Logical Operators:

Chinese Checkers

Iteration Two

February 27th, 2014

Curtis Smith

Peter Pobojewski

Saajid Mohammed

Chris Kellendonk

James Kostiuk

Kuba Subczynski

Ben Stitt

Taylor Whited

1 Table of contents

2 Planning

2.1 Team capacity calculations

2.11 Release level plan

Start	9-Jan
Finish	27-Mar
Last day of class	4-Apr
Work days	77
Work weeks	11
# of people	8
Hours/Week/Person	6
Total hours	528

2.12 Iteration level plan

	# Weeks	**Hours**
1st iteration	4	192
2nd iteration	3	144
3rd iteration	3	144
4th iteration	1	48
Total hours		528

2.2 Meeting minutes

2.21 February 6, 2014

After some preparation, the group presented the results of iteration one to Vlad. This included introducing the team members, as well as answering questions about tools such as Android Studio, GitHub, and Travis.

After the presentation, the group conducted a regular meeting.

Firstly, the idea of a virtual suggestion box was put forward, and agreed upon.

Then, Taylor made suggestions regarding the format of documentation. Firstly, she volunteered to take over collecting documentation into binders for each iteration. She also suggested that the directories on GitHub should be organized by creator name and by iteration number. It was decided that the group would continue to use the master branch for documentation.

There was also a discussion on format. Documents should use references in APA format, and appendices should have clear and unique titles. Also, documents should be written in present tense, using no personal pronouns.

Next, issues with the application were discussed. Touch areas were not being scaled correctly. Some visual adjustments needed to be made as well, including vertically centering buttons, as well as adjusting the colours of text boxes and adjustments to spacing.

The group agreed to link to commits of related materials in GitHub tasks.

Peter expressed some technical issues involving Android emulation, and having trouble implementing a Java interface.

There were also issues with Travis. It wasn't finding everything, and it was pulling from every branch.

Ben shared his thanks with the group, and resolved to use more tasks to communicate on GitHub.

The group agreed to try to complete tasks at a more even pace, and if a task is too large to complete at once, to break it into smaller tasks.

It was also agreed that the group needed to collaborate more on establishing the architecture of the application. This could be done partly by assigning brainstorming tasks, and they could be brought up at the next meeting to share with the group.

At this point, the group moved to another room to continue the discussion.

The group then discussed what goals or use cases to include in the second iteration. It was decided that an AI and save and continue features should be included. However, a display of pegs remaining to win was deemed an unnecessary frill, and left out.

It was decided that the next meeting be held the following Thursday.

The group was then assigned brainstorming tasks, and spoke briefly on what should occur when a game ends. The group then adjourned.

2.22 February 13, 2014

The group discussed the perfect mark received for the first iteration, and wondered which percentage of the total was represented by this mark.

Then the topic turned to the use cases which had been posted to GitHub, with the intention of dividing these into individual tasks. If time allowed, the group would also speak about issues with Travis.

The first topic discussed was how to save a game. It was decided that it should be automatically saved when the game was closed as a serialized object.

Next, loading a game was discussed. The file should be loaded automatically, and prompt the user to continue the next time they select offline play.

It was decided that possible moves should be shown for a peg when it is tapped. There will be no option to toggle this feature in this iteration.

It was announced that unit tests for the game board are working well.

The discussion returned to saving and loading games. The assigned tasks included making UI mock-ups for the load dialog, updating existing flow diagrams, coding for UI flow control and for loading the board, and creating unit tests. The entire game state must be serialized, rather than just the game board.

Next, the discussion returned to how to show possible moves. The tasks assigned to this use case include creating a UI mock-up, creating a UML or flow diagram, and coding. This feature is difficult to test, and must be done manually.

It was agreed that Hot Seat mode should be renamed to Offline mode.

The topic of in-game help arose next. Tutorial-style overlays were considered, but will not be included in this iteration. Instead, the help section can reside in the settings screen native to Android. This help section should include an FAQ, and rules. Tasks assigned for this use case include creating a UI mock-up, new activity classes in UML, and coding, including a parent class for pages. This can be tested automatically.

Next, it was decided that a game should be saved using the onPause event, and loaded using the onCreate event. The loading dialog box will only be displayed when entering an offline game from the main menu. Loading will be performed while the dialog box is being displayed, so there is no delay after selecting to continue.

Next, the option to play against an AI was discussed. The tasks assigned include the UML and activity diagrams, and an interface should be coded before implementation. It was also decided that the AI should hold its own temporary copy of the game state.

Next, the group discussed how to toggle AI players in the configuration activity. It was decided that a button should appear next to each player's name entry box. Once activated, the applicable text box changes to three radio buttons, indicating the difficulty level. Two UI mock-ups are to be created for this: one for a single button toggle, and one for a switch-style toggle. These changes require modification of the existing offline play flow control diagram as well.

To test AI players, one may set up games in which AIs play against each other. Foremost, an AI player must not break the rules. After this, their ability to win against each other should be proportional to their respective difficulty levels.

The next group meeting was scheduled for the following Thursday, during reading week. Absent group members may attend remotely via Google Hangouts or a similar system.

The group then discussed what would happen when a game is finished. It was decided that the game screen should be blurred, as if behind frosted glass, and a dialog announcing the winner should be displayed, along with one or two buttons. For offline play, it should display a button to return to the main menu. For online play, it should display two buttons: one to return to the main menu, one to spectate for the remainder of the game.

Lastly, Taylor volunteered to reorganize some folders on GitHub.

2.23 February 20, 2014

Note: Due to online meeting, these minutes may be incomplete.

The group spoke over the Internet via Google hangouts.

The meeting began with a review of the iteration 2 burn-down. It seems that the group is opening too many tasks without closing them.

Peter expressed concern over how to implement unit tests. Thus far, he had been testing in JCreator, but will test further in Android Studio.

Next, the group covered each use case in turn.

Taylor showed some test mock-ups for the endgame dialog. The group decided that the game screen should be dimmed, but not blurred.

Kuba completed load dialogs, and Saajid opened a separate issue for testing this.

Next, the group reviewed a mock-up for showing possible moves, and tried to decide on a colour scheme for unoccupied home spaces and for selected pegs. The group agreed that unoccupied home spaces should be lighter, and selected pegs should be darker.

The group was presented with a mock-up for a help screen, and agreed that it was good.

In order to implement an AI player, James needed an interface or guidelines to complete the code.

The group was shown mock-ups for AI selection on the configuration screen, and decided it should start by displaying all robots except for the first player.

For the proper endgame mock-up, Taylor needs a screenshot from Kuba. It was decided that the win condition can be determined from the game board alone.

New tasks were discussed, including the phase 2 testing plan, UI tables for activities, game board animation, acceptance testing, activity and integration tests, the network API, touch detection, and the EULA.

Saajid needs some time for testing, so the group agreed to try to get some modules completed ahead of time.

The group agreed that most or all use cases could be completed by the deadline, but likely cannot all be tested.

The group agreed that play should proceed in a clockwise direction. Variables need to be assigned to determine player home positions on the game board, and a variable needs to be assigned to show that a peg has jumped another.

It was also agreed that only one level of AI needs to be completed for this iteration.

See Appendix A for breakdown of progress of iteration.

2.24 February 25, 2014

Note: These minutes may be incomplete due to the note taker arriving late.

The group discussed what to include in the iteration 2 binder. The iteration 1 presentation will be included.

The group will try to have coding done by Wednesday night.

For acceptance testing, the user must be able to start a game for up to six players. Scripts should be made regarding tasks and expected consequences. The write-up and testing should be done by early Thursday afternoon. A reference should be included for the template used.

The group discussed whether it was worth it to continue using Travis. It was decided that since it does no harm, Travis will be kept in place for the time being.

The group members are to sort their binder directories into subdirectories to designate groupings of files.

The EULA is to be kept as a class rather than an activity, and should use a custom dialog similar to the OfflineGameResumeDialog.

For AI documentation, a flow diagram should be provided. Currently, the AI is selectable, but will not run as intended.

Use cases can be placed under the planning section in the binder.

The burn down looked bad, but was improving. This was further improved by closing completed issue.

2.3 Use cases

Use case templates ("Use Case Templates", 2014) provided by TechnoSolutions.

2.31 Play offline game (IT-2-UC-3)

Revision history

Date	Author	Description of change
2014/02/11	Ben Stitt	Authored use case.
2014/02/16	Ben Stitt	Revised to meet documentation specifications and name change.

Use case

Play offline game

Id

IT-2-UC-3

Description

The user would like to play a complete local game with their friends. This must be made possible without communicating with an external server, or employing an AI.

Level

User goal

Primary actor

The user

Supporting actors

Friends of the user who are present

Stakeholders and interests

The user and their friends

Pre-conditions

The user must have the application installed and running, and have completed the Offline Configuration Activity.

Post conditions

Success end condition

The appropriate game end screen is displayed. It informs the players which of them has won.

