing feedback form

Give each of the tasks a rating from 0 to 10 based on the difficulty of accomplishing the relevant tasks.

Task 1 Score: \_\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 2 Score: \_\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 3 Score: \_\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 4 Score: \_\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 5 Score: \_\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 6 Score: \_\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 7 Score: \_\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

General Remarks: \_\_\_\_\_

\_\_\_\_\_

## 6.3 Appendix C

### Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4      Test Phase: Manual Testing - Push Notifications  
04/02/14

Date:

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure):

- Push notifications displayed appropriately - Pass
- Push notifications takes player to readied game - Pass

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4      Test Phase: Manual Testing - OnlineListActivity  
04/02/14

Date:

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure):

- Game list populated appropriately - Pass
- Games can be joined, deleted and forfeited - Pass

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4      Test Phase: Manual Testing - Online Matchmaking      Date:  
04/02/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure): N/A

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4      Test Phase: Manual Testing - Online Game Mechanics      Date:  
04/02/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure): N/A

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4      Test Phase: Manual Testing - Hard AI      Date: 04/02/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure): N/A

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4    Test Phase: Manual Testing - Difference In AI Difficulties    Date:  
04/02/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Extra Consideration Needed

Remarks (If fail what caused failure):

- Noticeable difference in Easy and Medium - Pass
- Noticeable difference in Medium and Hard - Extra Consideration Needed
  - Hard is sometimes easier than medium

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4

Test Phase: Manual Testing - Difference In AI Difficulties Retest

Date: 04/03/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure):

- Noticeable difference in Easy and Medium - Pass
- Noticeable difference in Medium and Hard - Pass

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed



## 6.4 Appendix D

### Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4      Test Phase: Regression Testing - Manual

Date: 04/02/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure):

- Easy AI - Pass
- Medium AI - Pass
- Difference in AI skill - Easy and Medium - Pass
- Game Mechanics - Pass
- Dialogs - Pass

Test Output (If Applicable): N/A

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## Test feedback form

Project: Chinese Checkers for Android

Project Phase: 4      Test Phase: Regression Testing - Automated UI      Date: 04/02/14

Tester:

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure): N/A

Test Output (If Applicable): Attached in Automated Testing Printouts

Action Taken: N/A

Tester Signature: \_\_\_\_\_

Project Lead Signature: \_\_\_\_\_

Test Lead Signature: \_\_\_\_\_

Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

## 6.5 Appendix E

### Ease of use testing feedback form

Give each of the tasks a rating from 0 to 10 based on the difficulty of accomplishing the relevant tasks.

Task 1 Score: \_2\_\_\_\_

Task Remarks: \_Easy. \_\_\_\_\_

\_\_\_\_\_

Task 2 Score: \_2\_\_\_\_

Task Remarks: \_Also easy. \_\_\_\_\_

\_\_\_\_\_

Task 3 Score: \_2\_\_\_\_

Task Remarks: \_I like that it looks the same as the offline game.

\_\_\_\_\_

\_\_\_\_\_

Task 4 Score: \_2\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 5 Score: \_2\_\_\_\_

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 6 Score: \_8\_\_\_\_

Task Remarks: \_When I win I think it would be cool if there were some effects as well as the message that appears.

\_\_\_\_\_

Task 7 Score: \_10\_\_\_\_

Task Remarks: \_I like the music. It has a fun melody.

\_\_\_\_\_

General Remarks: \_\_\_\_\_

\_\_\_\_\_

### Ease of use testing feedback form

Give each of the tasks a rating from 0 to 10 based on the difficulty of accomplishing the relevant tasks.

Task 1 Score: 2

Task Remarks: Easy.

\_\_\_\_\_

Task 2 Score: 2

Task Remarks: Also easy.

\_\_\_\_\_

Task 3 Score: 2

Task Remarks: This is the same as the other kind of game (offline).

\_\_\_\_\_

\_\_\_\_\_

Task 4 Score: 2

Task Remarks: Just a couple of touches to the screen is not hard to do.

\_\_\_\_\_

\_\_\_\_\_

Task 5 Score: 2

Task Remarks: \_\_\_\_\_

\_\_\_\_\_

Task 6 Score: 7

Task Remarks: It's kind of fun but it would be cool if you could get points when you win that you could use to buy stuff at store.

\_\_\_\_\_

\_\_\_\_\_

Task 7 Score: 10

Task Remarks: The music is fast and fun.

\_\_\_\_\_

\_\_\_\_\_

General Remarks: \_\_\_\_\_

\_\_\_\_\_

Ease of use testing feedback form

Give each of the tasks a rating from 0 to 10 based on the difficulty of accomplishing the relevant tasks.

Task 1 Score: \_0\_\_\_\_

Task Remarks: \_\_\_\_\_  
\_\_\_\_\_

Task 2 Score: \_0\_\_\_\_

Task Remarks: \_\_\_\_\_  
\_\_\_\_\_

Task 3 Score: \_1\_\_\_\_

Task Remarks: \_Feels like the same game as offline play (good thing).\_\_\_\_\_  
\_\_\_\_\_

Task 4 Score: \_3\_\_\_\_

Task Remarks: \_Was not familiar with Android devices and did not realize long-presses were a thing. Once learned, it was easy to do.\_\_\_\_\_  
\_\_\_\_\_

Task 5 Score: \_1\_\_\_\_

Task Remarks: \_Once long-presses were learned, this was a breeze.\_\_\_\_\_  
\_\_\_\_\_

Task 6 Score: \_10\_\_\_\_

Task Remarks: \_I really enjoyed how relaxing the game was to play. Enjoyed the lack of immediate competition.\_\_\_\_\_  
\_\_\_\_\_

Task 7 Score: \_1\_\_\_\_

Task Remarks: \_I prefer to play games without music, but the audio sounded fine and the volume controls were easy to use.\_\_\_\_\_  
\_\_\_\_\_

General Remarks: \_\_\_\_\_  
\_\_\_\_\_

## 6.6 Appendix F

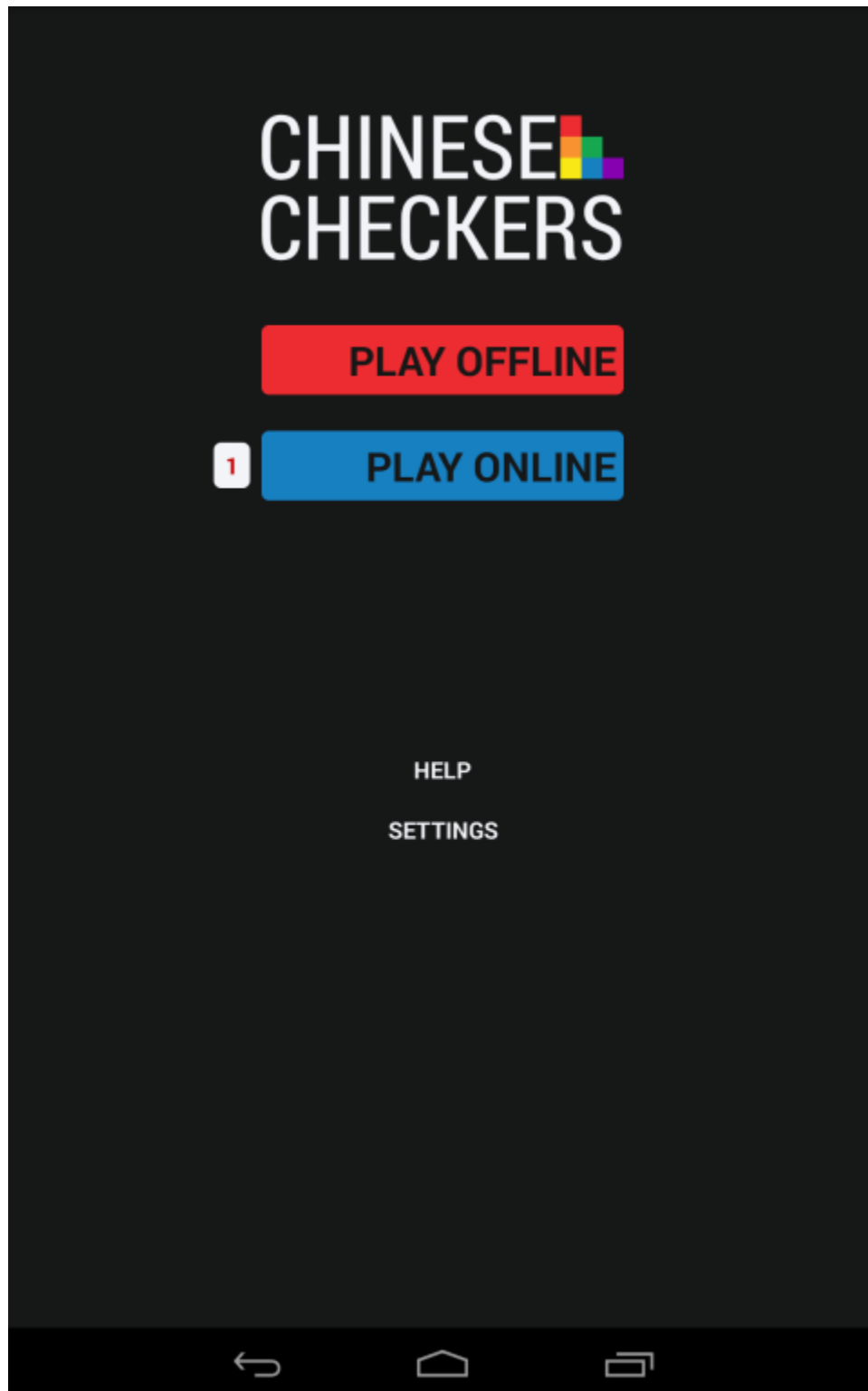


Figure 6.1 Main activity

### How do I start a game by myself or with friends?

Tap the Offline button on the main menu. This will bring you to a second screen where you can set up your game. Four buttons at the top of the screen allow you to change the number of players, including yourself and computer controlled players, or AI. By default, only the first player is set to human. You can enter your name in the text box, and tap the buttons on the others to change the AI difficulty. You can change the AI players to humans by tapping the picture of a robot to the right of each set of difficulty buttons. Tap Start Game to begin. At the end of each turn, pass the game to the next player.

### How do I play?

Once you have a game started, tap one of your game pieces. The spaces you can move it to will be highlighted. Tap one of these spaces to move your piece. If the piece can be used to hop another, it may do so multiple times, even if you started by moving it one space. You can click Reset to move it back, or tap Done to end your turn. If you tap Done without moving a piece, you will skip your turn. To win, move all your pieces into the opposite corner before your opponents do the same. If an opponent is blocking any of your goal spaces, filling the rest will count as a win.

### Can I watch a game to see how it's played?

Yes. Just tap the Offline button, tap the first player's human icon to change it to an AI, and then tap Start Game. Watch and learn.

### Why do I have a username?

This name is used for online games. This is the name other users will see when you play online. You can change it anytime your device is connected to the Internet by entering a new one in the settings menu.

### How do I start a game over the Internet?

Tap the Online button on the main menu. This will bring you to a second screen that lists your online games in progress. To start a new one, click the New Game button at the bottom of the screen. You can then select the number of players by tapping one of the buttons that appear. After a short