Failure end condition

The user is unable to launch the Offline Game Activity.

Minimal guarantee

The application does not crash, and data on the device remains unaffected.

Trigger

The user completes the Hot Seat Configuration Activity.

Main success scenario

1. The user clicks the Start Game button on the Offline Configuration screen.
2. Each player makes their moves in turn.
3. The Offline Game Activity ends with the appropriate victory screen.

Extensions

Alternatively, the user could include one or more AI players.

Frequency

This use case should occur each time the user starts an offline game. As such, it could occur weekly to hourly.

Assumptions

The user has at least one friend present.

Special requirements

The device must run Android and have the necessary space to install the game.

Issues

1. How does the local server coordinate player turns?

To do

1. Return to main menu.

2.32 Play against an AI (IT-2-UC-4)

Revision history

Date	Author	Description of change
2014/02/12	Ben Stitt	Authored use case.

Use case

Play against an AI

Id

IT-2-UC-4

Description

The user has completed the applicable Configuration Activity, and has opted to include one or more AI players. They then wish to play through a game with these AI players.

Level

User Goal

Primary actor

The user

Stakeholders and interests

The user wishes to play a game of Chinese Checkers, but does not have the appropriate number of players present.

Pre-conditions

The user must have the application installed and running, and have completed the appropriate Configuration Activity.

Post conditions

Success end condition
The appropriate win or loss screen is displayed.

Failure end condition
The AI player fails to launch, or fails to take its turns.

Minimal guarantee
The application does not crash, and data on the device remains unaffected.

Trigger

The user completes the appropriate Configuration Activity, and has included one or more AI players in the game.

Main success scenario

1. The user clicks the Start Game button on the appropriate Configuration screen.
2. The user and the AI players make their moves in turn.
3. The Game Activity ends with the appropriate win or loss screen.

Extensions

An AI player could be included in either a local game or a network game.

Variations

An AI can be run either locally, or from a remote server.

Frequency
This use case should occur each time the user starts game that includes an AI player. As such, it could occur weekly to hourly.

Assumptions
The user wishes to play with a number of players different than the number of players otherwise available.

Special requirements

The device must run Android and have the necessary space to install the game. If the AI player is run from a remote server, the user must be able to connect to the Internet.

Issues

1. Is the AI process prepared to run?
2. Is the local and/or network server process prepared to interact with the AI process?

To do

1. Return to main menu.

2.33 Show possible moves (IT-2-UC-5)

Revision history

Date	Author	Description of change
2014/02/12	Ben Stitt	Authored use case.
2014/02/13	Peter Pobojewski	Edited use case to more accurately portray the process.
2014/02/16	Ben Stitt	Revised to meet documentation syntax.

Use case

Show possible moves

Id

IT-2-UC-5

Description

The user has a game in progress, and it is their turn. They wish to see which moves they can make using a certain peg.

Level

User goal

Primary actor

The user

Stakeholders and interests

The may be unfamiliar with the rules of Chinese Checkers, or may not be able to see which moves can be made.

Pre-conditions

The user must have the application installed and running. The user must have a game active, and it must be their turn. They must also be currently selecting a piece they control.

Post conditions

Success end condition

The user sees their possible moves, and is able to make any one of them.

Failure end condition

The user is unable to see their possible moves, or game play is interrupted.

Minimal guarantee

The application does not crash, and data on the device remains unaffected.

Trigger

User touches a piece they wish to move.

Main success scenario

1. The user interacts with the board by touching a piece.
2. The game responds by somehow animating or illuminating places this piece can move to.
3. The player continues their turn as usual.

Frequency

This function is activated whenever a user interacts with a piece they control.

Assumptions

If there is a toggle for this feature, the user has enabled the feature.

Special requirements

The device must run Android and have the necessary space to install the game. The user must also have a game active, and it must be their turn.

Issues

1. Can this feature be disabled manually by the player?
2. How do we distinguish possible moves for a piece?

To do

1. The user may take their turn.

2.34 Save a game (IT-2-UC-6)

Revision history

Date	Author	Description of change
2014/02/13	Ben Stitt	Authored use case
2014/02/13	Peter Pobojewski	Edited use case to be more accurate.
2014/02/16	Ben Stitt	Revised to meet documentation syntax and reflect decisions made in most recent meeting.

Use case

Save a game

Id

IT-2-UC-6

Description

The user wishes to continue an active local game at a later time.

Level

User goal

Primary actor

The user

Stakeholders and interests

The user has more pressing issues at hand and may want to continue their game later. It is important that the user is given the ability to continue a game at their next earliest convenience.

Pre-conditions

The user must have the application installed and running, and be in the midst of an active offline game.

Post conditions

Success end condition

The current game is saved in some file on their device.

Failure end condition

The current game is not saved, the user is notified.

Minimal guarantee

The application does not crash, and data on the device remains unaffected.

Trigger

User exits game and is prompted with a “Would you like to save this game?”

Main success scenario

1. The user triggers the save in a certain way
2. The current game is saved to a local file
3. The application returns to the main menu screen.

Variations

Network games will automatically be saved to a remote server.

Frequency

This use case will be triggered each time a user leaves an offline game without completing it. As such, it could be triggered daily to monthly.

Assumptions

The user wishes to complete the game at a later time, and their device has enough free space to save a game.

Special requirements

The device must run Android and have the necessary space to install the application, and to save the current game. If the game is being played over the network, the device must be connected to the Internet.

Issues

1. Where is the game being saved?

To do

1. Be prepared to load the game.

2.35 Add an AI player (IT-2-UC-7)

Revision history

Date	Author	Description of change
2014/02/15	Ben Stitt	Authored use case.
2014/02/16	Ben Stitt	Revised to meet documentation syntax.

Use case

Add an AI player

Id

IT-2-UC-7

Description

The user has entered the Offline Configuration Activity, and has selected the desired number of players. They wish to exchange slots for human players with those for AI players.

Level

User goal

Primary actor

The user

Stakeholders and interests

The user wishes to play a game of Chinese Checkers, but does not have the appropriate number of players present.

Pre-conditions

The user must have the application installed and running, and have entered the Offline Configuration Activity.

Post conditions

Success end condition

The user is able to add an AI player to the game, taking the place of a potential human player.

Failure end condition

The AI player is not added, leaving the human player in place.

Minimal guarantee

The application does not crash, and data on the device remains unaffected.

Trigger

The user selects the AI button next to the player name text entry box.

Main success scenario

1. The user taps the AI button.
2. The text entry box is replaced by three radio buttons, indicating the desired AI difficulty level.
3. The user taps the desired difficulty button.

Variations

In a network game, AI players may be added automatically when a human player leaves.

Frequency

The use case could occur multiple times during the configuration of a game. As such, time between triggers could range from seconds to weeks.

Assumptions

The user wishes to play with a number of players different than the number of players otherwise available.

Special requirements

The device must run Android and have the necessary space to install the game. If the AI player is run from a remote server, the user must be able to connect to the Internet.

Issues

1. How are AI players generated and passed in?

To do

1. Generate and pass in AI players at game start.

2.36 Finish a game (IT-2-UC-8)

Revision history

Date	Author	Description of change
2014/02/15	Ben Stitt	Authored use case.
2014/02/16	Ben Stitt	Revised to meet documentation syntax.

Use case

Finish a game

Id

IT-2-UC-8

Description

A player has triggered a win condition, and the players must be informed of which of them has won, and end the game if necessary.

Level

User goal

Primary actor

Any player in the active game

Supporting actors

Other players in the active game

Stakeholders and interests

All players in the active game wish to know who has won the game, and if necessary, end it.

Pre-conditions

A player has made a move resulting in all their pegs placed in the opposite triangle.

Post conditions

Success end condition

The user is presented with a dialog box, allowing them to return to the main menu.

Failure end condition

The game fails to end.

Minimal guarantee

The application does not crash, and data on the device remains unaffected. The user is able to force the application to close if necessary.

Trigger

A player's pegs are all moved into the opposite triangle.

Main success scenario

1. The win condition is triggered.
2. The user is presented with a dialog box announcing the winner.
3. The user clicks the button on the dialog box, and is returned to the main menu.

Extensions

In a network game, step 2 is only presented to the winner, and the dialog contains two buttons. If the user does not tap the button from Main success scenario 3 then the user remains in the game as a spectator.

Frequency

This use case occurs at the end of each game. As such, it could occur hourly to weekly.

Assumptions

The user has not quit the active game.

Special requirements

The device must run Android and have the necessary space to install the game. To trigger this use case from a network game, the user must have an active Internet connection.

To do

1. Clear the applicable game save data.

2.37 Move a peg (IT-2-UC-9)

Revision history

Date	Author	Description of change
2014/02/17	Ben Stitt	Authored use case.

Use case

Move a peg

Id

IT-2-UC-9

Description

The user drags their peg from its current location to a new one.