Figure 6.2 Help activity

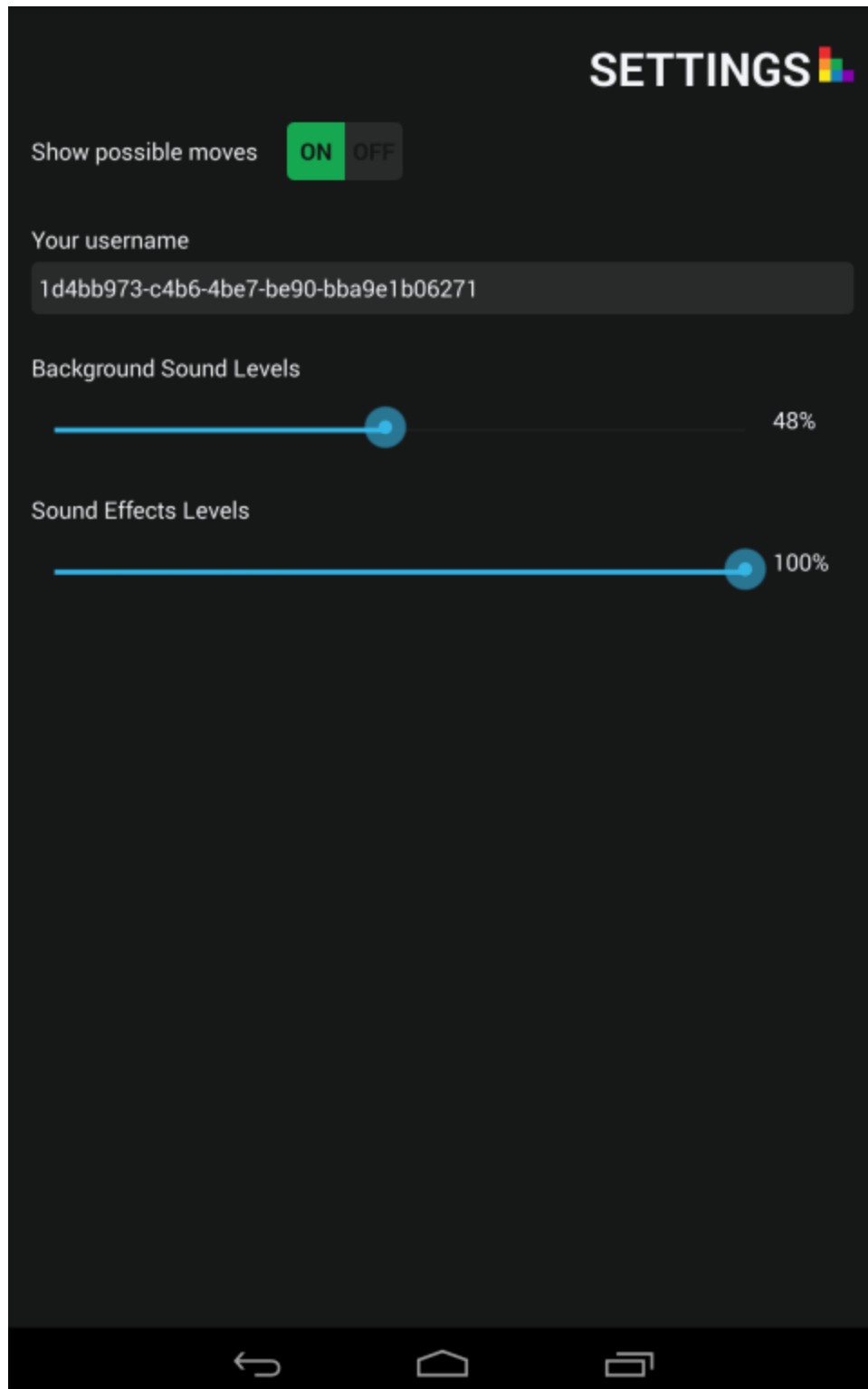


Figure 6.3 Settings activity



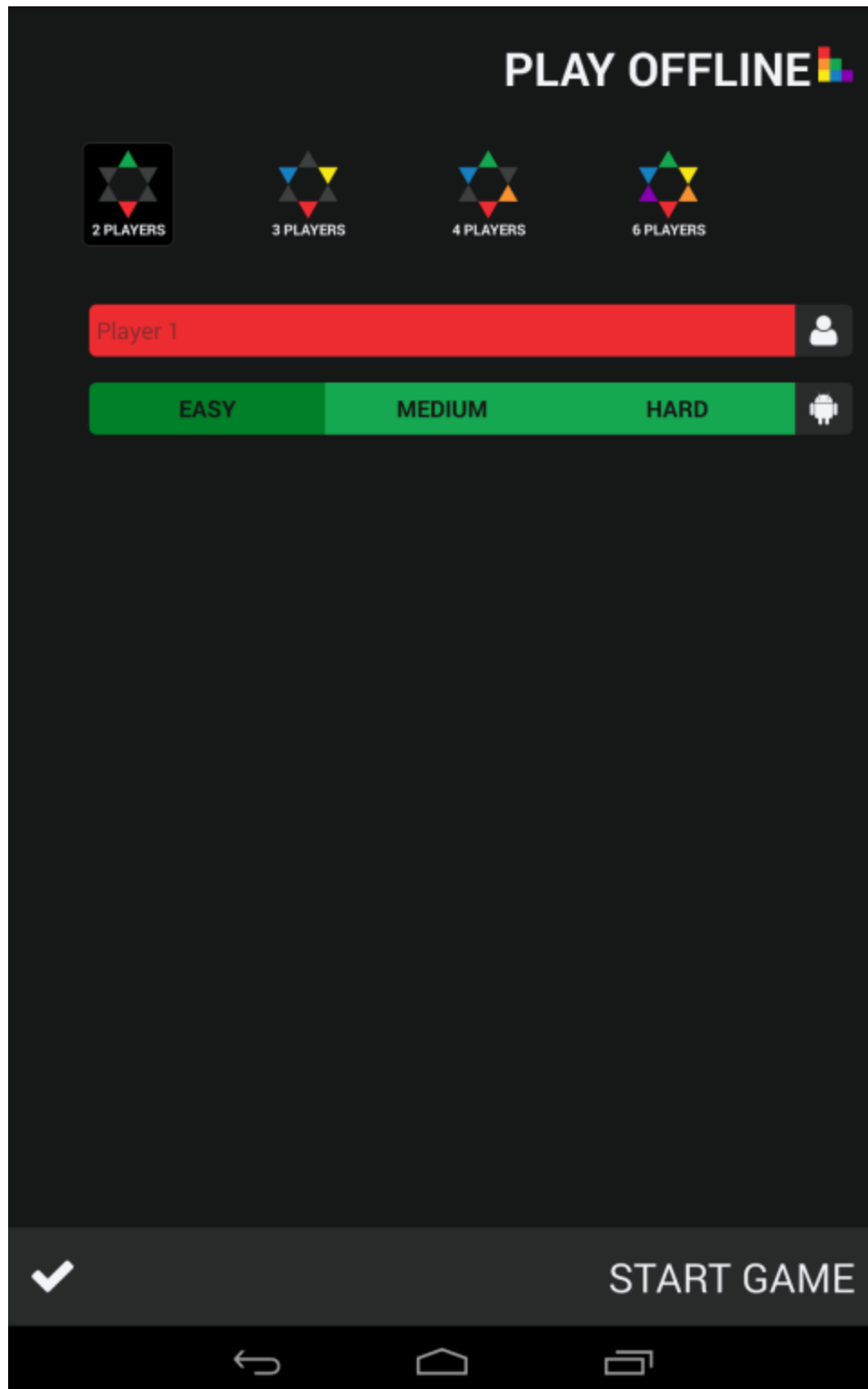


Figure 6.4 Play offline configuration activity: 2 players

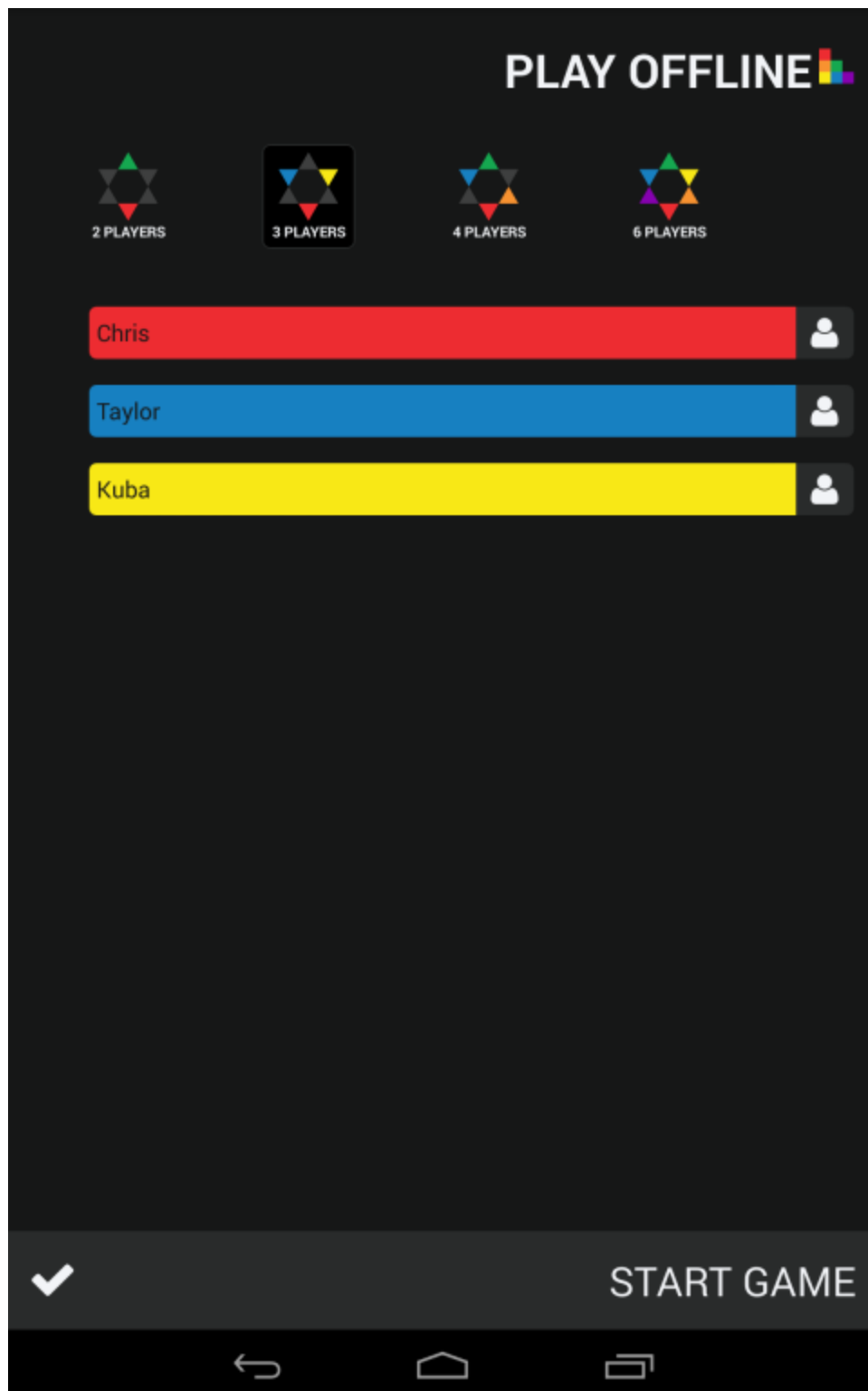


Figure 6.5 Play offline configuration activity: 3 players with names

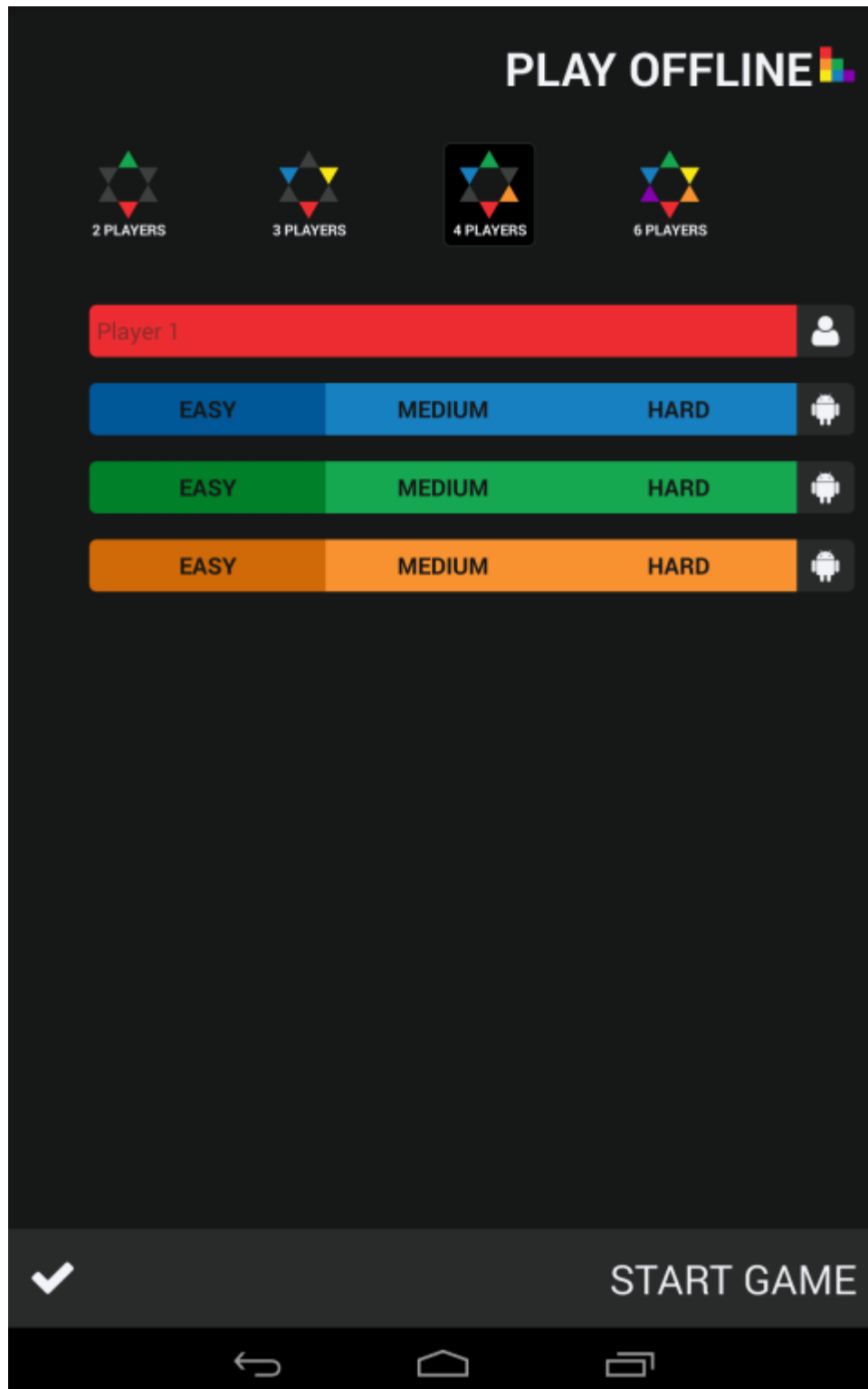


Figure 6.6 Play offline configuration activity: 4 players

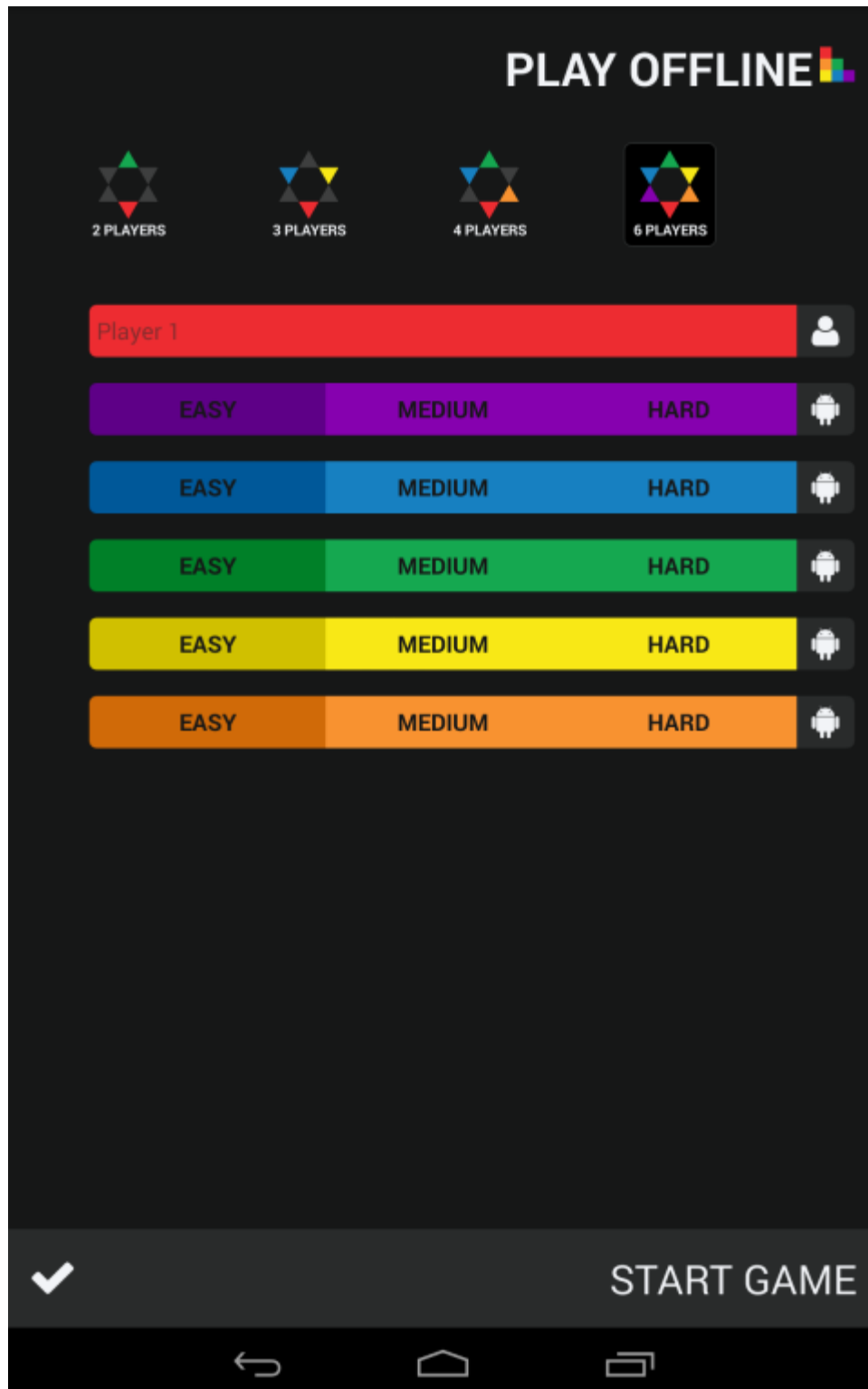


Figure 6.7 Play offline configuration activity: 6 players

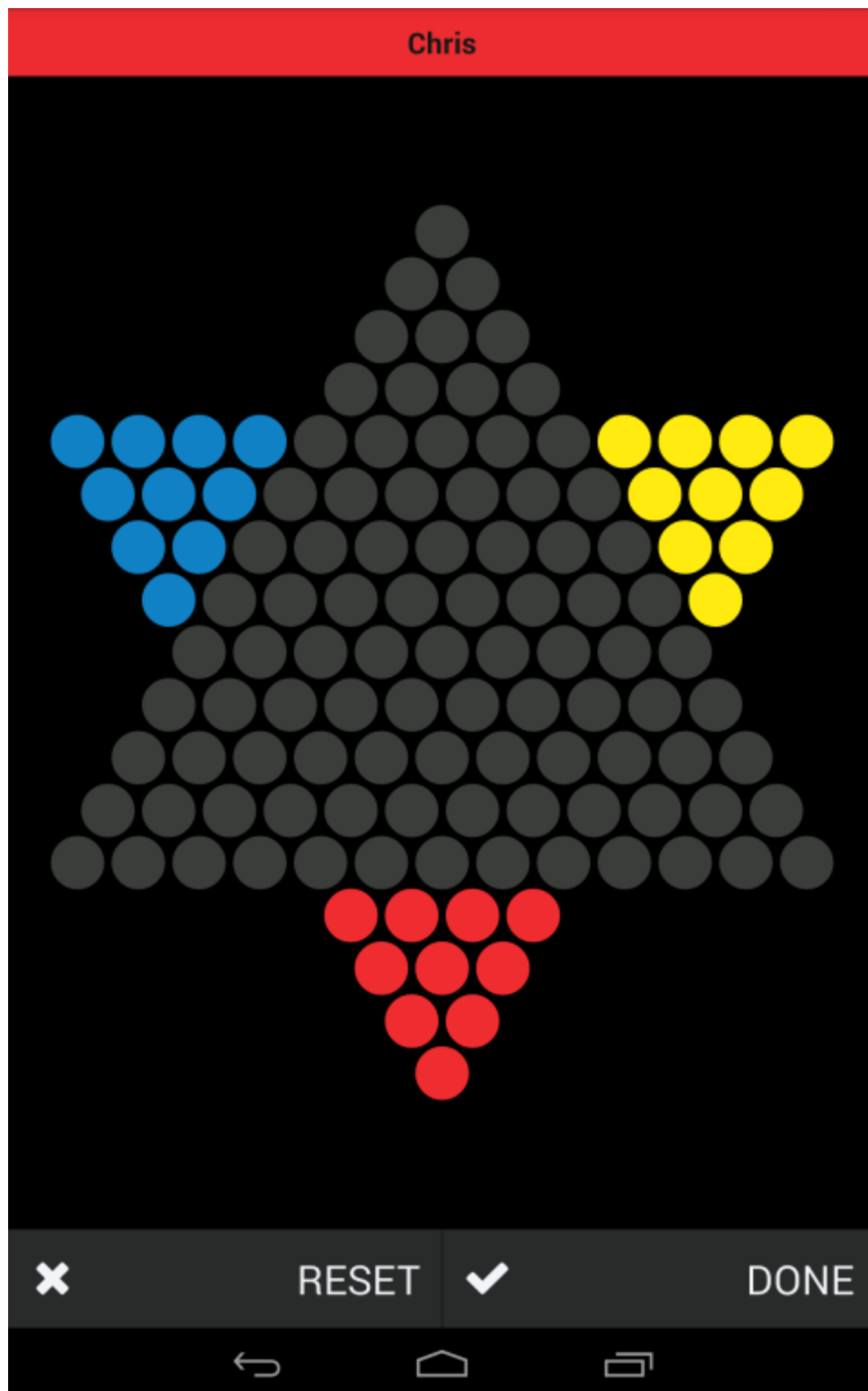


Figure 6.8 Start of a three player game

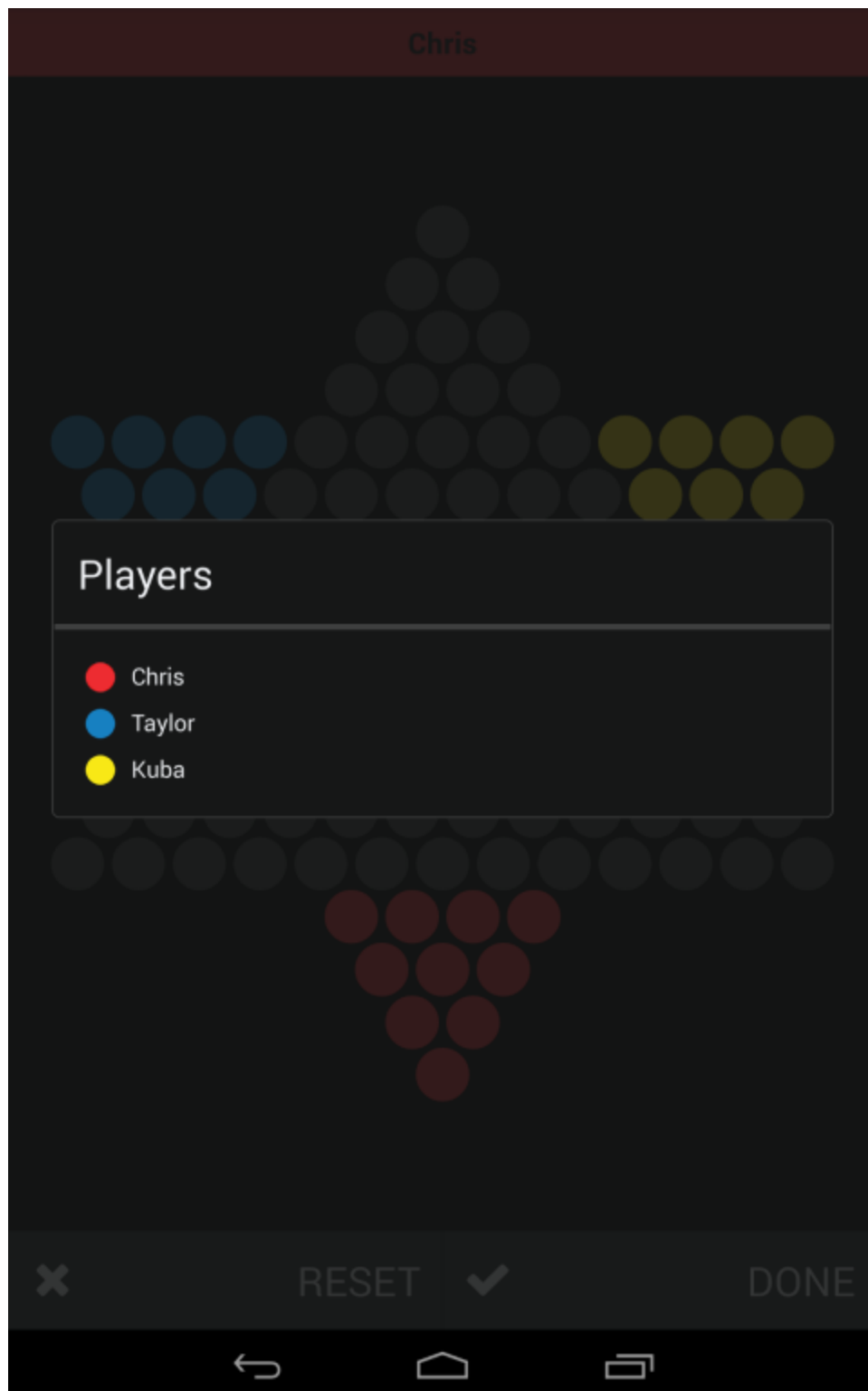


Figure 6.9 Players in a three player game



Figure 6.10 Six player game in progress

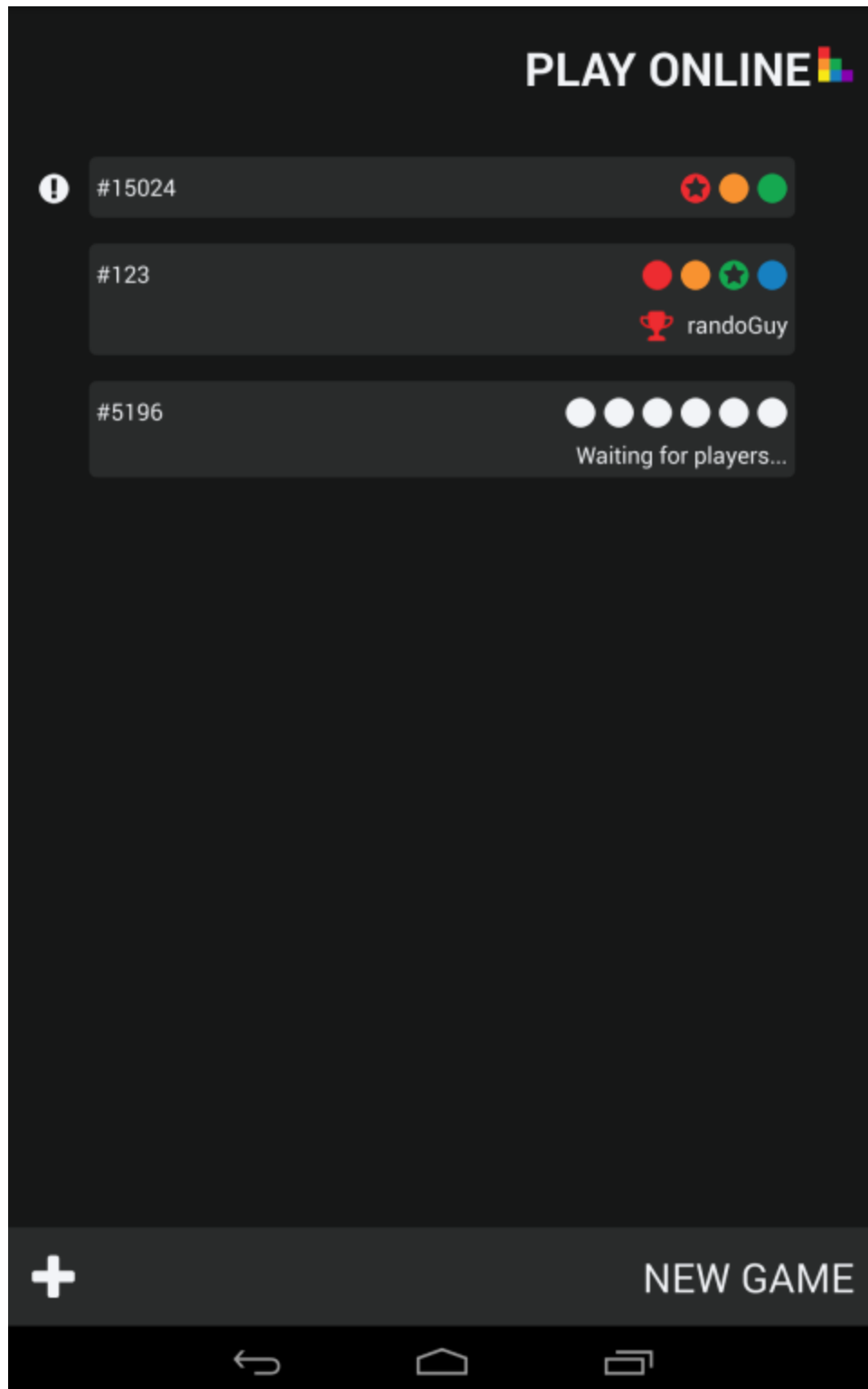


Figure 6.11 Play online activity



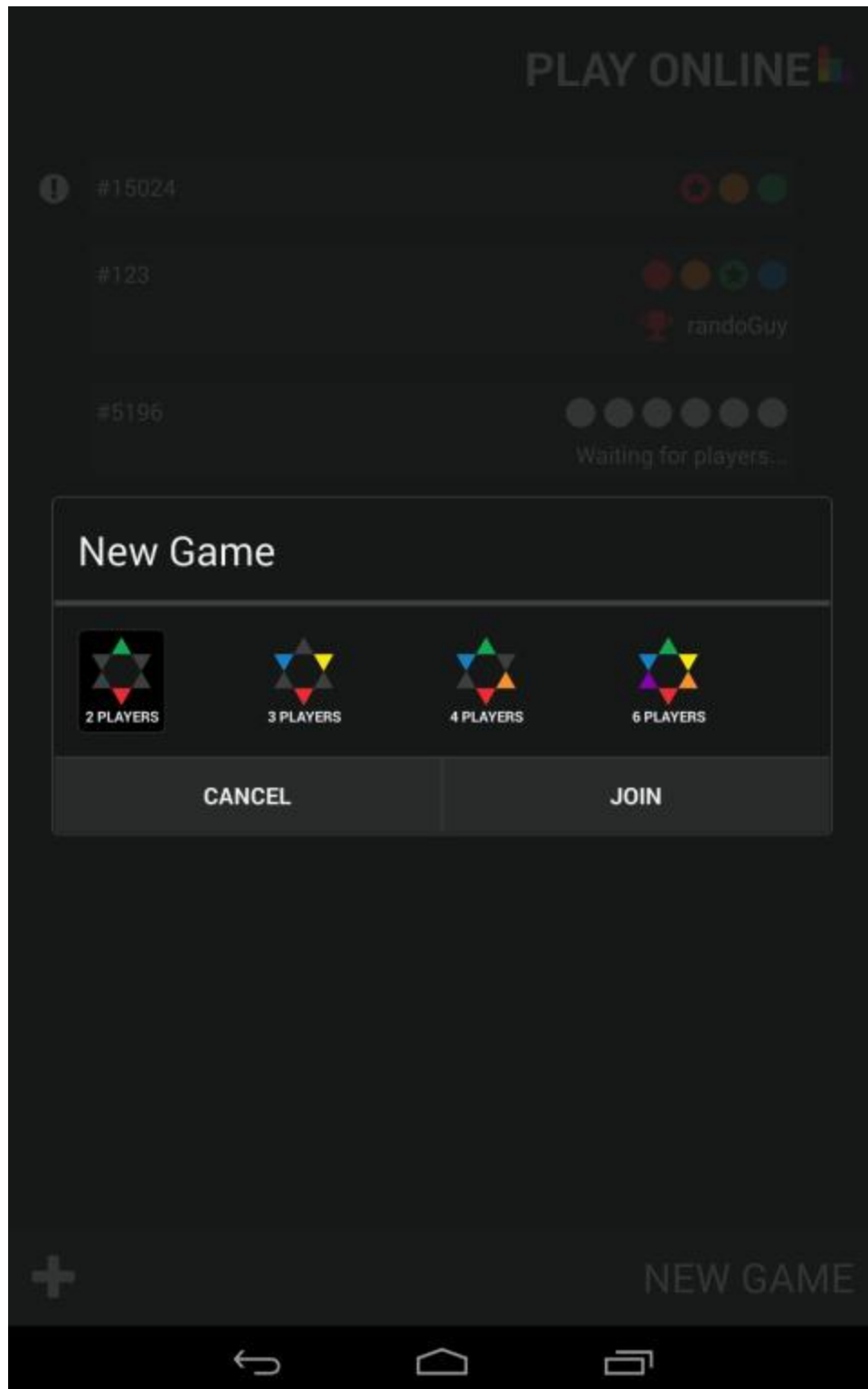


Figure 6.12 Play online activity: new game

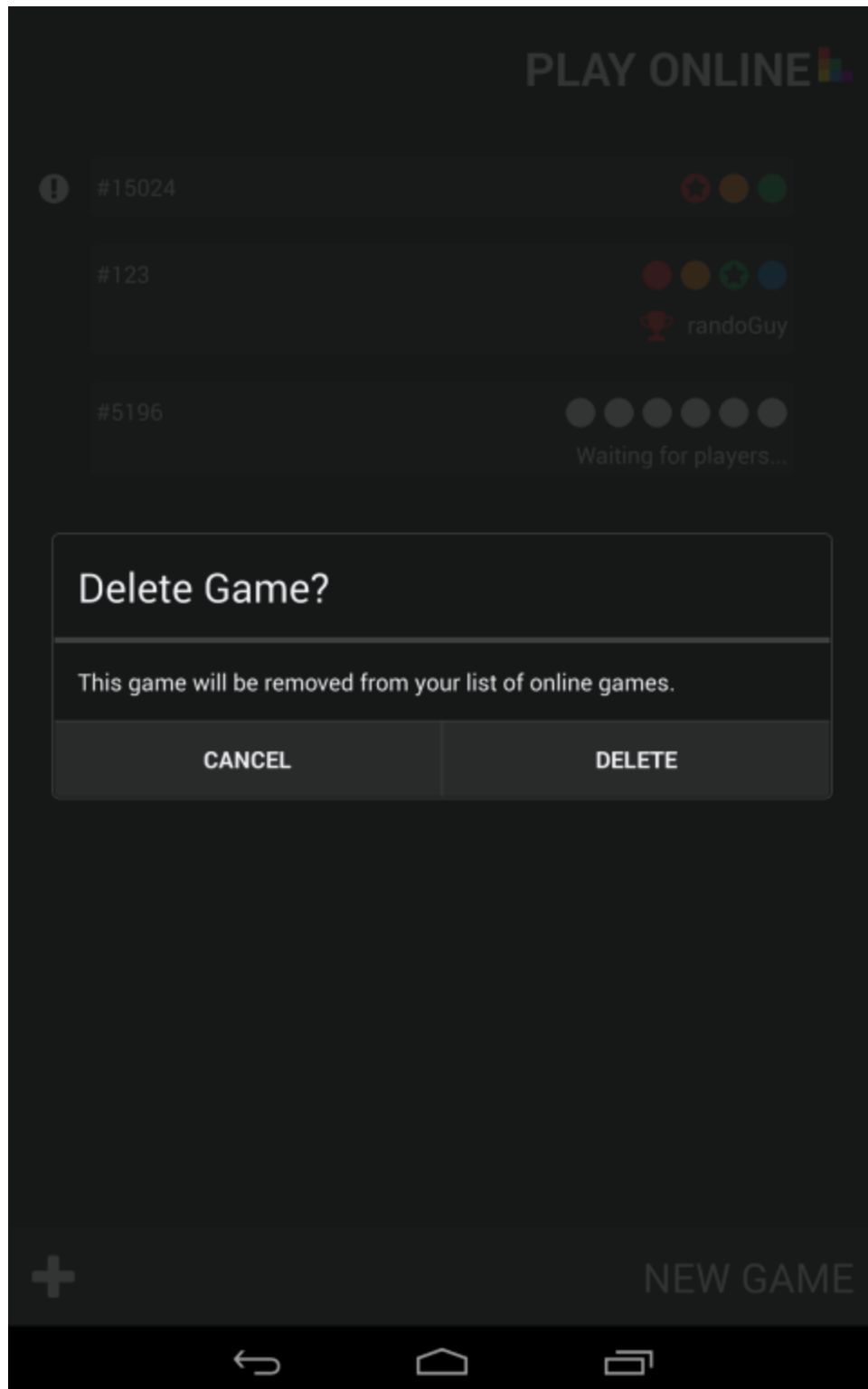


Figure 6.13 Play online activity: delete game

## 6.7 Appendix G

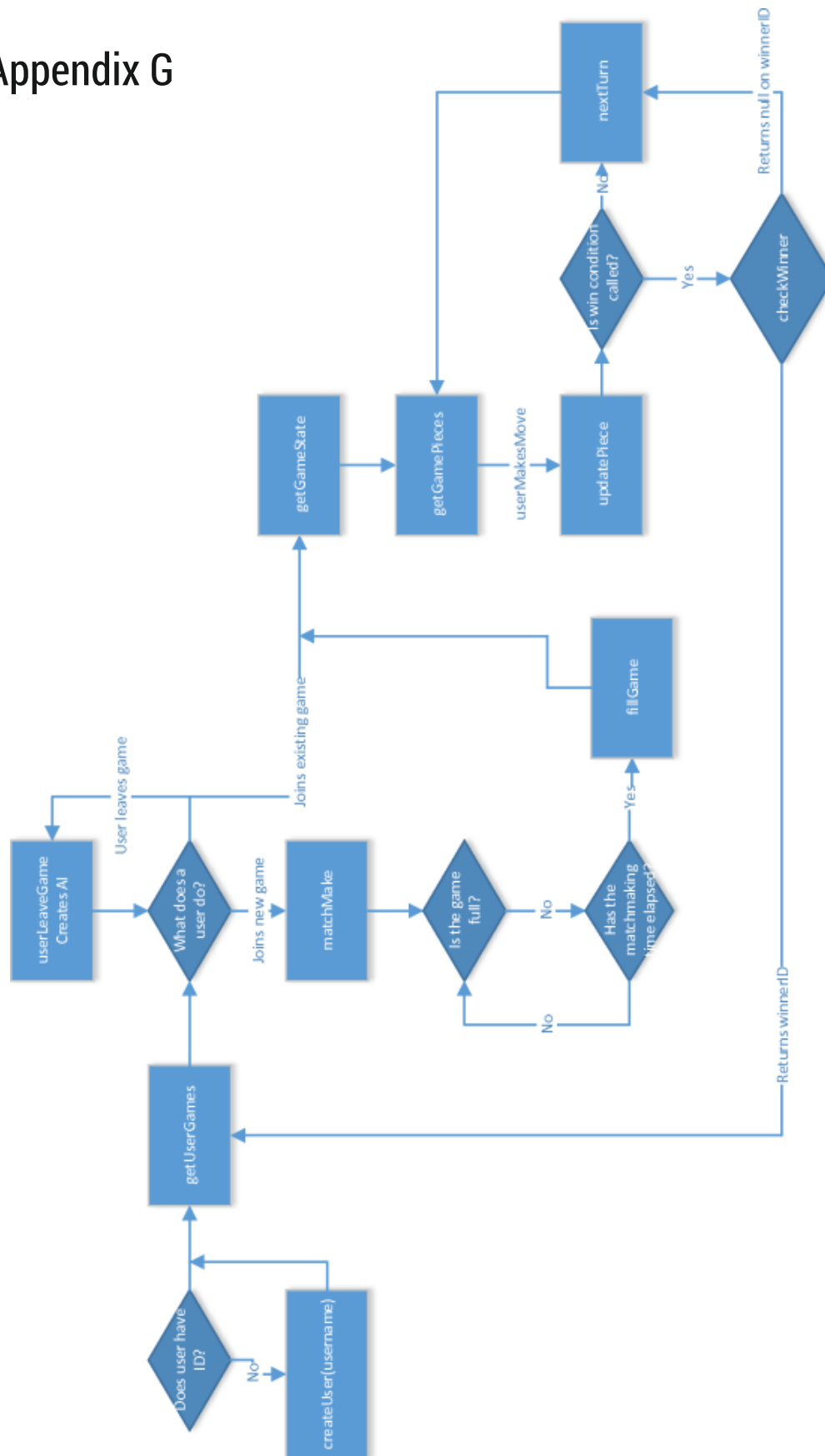


Figure 6.14 Database control flow diagram

## 6.8 Appendix H

### Package ca.brocku.chinesecheckers.tests

[all](#) > ca.brocku.chinesecheckers.tests

26 2 1m14.33s 92%

tests failures duration successful

- [Failed tests](#)
- [Classes](#)

#### Failed tests

Class	Test
	<a href="#">ActivityAITest testChainHopping</a>
	<a href="#">ActivityAITest testCorrectDirection</a>

#### Classes

Class	Tests	Failures	Duration	Success rate
<a href="#">ActivityAITest</a>	3	2	0.031s	33%