Level

User goal

Primary actor

The user

Stakeholders and interests

The user wishes to complete their turn by moving a game peg across the board.

Pre-conditions

The user must have the application installed and running. The user must have a game active, and it must be their turn.

Post conditions

Success end condition

The peg is moved to the new location on the game board. This change is reflected in the stored game board state, and displayed to the user.

Failure end condition

The peg remains in its current location. No change is made to the game board state, or the change is not displayed.

Minimal Guarantee

The application does not crash, and data on the device remains unaffected. The state of the game board is not altered in any other way.

Trigger

The user drags the peg the wish to move to a valid position.

Main success scenario

1. The user drags the peg.
2. The peg's location is updated on a temporary copy of the game board.
3. The display is updated to show the peg in its new position.
4. The user clicks the "done" button.
5. The permanent game board is updated.

Extensions

In the case that the peg can jump other pegs, steps 1 through 3 can be repeated any number of times.

Variations

In the case that an AI player moves a peg, the process occurs as follows.

1. The AI player reports its move to the game state.
2. The permanent game board is updated.
3. The display is updated to show the peg in its new position.

Frequency

This use case is triggered whenever a player moves a game peg. As such, it could occur seconds or hours apart.

Assumptions

The user is prepared to complete their turn.

Special requirements

The device must run Android and have the necessary space to install the game. The user must also have a game active, and it must be their turn.

Issues

1. Does pixel-position conversion need to be considered?

To do

1. Proceed to the next player's turn.

2.38 View help pages (IT-2-UC-10)

Revision history

Date	Author	Description of change
2014/02/18	Ben Stitt	Authored use case.

Use case

View help pages

Id

IT-2-UC-10

Description

The user wishes to open the in-game help.

Level

User goal

Primary actor

The user requires additional knowledge.

Stakeholders and interests

The user wishes to play the game in a knowledgeable and efficient manner.

Pre-conditions

The user must have the application installed and running.

Post conditions

Success end condition
The user is able to read through and close the help pages.

Failure end condition
The help pages fail to open.

Minimal Guarantee
The application does not crash, and data on the device remains unaffected.

Trigger

The user taps the icon for the Android-native help and settings menu.

Main success scenario

1. The user taps the appropriate menu item.
2. The user reads the help page.
3. The user taps the close button.

Extensions

This process may be applied to other types of pages.

Variations

These pages may also be viewed from an external web site.

Frequency
This use case is only executed when the user requires knowledge of the game. As such, it may only be minutes between uses. Later, the user may never use them.

Assumptions
The user is not aware of, or does not know all of the information stored in these pages.

Special requirements

The device must run Android and have the necessary space to install the application. The device must give the user access to the help and settings menu.

Issues

1. How does Android handle pages of this type?

To do

1. Return to normal execution of the application.

2.39 Read the EULA (IT-2-UC-11)

Revision history

Date	Author	Description of change
2014/02/23	Ben Stitt	Authored use case.

Use case

Read the EULA

Id

IT-2-UC-11

Description

The user opens the latest version of the application for the first time. Before they are able to continue, they must agree to the End User License Agreement.

Level

User Goal

Primary actor

The user

Stakeholders and interests

The user wishes to play Chinese Checkers, and Logical Operators would like to protect their intellectual property.

Pre-conditions

The user must have the application installed and it must be the first time running it since the last update.

Post conditions

Success end condition

The user accepts the EULA, and is able to use the application.

Failure end condition

The user is unable to see or agree to the EULA, or cannot use the application.

Minimal Guarantee

The application does not crash, and data on the device remains unaffected.

Trigger

The user starts the application for the first time since the latest update.

Main success scenario

1. The user is presented with the EULA, and is prompted to either accept or refuse it.
2. The user taps the “Accept” button.
3. The main menu is displayed.

Extensions

1. The user taps the “Refuse” button.
2. The application closes.

Variations

The EULA may also be displayed on a web page for the application.

Frequency

This use case only occurs the first time a user runs a new version of the application. As such, it may be months or years between uses, or it may only be used once.

Assumptions

The user is able to make an informed choice regarding the EULA.

Special requirements

The device must run Android and have the necessary space to install the game.

Issues

1. Will displaying the EULA affect testing in any way?

To do

1. Ensure the EULA is kept up to date for each version released.

2.4 Burn down

2.41 Iteration two progress

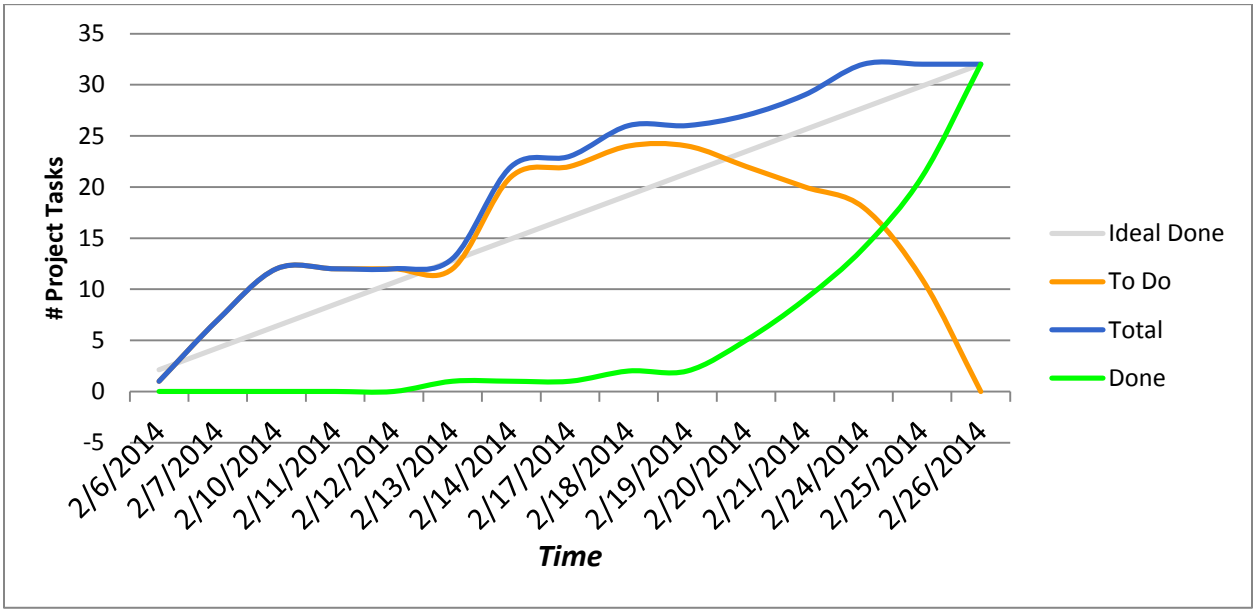


Figure 2.1 Iteration two progress

2.42 Iteration two versus iteration one

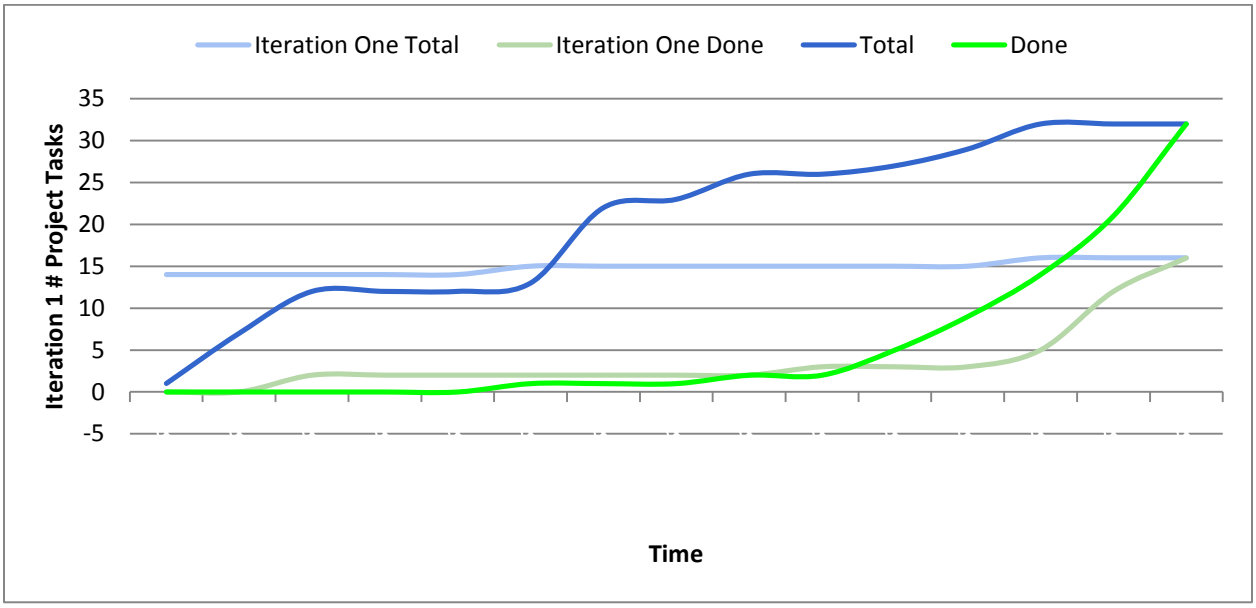


Figure 2.2 Iteration two versus iteration one

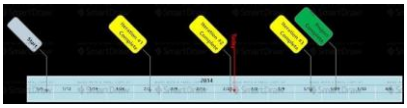
2.5 Presentation

The following presentation will be presented during the retrospective meeting on February 27, 2014. Order follows: left to right, top to bottom.

Review

- Using Agile principles
- Split project into time-boxed iterations
- Created and prioritized the list of features
- Pulled highest priority features into first iteration

Project Timeline

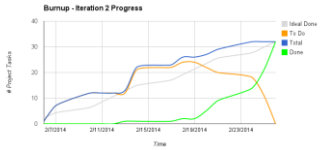


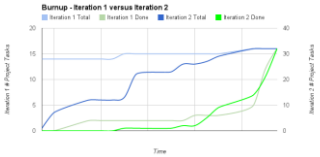
Iteration #2 Goals

- Pulled the next set of features from the backlog
  - Complete the offline version of the game
  - Include "New Game" as well as "Save/Resume Game" features
  - Initiate artificial intelligence integration
- Improve project progress tracking

Planning

- All tasks are tracked in GitHub
- Began with 6 high level use cases which we broke down into project tasks
- Analysis, design, construction and testing of the use cases spawned many subtasks eventually leading to a total of 32 project tasks





Observations

- Iteration 2 was 1 week shorter
- Less effort needed to be dedicated to learning tools
- Some improvement in progress tracking was observed
- Goal for next iteration will be to trend as closely as possible to the "Ideal Done" line

Other Iteration Activities

- GitHub used to house all artifacts, source code and automated tests
- Analysis and design artifacts followed conventions from iteration #1
- Construction and testing was done using AndroidStudio
- travis-ci.org used for continous integration

Ready For A Demo?

Agile Principle #7:

"Working software is the primary measure of progress."

Source: agilemanifesto.org/principles.html

Demonstration

Next Iteration Goals

- Complete AI integration
- Implement network capabilities for online features and game play over the internet

2.6 Policies

2.61 End-user license agreement (EULA)

By downloading any application from Logical Operators or Google™ (hereafter referred to as "Google"), installing or using this application or any portion thereof ("Application"), you agree to the following terms and conditions (the "Terms and Conditions").

1. Use of application

a. Logical Operators grants you the non-exclusive, non-transferrable, limited right and license to install and use this Application solely and exclusively for your personal use.

b. You may not use the Application in any manner that could damage, disable, overburden, or impair the Application (or servers or networks connected to the Application), nor may you use the Application in any manner that could interfere with any other party's use and enjoyment of the Application (or servers or networks connected to the Application).

c. You agree that you are solely responsible for (and that Logical Operators has no responsibility to you or to any third party for) your use of the Application, any breach of your obligations under the Terms and Conditions, and for the consequences (including any loss or damage which Logical Operators may suffer) of any such breach.

2. Proprietary rights

You acknowledge that (a) the Application contains proprietary and confidential information that is protected by applicable intellectual property and other laws, and (b) Logical Operators and/or third parties own all right, title and interest in and to the Application and content, excluding content provided by you, that may be presented or accessed through the Application, including without limitation all Intellectual Property Rights therein and thereto. "Intellectual Property Rights" means any and all rights existing from time to time under patent law, copyright law, trade secret law, trademark law, unfair competition law, and any and all other

proprietary rights, and any and all applications, renewals, extensions and restorations thereof, now or hereafter in force and effect worldwide. You agree that you will not, and will not allow any third party to, (i) copy, sell, license, distribute, transfer, modify, adapt, translate, prepare derivative works from, decompile, reverse engineer, disassemble or otherwise attempt to derive source code from the Application or content that may be presented or accessed through the Application for any purpose, unless otherwise permitted, (ii) take any action to circumvent or defeat the security or content usage rules provided, deployed or enforced by any functionality (including without limitation digital rights management functionality) contained in the Application, (iii) use the Application to access, copy, transfer, transcode or retransmit content in violation of any law or third party rights, or (iv) remove, obscure, or alter Logical Operators', or any third party's copyright notices, trademarks, or other proprietary rights notices affixed to or contained within or accessed in conjunction with or through the Application.

3. Google terms of service and privacy policy

a. Google's Privacy Policy (located at http://www.google.com/privacypolicy.html) explains how Google treats your information and protects your privacy when you use the Application. You agree to the use of your data in accordance with The Google's privacy policies.

b. The Application may contain features that are used in conjunction with Google's search and other services. Accordingly, your use of such features of the Application is also governed by Google's Terms of Service located at http://www.google.com/terms_of_service.html, Google's Privacy Policy located at http://www.google.com/privacypolicy.html, as well as any applicable Google Service-specific Terms of Service and Privacy Policy, which may be updated from time to time and without notice.

4. Logical Operators terms of service and privacy policy

a. Logical Operators may collect information through the Application in order to build an online profile and establish rankings. Logical Operators will not use this information for any other purpose, and will not share it with any third party.

b. Logical Operators will abide by Google's Privacy Policy as outlined in section 3. In addition, Logical Operators will require you to follow Google's Terms of Service as outlined in section 3.

5. Termination

These Terms and Conditions will continue to apply until terminated by either you or Logical Operators as set forth below. You may terminate these Terms and Conditions at any time by permanently deleting the Application from your mobile device in its entirety. Your rights

automatically and immediately terminate without notice from Logical Operators, or any Third Party if you fail to comply with any provision of these Terms and Conditions. In such event, you must immediately delete the Application

6. Indemnity

To the maximum extent permitted by law, you agree to defend, indemnify and hold harmless Logical Operators, their affiliates and their respective directors, officers, employees and agents from and against any and all claims, actions, suits or proceedings, as well as any and all losses, liabilities, damages, costs and expenses (including reasonable attorneys fees) arising out of or accruing from your use of the Application, including your downloading, installation, or use of the Application, or your violation of these Terms and Conditions.

7. Disclaimer of warranties

a. You expressly understand and agree that your use of the Application is at your sole discretion and risk, and that the Application is provided as is and as available, without warranty of any kind.

b. You are solely responsible for any damage to your mobile device, or other device, or loss of data that results from such use.

c. Logical Operators further expressly disclaim all warranties and conditions of any kind, whether express or implied, including, but not limited to the implied warranties and conditions of merchantability, fitness for a particular purpose and non-infringement, with respect to the Application.

d. The Application is not intended for use in the operation of nuclear facilities, life support systems, emergency communications, aircraft navigation or communication systems, air traffic control systems, or any other activities in which the failure of the Application could lead to death, personal injury, severe injury, or environmental damage.

8. Limitation of liability

You expressly understand and agree that Logical Operators, their subsidiaries and affiliates, and their licensors are not liable to you under any theory of liability for and direct, indirect, incidental, special consequential, or exemplary damages that may be incurred by you through your use of the Application, including any loss of data or damage to your mobile device, whether or not Logical Operators, or their representatives have been advised of or should be aware of the possibility of any such loss arising.

9. Miscellaneous

a. These Terms and Conditions constitute the entire Agreement between you and Logical Operators relating to the Application and govern your use of the Application, and completely replace any prior or contemporaneous agreements between you and Logical Operators regarding the Application.

b. The failure of Logical Operators to exercise or enforce any right or provision of these Terms and Conditions does not constitute a waiver of such right or provision, which will still be available to Logical Operators.

c. If any court of law, having the jurisdiction to decide on this matter, rules that any provision of these Terms and Conditions is invalid, then that provision will be removed from the Terms and Conditions without affecting the rest of the Terms and Conditions. The remaining provisions of these Terms and Conditions will continue to be valid and enforceable.

d. The rights granted in these Terms and Conditions may not be assigned or transferred by either you or Logical Operators without the prior written approval of any of these parties. Neither you nor Logical Operators are permitted to delegate their responsibilities or obligations under these Terms and Conditions without the prior written approval of the other party.

e. These Terms and Conditions and your relationship with Logical Operators under these Terms and Conditions will be governed by the laws of the Province of Ontario, Canada without regard to its conflict of laws provisions. You and Logical Operators agree to submit to the exclusive jurisdiction of the courts of the Province of Ontario to resolve any legal matter arising from these Terms and Conditions. Notwithstanding this, you agree that Logical Operators will still be allowed to apply for injunctive remedies (or an equivalent type of urgent legal relief) in any jurisdiction.