<a href="#">GridUnitTest</a>	18	0	0.079s	100%
<a href="#">MainActivityUnitTest</a>	1	0	8.578s	100%
<a href="#">MainAndConfigAndGameIntegrationTest</a>	1	0	27.699s	100%
<a href="#">MainAndConfigIntegrationTest</a>	1	0	19.870s	100%
<a href="#">OfflineConfigurationActivityUnitTest</a>	1	0	10.260s	100%
<a href="#">OfflineGameActivityUnitTest</a>	1	0	7.808s	100%

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.9 Appendix I

### Class `ca.brocku.chinesecheckers.tests.GridUnitTest`

[all](#) > [ca.brocku.chinesecheckers.tests](#) > GridUnitTest

18 0 0.079s 100%

tests failures duration successful

- [Tests](#)

#### Tests

Test	SM-N900W8 - 4.4.2
testAccuracyOfGetPossibleMoves	passed (0.027s)
testAndroidTestCaseSetupProperly	passed (0.025s)
testBoardCreate	passed (0s)
testCorrectNumberOfPiecesReturnend	passed (0s)
testCreateBoardWithValidNumberOfPlayers	passed (0s)
testDoubleSet	passed (0.001s)
testGetAllPiecesNotNull	passed (0s)
testGetPieceEmpty	passed (0s)
testGetPiecesValid	passed (0s)
testInvalidMove	passed (0s)
testMoveOnExistingPiece	passed (0s)
testMovePieceValid	passed (0s)
testOutOfBoundsPiece	passed (0.026s)
testPossibleMovesCountCorners	passed (0s)
testPossibleMovesForCenterPiece	passed (0s)
testSetOutOfBounds	passed (0s)
testSetandGetPiece	passed (0s)
testValidMoves	passed (0s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.10 Appendix J

### Class `ca.brocku.chinesecheckers.tests.ActivityAITest`

[all](#) > [ca.brocku.chinesecheckers.tests](#) > ActivityAITest

3	2	0.031s	33%
---	---	--------	-----

tests	failures	duration	successful
-------	----------	----------	------------

- [Failed tests](#)
- [Tests](#)

#### Failed tests

##### `testChainHopping`

```
junit.framework.AssertionFailedError: Final row should be 4 expected:<4> but was:<10>
at ca.brocku.chinesecheckers.tests.ActivityAITest.hoppingEvaluation(ActivityAITest.java:104)
at ca.brocku.chinesecheckers.tests.ActivityAITest.testChainHopping(ActivityAITest.java:82)
at java.lang.reflect.Method.invokeNative(Native Method)
at android.test.AndroidTestRunner.runTest(AndroidTestRunner.java:191)
at android.test.AndroidTestRunner.runTest(AndroidTestRunner.java:176)
at android.test.InstrumentationTestRunner.onStart(InstrumentationTestRunner.java:554)
at android.app.Instrumentation$InstrumentationThread.run(Instrumentation.java:1741)
```