3 Design

3.1 Class diagrams

3.11 Game board engine

- See Appendix F

3.2 User interface control elements

3.21 Main activity

Component ID	Control Type	Effect
offlineActivityConfigurationButton	Button	Starts the OfflineConfigurationActivity OR OfflineGameActivity (with resume dialog) if game is saved

3.22 Offline Configuration activity

Component ID	Control Type	Effect
offlineTwoPlayerButton	ToggleButton	Displays two input fields: • offlineRedPlayerNameEditText • offlineBluePlayerNameEditText
offlineThreePlayerButton	ToggleButton	Displays three input fields: • offlineRedPlayerNameEditText • offlinePurplePlayerNameEditText • offlineYellowPlayerNameEditText
offlineFourPlayerButton	ToggleButton	Displays four input fields: • offlineRedPlayerNameEditText • offlineGreenPlayerNameEditText • offlineBluePlayerNameEditText • offlineOrangePlayerNameEditText
offlineSixPlayerButton	ToggleButton	Displays all six input fields
offlineRedPlayerTypeButton	ToggleButton	Human by default; toggles to AI
offlineGreenPlayerTypeButton	ToggleButton	AI by default; toggles to Human

offlinePurplePlayerTypeButton	ToggleButton	
offlineBluePlayerTypeButton	ToggleButton	
offlineYellowPlayerTypeButton	ToggleButton	
offlineOrangePlayerTypeButton	ToggleButton	
Offline<COLOUR>PlayerEasyButton	ToggleButton	Note: replace <COLOUR> with the 6 player colours. This has been done to be concise. Easy difficulty is the default Selecting another difficulty deselects the current one
Offline<COLOUR>PlayerMediumButton	ToggleButton	
Offline<COLOUR>PlayerHardButton	ToggleButton	
offlineRedPlayerNameEditText	EditText	Allows for textual entry Validation: • Can't be blank depending on number of human players selected
offlineGreenPlayerNameEditText	EditText	
offlinePurplePlayerNameEditText	EditText	
offlineBluePlayerNameEditText	EditText	
offlineYellowPlayerNameEditText	EditText	
offlineOrangePlayerNameEditText	EditText	
offlineGameActivityButton	Button	Finishes this activity and starts OfflineGameActivity

3.23 Offline Game activity

Component ID	Control Type	Effect
offlineMoveResetButton	Button	Undoes any peg movements since start of current turn
offlineMoveDoneButton	Button	Updates game with player's move Rotates the board Changes the current player label at the top of the screen

3.24 Offline Game activity: resume dialog

Component ID	Control Type	Effect
offlineAcceptContinuationButton	Button	Dialog disappears
offlineDeclineContinuationButton	Button	This activity finishes and OfflineConfigurationActivity starts Saved game is deleted

3.25 Offline Game activity: end of game dialog

Component ID	Control Type	Effect
offlineGameEndToHomeButton	Button	This activity finishes and MainActivity starts

offlineGameEndToNewButton	Button	This activity finishes and OfflineConfigurationActivity starts

3.3 User interface control flow

3.31 Overview

- See Appendix G

3.32 Main activity

- See Appendix H

3.33 Offline game

- See Appendix I

3.34 Offline game activity: resume dialog

- See Appendix J

3.35 Offline game activity: end of game dialog

- See Appendix K

3.36 Offline configuration activity

- See Appendix L

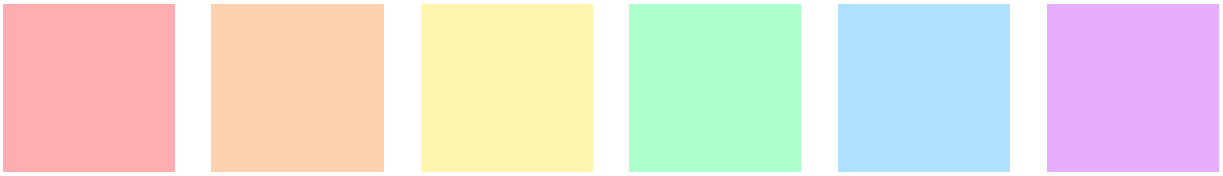
3.4 Visual design

3.41 Introduction

The purpose of this document is to analyse and describe why and how Logical Operators styled the Chinese Checkers interface for iteration two of the application. All visual designs and styles were created after an agreed upon user interface was created for the application. The iteration two visual design process is an extension of the iteration one visual design process with more modifications than new features.

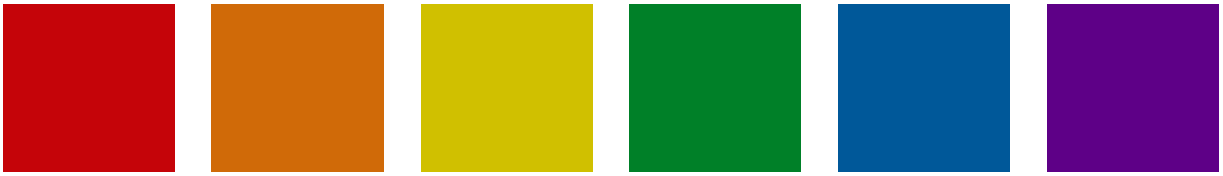
3.42 Colour swatches

Light Primary Palette



rgb(255, 173, 176) rgb(255, 210, 173) rgb(255, 246, 173) rgb(173, 255, 203) rgb(173, 225, 255) rgb(231, 173, 225)

Dark Primary Palette



rgb(197, 4, 9) rgb(208, 106, 8) rgb(208, 192, 0) rgb(0, 128, 40) rgb(0, 88, 153) rgb(94, 0, 135)

Chosen swatches

The Primary Palette and Monochrome Palette from iteration one still contain the most prominent swatches used throughout the application. The Light Primary Palette contains the swatches for a lighter shade of a colour from the Primary Palette. The lighter colours will be predominantly used to signify an empty peg space within a player's home. The Dark Primary

Palette contains swatches for a darker shade of colour from the Primary Palette. The darker colours show which peg is currently selected by the user.

Rejected swatches

Rejected swatches may be revisited in future iterations, but are not significant enough to this iteration to include in this design document.

3.43 Mock-ups

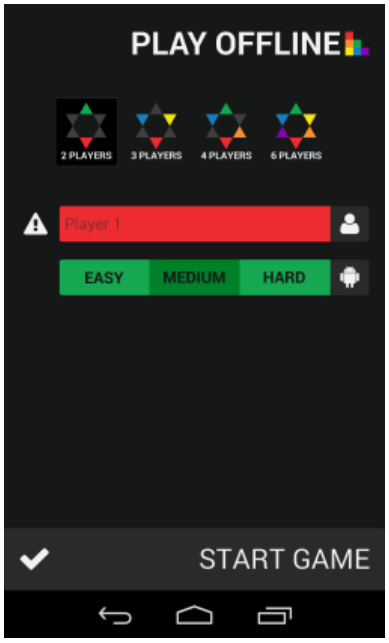


Figure 3.1 Play offline configuration

See Appendix B for full-scale mock-up image

Figure 3.3 Application dialog

See Appendix D for full-scale mock-up image

Figure 3.2 Play offline game

See Appendix C for full-scale mock-up image

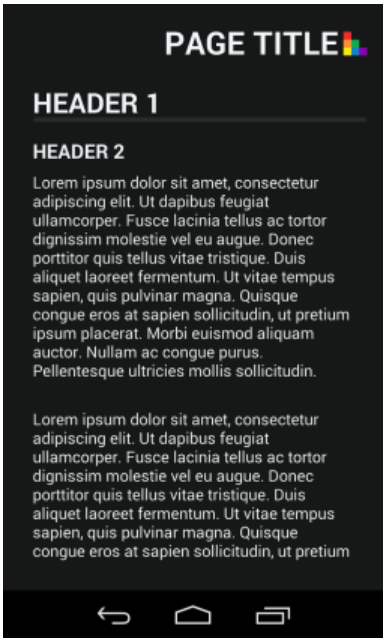


Figure 3.4 Static page

See Appendix E for full-scale mock-up image

Taking into account the design requirements specified in this document, four mock-up images have been produced that display the layouts of the application for iteration one as accurately as possible:

- The Play offline configuration mock-up has been modified since iteration one to show how a user can select an AI to play against as well. The title of the activity has also been changed to “PLAY OFFLINE” to reflect the change of renaming “hotseat” to “offline”.
- Playing an offline game now shows which peg is selected, which pegs are missing from their home and the possible moves which are outline in a light grey from the Monochrome Palette.
- Application dialogs will be used within the application to give appropriate feedback to the user and allow them to navigate appropriately between activities. The design strictly follows the Monochrome Palette by default, but can be customized with buttons and text flavoured to the other palettes.
- A static page activity has been designed for pages such as Help and the EULA. It has been designed to be bare-bones basic to make way for generalization.