##### `testCorrectDirection`

```
junit.framework.AssertionFailedError: Final index should be 8 expected:<8> but was:<6>
at ca.brocku.chinesecheckers.tests.ActivityAITest.directionEvaluation(ActivityAITest.java:62)
at ca.brocku.chinesecheckers.tests.ActivityAITest.testCorrectDirection(ActivityAITest.java:21)
at java.lang.reflect.Method.invokeNative(Native Method)
at android.test.AndroidTestRunner.runTest(AndroidTestRunner.java:191)
at android.test.AndroidTestRunner.runTest(AndroidTestRunner.java:176)
at android.test.InstrumentationTestRunner.onStart(InstrumentationTestRunner.java:554)
at android.app.Instrumentation$InstrumentationThread.run(Instrumentation.java:1741)
```

#### Tests

Test	SM-N900W8 - 4.4.2
testAndroidTestCaseSetupProperly	passed (0s)
testChainHopping	failed (0.005s)
testCorrectDirection	failed (0.026s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.11 Appendix K

### Class `ca.brocku.chinesecheckers.tests.MainActivityUnitTest`

[all](#) > [ca.brocku.chinesecheckers.tests](#) > MainActivityUnitTest

1	0	8.578s	100%
---	---	--------	------

tests	failures	duration	successful
-------	----------	----------	------------

- [Tests](#)

#### Tests

Test **SM-N900W8 - 4.4.2**

testActivity passed (8.578s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.12 Appendix L

### Class

#### **ca.brocku.chinesecheckers.tests.MainAndConfigIntegrationTest**

[all](#) > [ca.brocku.chinesecheckers.tests](#) > MainAndConfigIntegrationTest

1	0	19.870s	100%
---	---	---------	------

tests	failures	duration	successful
-------	----------	----------	------------

- [Tests](#)

### Tests

Test SM-N900W8 - 4.4.2

testActivity passed (19.870s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM



## 6.13 Appendix M

### Class

#### ca.brocku.chinesecheckers.tests.MainAndConfigAndGameIntegrationTest

[all](#) > [ca.brocku.chinesecheckers.tests](#) > MainAndConfigAndGameIntegrationTest

1	0	27.699s	100%
---	---	---------	------

tests	failures	duration	successful
-------	----------	----------	------------

- [Tests](#)

### Tests

**Test** SM-N900W8 - 4.4.2

testActivity passed (27.699s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.14 Appendix N

### Class

#### **ca.brocku.chinesecheckers.tests.OfflineGameActivityUnitTest**

[all](#) > [ca.brocku.chinesecheckers.tests](#) > OfflineGameActivityUnitTest

1	0	7.808s	100%
---	---	--------	------

tests	failures	duration	successful
-------	----------	----------	------------

- [Tests](#)

### Tests

Test SM-N900W8 - 4.4.2

testActivity passed (7.808s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.15 Appendix 0

### Class

#### **ca.brocku.chinesecheckers.tests.OfflineConfigurationActivityUnitTest**

[all](#) > [ca.brocku.chinesecheckers.tests](#) > OfflineConfigurationActivityUnitTest

1    0    10.260s 100%

tests failures duration successful

- [Tests](#)

### Tests

Test    SM-N900W8 - 4.4.2

testActivity passed (10.260s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.16 Appendix P

### Package `ca.brocku.chinesecheckers.tests.uiengine`

[all](#) > `ca.brocku.chinesecheckers.tests.uiengine`

10 0 0.028s 100%

tests failures duration successful

- [Classes](#)

#### Classes

Class	Tests	Failures	Duration	Success rate
<a href="#">DimensionsTest</a>	2	0	0.001s	100%
<a href="#">PieceDrawingDetailsTest</a>	2	0	0s	100%
<a href="#">PiecePositionSystemTest</a>	3	0	0.027s	100%
<a href="#">PixelPositionTest</a>	3	0	0s	100%

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.17 Appendix Q

### Class `ca.brocku.chinesecheckers.tests.uiengine.DimensionsTest`

[all](#) > [ca.brocku.chinesecheckers.tests.uiengine](#) > DimensionsTest

2	0	0.001s	100%
---	---	--------	------

tests	failures	duration	successful
-------	----------	----------	------------

- [Tests](#)

#### Tests

**Test**    **SM-N900W8 - 4.4.2**

testGetters passed (0s)

testSetters passed (0.001s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.18 Appendix R

### Class

#### **ca.brocku.chinesecheckers.tests.uiengine.PixelPositionTest**

[all](#) > [ca.brocku.chinesecheckers.tests.uiengine](#) > PixelPositionTest

3	0	0s	100%
---	---	----	------

tests	failures	duration	successful
-------	----------	----------	------------

- [Tests](#)

### Tests

**Test**      **SM-N900W8 - 4.4.2**

testGettersXY passed (0s)

testOffset    passed (0s)

testSettersXY passed (0s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.19 Appendix S

### Class

#### ca.brocku.chinesecheckers.tests.uiengine.PiecePositionSystemTest

[all](#) > [ca.brocku.chinesecheckers.tests.uiengine](#) > PiecePositionSystemTest

3 0 0.027s 100%

tests failures duration successful

- [Tests](#)

### Tests

Test	SM-N900W8 - 4.4.2
testPositionXyCalculation	passed (0s)
testUnmodifiablePositionList	passed (0s)
testUnmodifiablePositionMapping	passed (0.027s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM

## 6.20 Appendix T

### Class

#### ca.brocku.chinesecheckers.tests.uiengine.PieceDrawingDetailsTest

[all](#) > [ca.brocku.chinesecheckers.tests.uiengine](#) > PieceDrawingDetailsTest

2	0	0s	100%
---	---	----	------

tests	failures	duration	successful
-------	----------	----------	------------

- [Tests](#)

### Tests

Test	SM-N900W8 - 4.4.2
------	-------------------

testGetterPosition passed (0s)

testGetterRadius passed (0s)

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM



## 6.21 Appendix U

### Test Summary

36 2 1m14.35s 94%

tests failures duration successful

- [Failed tests](#)
- [Packages](#)
- [Classes](#)

### Failed tests

Class	Test
<a href="#">ActivityAITest</a>	<a href="#">testChainHopping</a>
<a href="#">ActivityAITest</a>	<a href="#">testCorrectDirection</a>

### Packages

Package	Tests	Failures	Duration	Success rate
<a href="#">ca.brocku.chinesecheckers.tests</a>	26	2	1m14.33s	92%
<a href="#">ca.brocku.chinesecheckers.tests.uiengine</a>	10	0	0.028s	100%

### Classes

Class	Tests	Failures	Duration	Success rate
<a href="#">ca.brocku.chinesecheckers.tests.ActivityAITest</a>	3	2	0.031s	33%
<a href="#">ca.brocku.chinesecheckers.tests.GridUnitTest</a>	18	0	0.079s	100%
<a href="#">ca.brocku.chinesecheckers.tests.MainActivityUnitTest</a>	1	0	8.578s	100%
<a href="#">ca.brocku.chinesecheckers.tests.MainAndConfigAndGameIntegrationTest</a>	1	0	27.699s	100%
<a href="#">ca.brocku.chinesecheckers.tests.MainAndConfigIntegrationTest</a>	1	0	19.870s	100%
<a href="#">ca.brocku.chinesecheckers.tests.OfflineConfigurationActivityUnitTest</a>	1	0	10.260s	100%
<a href="#">ca.brocku.chinesecheckers.tests.OfflineGameActivityUnitTest</a>	1	0	7.808s	100%
<a href="#">ca.brocku.chinesecheckers.tests.uiengine.DimensionsTest</a>	2	0	0.001s	100%
<a href="#">ca.brocku.chinesecheckers.tests.uiengine.PieceDrawingDetailsTest</a>	2	0	0s	100%
<a href="#">ca.brocku.chinesecheckers.tests.uiengine.PiecePositionSystemTest</a>	3	0	0.027s	100%
<a href="#">ca.brocku.chinesecheckers.tests.uiengine.PixelPositionTest</a>	3	0	0s	100%

Generated by [Gradle 1.10](#) at Apr 3, 2014 4:05:08 PM