3.5 Artificial intelligence (AI) design

3.51 Internal flow of artificial intelligence players

The decided upon flow plan for the internal components of the different AI difficulties was one built for maximal code reuse and minimal complexity. See Appendix M for the visual representation of the flow of AI. During the setup of a game, players will choose the number and difficulties of artificial intelligence players. Upon game start a new AI object will be created for each one of these players, storing their player number, difficulty, and a randomly chosen name with the suffix “bot”.

These objects will wait until the server calls on them to provide a move of accurate level for their difficulty at which point the AI will create a data structure containing all game states currently able to be reached by the board in one move. Note that restrictions will be applied to reduce redundancies and increase response speed. During this process the AIs will gage the moves based on their individually customized heuristics, saving the preferred move to be sent back to the server either after all possibilities have been processed, or a time limit has been reached. This process repeats whenever the server calls on the artificial player for a move, or until the game ends at which point all AI objects are destroyed.

4 Testing

4.1 Iteration two testing plan

4.11 Revision history

Date	Author	Description of change
2014/02/18	Saajid Mohammed	Authored testing plan
2014/02/20	Ben Stitt	General revision and editing

4.12 Introduction

The program to be tested is a full functioning offline version of a Chinese checkers game for Android, with the capability of playing against artificial intelligence. It contains four activities. The first activity has a button leading to the second. The second 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 player is a computer opponent. It then launches the game activity that contains a canvas with the game board drawn on it and buttons to confirm a player's move and undo a move. The last activity is a help menu that is accessible through the context menu of every activity, where the players would be able to view frequently asked questions and game information here.

4.13 Objectives

Verify the current phase programming is functional and works consistently.

4.14 Testing phases

UI Automated Unit testing

- MainActivity
    - offlineActivityConfigurationButton
    - Verify clickabilty

    - Verify visibility
    - Verify proper methods are being called
- OfflineConfigurationActivity
    - offlineTwoPlayerButton
        - Verify clickabilty
        - Verify visibility
        - Verify displays 2 input fields are shown
    - offlineThreePlayerButton
        - Verify clickabilty
        - Verify visibility
        - Verify displays 3 input fields are shown
    - offlineFourPlayerButton
        - Verify clickabilty
        - Verify visibility
        - Verify displays 4 input fields are shown
    - offlineSixPlayerButton
        - Verify clickabilty
        - Verify visibility
        - Verify displays 6 input fields are shown
    - offlineRedPlayerNameEditText
        - Verify visibility
        - Verify accepts text
        - Verify proper error on no input
    - offlineRedPlayerTypeButton
        - Verify visibility
        - Verify proper default selection
        - Verify clickability
        - Verify proper on AI selection difficulty settings are produced
    - OfflineRed PlayerEasyButton
        - Verify visibility
        - Verify clickability
        - Verify proper default selction
        - Verify proper deselecting and selecting
    - OfflineRedPlayerMediumButton
        - Verify visibility
        - Verify clickability
        - Verify proper default selection
        - Verify proper deselecting and selecting

- OfflineRedPlayerHardButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selection
  - Verify proper deselecting and selecting
- offlineGreenPlayerNameEditText
  - Verify visibility
  - Verify accepts text
  - Verify proper error on no input
- offlineGreenPlayerTypeButton
  - Verify visibility
  - Verify proper default selection
  - Verify clickability
  - Verify proper on AI selection difficulty settings are produced
- OfflineGreenPlayerEasyButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selction
  - Verify proper deselecting and selecting
- OfflineGreenPlayerMediumButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selection
  - Verify proper deselecting and selecting
- OfflineGreenPlayerHardButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selection
  - Verify proper deselecting and selecting
- offlinePurplePlayerNameEditText
  - Verify visibility
  - Verify accepts text
  - Verify proper error on no input
- offlinePurplePlayerTypeButton
  - Verify visibility
  - Verify proper default selection
  - Verify clickability
  - Verify proper on AI selection difficulty settings are produced

- OfflinePurple PlayerEasyButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selction
  - Verify proper deselecting and selecting
- OfflinePurplePlayerMediumButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selection
  - Verify proper deselecting and selecting
- OfflinePurplePlayerHardButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selection
  - Verify proper deselecting and selecting
- offlineBluePlayerNameEditText
  - Verify visibility
  - Verify accepts text
  - Verify proper error on no input
- offlineBluePlayerTypeButton
  - Verify visibility
  - Verify proper default selection
  - Verify clickability
  - Verify proper on AI selection difficulty settings are produced
- OfflineBluePlayerEasyButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selction
  - Verify proper deselecting and selecting
- OfflineBluePlayerMediumButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selection
  - Verify proper deselecting and selecting
- OfflineBluePlayerHardButton
  - Verify visibility
  - Verify clickability
  - Verify proper default selection

        - Verify proper deselecting and selecting
    - offlineYellowPlayerNameEditText
        - Verify visibility
        - Verify accepts text
        - Verify proper error on no input
    - offlineYellowPlayerTypeButton
        - Verify visibility
        - Verify proper default selection
        - Verify clickability
        - Verify proper on AI selection difficulty settings are produced
    - OfflineYellowPlayerEasyButton
    - Verify visibility
        - Verify clickability
        - Verify proper default selction
        - Verify proper deselecting and selecting
    - OfflineYellowPlayerMediumButton
        - Verify visibility
        - Verify clickability
        - Verify proper default selection
        - Verify proper deselecting and selecting
    - OfflineYellowPlayerHardButton
        - Verify visibility
        - Verify clickability
        - Verify proper default selection
        - Verify proper deselecting and selecting
    - offlineOrangePlayerNameEditText
        - Verify visibility
        - Verify accepts text
        - Verify proper error on no input
    - offlineOrangePlayerTypeButton
        - Verify visibility
        - Verify proper default selection
        - Verify clickability
        - Verify proper on AI selection difficulty settings are produced
    - OfflineOrangePlayerEasyButton
        - Verify visibility
        - Verify clickability
        - Verify proper default selction

        - Verify proper deselecting and selecting
    - OfflineOrangePlayerMediumButton
        - Verify visibility
        - Verify clickability
        - Verify proper default selection
        - Verify proper deselecting and selecting
    - OfflineOrangePlayerHardButton
        - Verify visibility
        - Verify clickability
        - Verify proper default selection
        - Verify proper deselecting and selecting
    - offlineGameActivityButton
        - Verify clickabilty
        - Verify visibility
        - Verify proper methods are being called
- OfflineGameActivity
    - offlineMoveResetButton
        - Verify clickabilty
        - Verify visibility
        - Verify proper methods are being called
    - offlineMoveDoneButton
        - Verify clickabilty
        - Verify visibility
        - Verify proper methods are being called
    - gameCanvas
        - Verify pieces are movable
        - Verify game hints are being displayed
        - Verify invalid moves are found
        - Verify game is playable
- HelpActivity
    - Verify text is visible

UI Automated Integration testing

- MainActivity and OfflineConfigurationActivty and HelpActivity
    - Redo unit tests
    - Verify transitions between activities are functional

- MainActivity and OfflineConfigurationActivty and GameBoardActivity and HelpActivity
    - Redo unit tests
    - Verify transitions between activities are functional

Manual testing

- Verify game play is complaint with the rules of Chinese Checkers as stated at http://www.mastersgames.com/rules/chinese-checkers-rules.htm
- Verify dialog functionality
    - ResumeDialog
        - Verify is called at proper time
        - Verify visibility
        - offlineAcceptContinuationButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called
        - offlineDeclineContinuationButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called
    - EndofGameDialogDialog
        - Verify is called at proper time
        - Verify visibility
        - offlineGameEndToHomeButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called
        - offlineGameEndToNewButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called

Performance and stress testing

N/A

Regression testing

N/A. Modification were made on the activities that render the old test useless.

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 N). 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 O) shall be generated. The feedback form will be generated as an issue on GitHub with the contents of the (Appendix O). Further action shall be determined on case bases.

4.16 Features to be tested

- MainActivity
    - Button functionality
    - Transition to OfflineConfigurationActivity
    - Transition to HelpActivity
- OfflineConfigurationActivity
    - Button functionality
    - Transition to OfflineGameActivity
    - Transition to HelpActivity
    - Transition to MainActivity
- OfflineGameActivity
    - Button functionality
    - Game playability
    - Popup Dialogs
    - Transition to HelpActivity

- HelpActivity
    - Button functionality
    - Transition back to activity called from
- Game adheres to rules as specified at http://www.mastersgames.com/rules/chinese-checkers-rules.htm

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 Pobojewski	Deputy Lead	_____	_____
Ben Stitt	Documentation Lead	_____	_____
Saajid Mohammed	Test Lead	_____	_____

4.2 Integration testing

- See Appendix R

4.3 Unit testing

4.31 Main tests

- See Appendix S

GridUnitTest

- See Appendix T

MainActivityUnitTest

- See Appendix U

MainAndConfigAndGameIntegrationTest

- See Appendix V

MainAndConfigIntegrationTest

- See Appendix W

OfflineConfigurationActivityUnitTest

- See Appendix X

OfflineGameActivityUnitTest

- See Appendix Y

4.32 UI engine tests

- See Appendix Z

DimensionsTest

- See Appendix AA

PieceDrawingDetailsTest

- See Appendix AB

PiecePositionSystemTest

- See Appendix AC

PixelPositionTest

- See Appendix AD

4.33 Summary

- See Appendix AE

4.4 Acceptance testing

4.41 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.

4.42 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 operating system (OS) version 4.0, API version level 14). If there is time we will also consider an implementation that can be used on a suitable tablet platform.

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

Graphics will be created using OpenGL.

Networking will be accomplished using the internet via HTTP and centralized on a web server using Node.js. 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 Node 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.

4.43 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

4.44 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.

Testing

All tests taking place during acceptance testing will be outlined in this section. The following table is an example. Columns that are coloured grey mean that that particular test will not occur during that test phase.

#	Test Description	Expected Result	Phase 1	Phase 2	Phase 3	Phase 4
1	Action 1 performed...	Expected Result...				
2	Action 2 performed...	Expected Result...				

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 Offline game button	The Offline Game Configuration Activity launches, displaying the number of players, name entry fields, and icons indicating whether each player is a human or AI.	X	
2	Tap each button indicating a different number of players on the Offline Game Configuration Activity screen.	The number of player name entry fields change to the desired number.	X	
3	Tap the human/AI player selection button for each player on the Offline Game Configuration Activity screen. Repeat this action for each number of players selected.	The icon on the button changes back and forth between a silhouette of a person and that of a robot. The name field changes into 3 buttons indicating different difficulty levels, and subsequently returns to being a name entry field.	X	
4	On the Offline Game Configuration Activity screen with a player switched to AI control, tap each of the buttons indicating the difficulty level. Repeat this action for each difficulty level.	The button tapped is a darker colour, and the other buttons in the group are a lighter colour.	X	
5	Fill each player name entry field, and tap the Start Game button on the Offline Game Configuration Activity screen. Repeat this action for each number of players selected.	The Offline Game Activity is launched, and the game board is populated and displayed with pegs in the correct start locations for the specified number of players.	X	

6	In the Offline Game Activity, tap one of the pegs belonging to the active player.	The peg changes to a deeper colour. Empty spaces to which the peg can move change colour as well.	X	
7	In the Offline Game Activity, tap one of the pegs belonging to the active player. Tap a space not indicated as allowable.	Nothing happens.	X	
8	In the Offline Game Activity, tap one of the pegs belonging to the active player. Tap a space indicated as allowable.	The peg moves to the location indicated by the second tap, and the user is presented with the option to confirm or reset their move.	X	
9	In the Offline Game Activity, after moving a peg, tap the Reset button.	The peg returns to its position at the start of the turn, and the user is able to take their turn again from the beginning.	X	
10	In the Offline Game Activity, after moving a peg, tap the Confirm button.	The peg remains at its new location, and play passes to the next player.	X	
11	Repeat task 6, tapping a peg which should be able to jump another.	The result is the same as in task 6, except the possible moves include the jump.	X	
12	Repeat task 8, except in order to jump another peg.	The result is the same as in task 8, except the user is able to move again.	X	
13	Repeat task 7, except after jumping at least 1 peg.	Nothing happens.	X	
14	Repeat task 9, except after jumping at least 2 pegs.	The result is the same as in task 9.	X	
15	Repeat task 10, except after jumping at least 1 peg.	The result is the same as in task 10.	X	
16	Close the application and remove it from active memory while a game is active. Open it again, and tap the Offline game button.	The user is presented with the option to resume the game in progress, or begin a new one.	X	
17	After task 16, tap the Resume button.	The game resumes its progress from its previous state.	X	
18	After task 16, tap the Start New Game button.	The Offline Game Configuration Activity launches.	X	
19	Get all of one player's pegs into the opposite corner.	The user is presented with a dialog announcing the winner, and the option to start a new game, or return to the main menu.		X
20	After task 19, tap the Start New Game button.	The Offline Game Configuration Activity launches.		X
21	After task 19, tap the Main Menu button.	The application returns to the Main Activity.		X
22	Tap the Help and Settings button native to Android, and tap Help.	A sample help page is displayed.		X

Notes

At a resolution of 480x800, on the Offline Game Configuration Activity, the selection buttons for number of players are clipped, and when selecting 6 players, the Start Game button disappears from the bottom of the screen. The Start Game button is clipped when selecting 6 players at 720x1280.

Tapping the Done button without making a move skips the turn.

The user is sometimes prevented from jumping back into their home space.

The user is allowed to jump after a basic move.

The user is occasionally allowed to jump pegs that aren't there.

Triggering a win condition does nothing, and then crashes.

4.5 Manual testing

- See Appendix P

4.6 Ease of use testing

4.61 Ease of use testing tasks

- Task 1: Run the application.
- Task 2: Start a game with 4 players.
- Task 3: Configure but do not start a game with 6 players where
    - Task 3.1: Red is you.
    - Task 3.2: Green is a computer controlled opponent set to medium difficulty
    - Task 3.3: Orange is a computer controlled opponent set to easy difficulty
    - Task 3.4: Purple is a computer controlled opponent set to hard difficulty
    - Task 3.5: Blue and Yellow are both actual players with names of your choosing.
- Task 4: Access the help section from any of the screens.
- Task 5: Start a game with human players. Play a couple turns and close the app.
- Task 6: Resume the game you were previously playing.

- Task 7: Close the app once more and now start a new a game with settings of your choosing and play this game until a player wins.

4.62 Results

- See Appendix Q

5 Appendices

5.1 Appendix A

Description	Owner	Created	Status	Completed	
#44 Use Case Saving Game	@godamnpete	2/7/2014	Done	2/26/2014	
UI dialog box with similar to overlay					
UI mock up	@taywhited				
update existing activity diagram	@kubasub				
code save and load	@kubasub/@goda				
code UI flow control	@kubasub				
Activity testing (which method are we	@saajid				
#42 Use Case Show possible moves	@godamnpete	2/7/2014	Done	2/26/2014	
UI Mock up to show posssible moves	@taywhited				
Code showing possible moves	@chris				
activity diagrams to show the possible	@kubasub				
UML showing the events more like a	@chris				
Manual testing	@saajid				
#43 Use Case Hotseat Game Help	@taywhited	2/7/2014	Done	2/25/2014	
UI Mockup for the help page	@taywhited				
activity diagram	@kuba				
UML changes to show the code design	@chris				
Code and unit test	@kuba				
Automated Activity Tests	@saajid				
split into a general page class	@chris				
#41 Use Case Play Against an AI	@jameskostiuk				
Get AI to make a move	@jameskostiuk				
create the AI interface	@chris				
#61 Use Case to select AI		2/7/2014	Done	2/26/2014	
UI Mock ups	@taywhited				
update game UI flow control	@kuba				
code and unit testing	@kuba				
Activity tests	@saajid				
#40 Use Case - Finish Game	@kubasub	2/7/2014	Done	2/26/2014	
Checking for the win condition	@godamnpete				
UI Mock ups to show who won	@taywhited				Needs the screen
in hotseat tap button to go back to	@kubasub				
#62 Testing - Create Phase 2 Plan	@saajid	2/14/2014	Done	2/25/2014	
#77 Create UI control tables for	@kubasub	2/18/2014	Done	2/20/2014	
#76 Create Activity UI flowchart	@kubasub	2/18/2014	Done	2/20/2014	
#71 Testing - Create acceptance	@saajid	2/14/2014	Done	2/26/2014	
#69 Testing - Gather ease of use	@saajid	2/14/2014	Done	2/26/2014	
#68 Testing - Run tests on physical	@saajid	2/14/2014	Done	2/26/2014	
#66 Testing - implement test plan	@saajid	2/14/2014	Done	2/25/2014	
#65 Testing - implement test plan	@saajid	2/14/2014	Done	2/26/2014	
#64 Testing - Implement test plan	@saajid	2/14/2014	Done	2/25/2014	
#67 Testing - Implement test plan	@saajid	2/14/2014	Done	2/26/2014	
#63 Testing - Implement test plan	@saajid	2/14/2014	Done	2/25/2014	
#58 Construction - Unit testing of	@godamnpete	2/13/2014	Done	2/20/2014	
Design network API	@chriskdon	2/10/2014	Done	2/26/2014	
#52 Create AI interface	@chriskdon	2/10/2014	Done	2/22/2014	
#50 Design Gamestate internals	@chriskdon	2/10/2014	Done	2/26/2014	
#51 Add touch detection to the	@chriskdon	2/10/2014	Done	2/21/2014	
#47 Communication - create end	@benstitt	2/10/2014	Done	2/25/2014	
#75 Streamlining/updating the UI	@kubasub	2/18/2014	Done	2/18/2014	
#46 Construction - finalizing the	@godamnpete	2/7/2014	Done	2/13/2014	
#79 UI Mockups Collection	@taywhited	2/20/2014	Done	2/21/2014	
#84 Construction - Code offline	@kubasub	2/22/2014	Done	2/22/2014	
#81 Add Hint Drawing	@chriskdon	2/21/2014	Done	2/21/2014	
#83 Construction - convert offline	@kubasub	2/22/2014	Done	2/22/2014	
#74 Renaming HOTSEAT gameplay	@kubasub	2/16/2014	Done	2/22/2014	
#80 Construction - Adding a Win	@godamnpete	2/21/2014	Done	2/21/2014	
#38 Use Case - peg animation	@taywhited	2/6/2014	Done	2/25/2014	
#82 Construction - code dialogs	@kubasub	2/22/2014	Done	2/23/2014	

Figure 5.1 Iteration Two progress from February 20, 2014 meeting

5.2 Appendix B

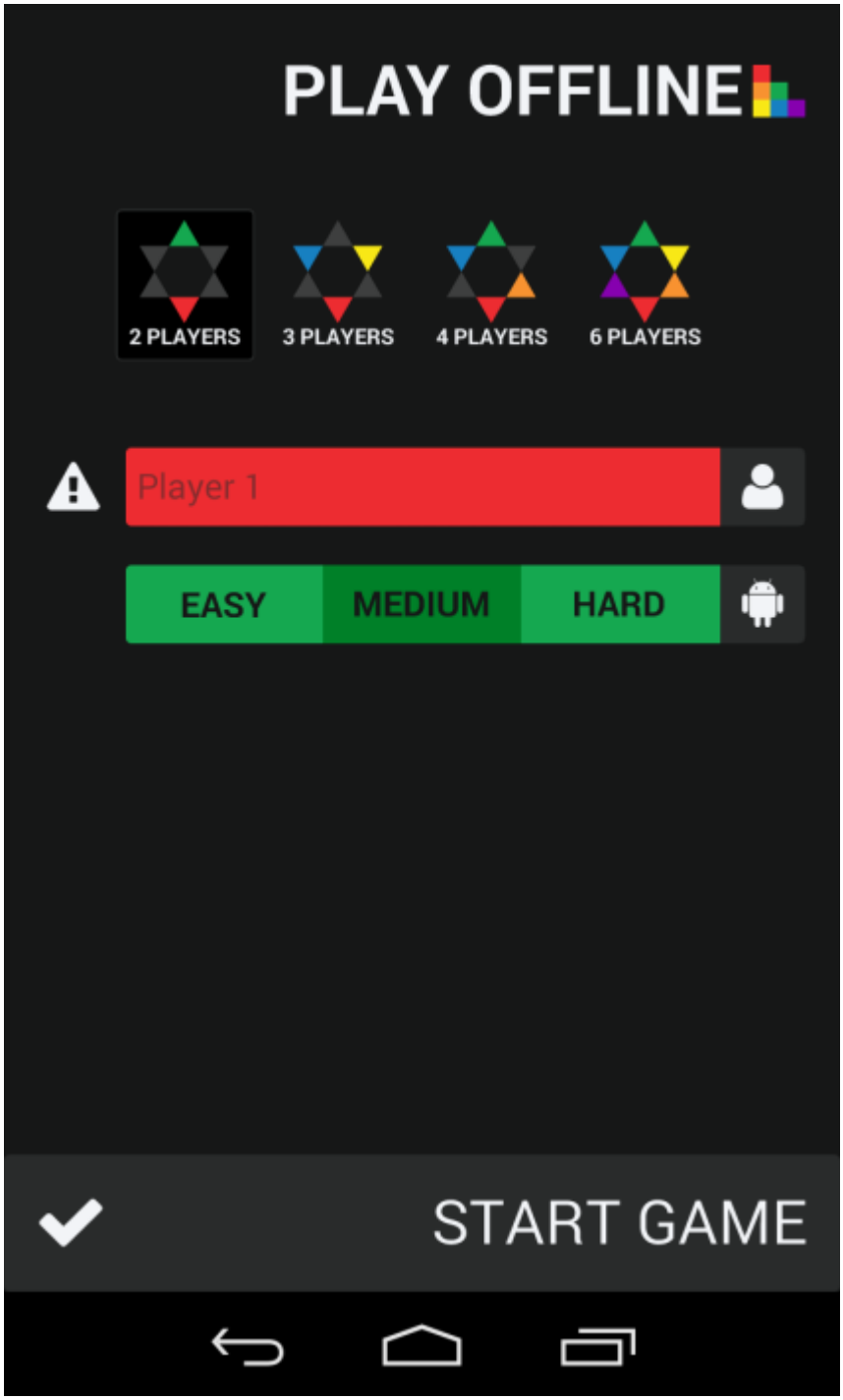


Figure 5.2 Play offline configuration mock-up

5.3 Appendix C

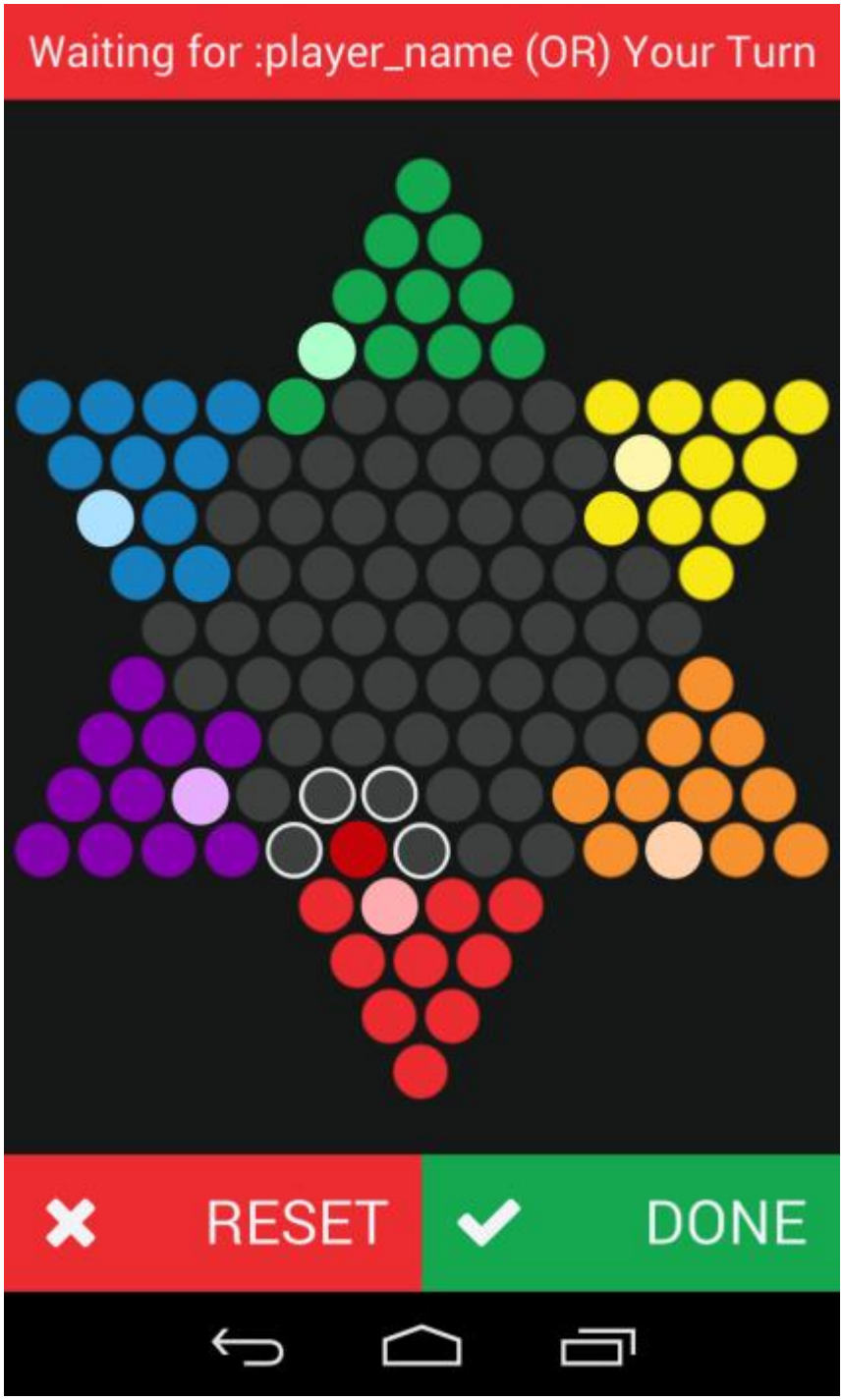


Figure 5.3 Play offline game mock-up

5.4 Appendix D

Figure 5.4 Application dialog mock-up

5.5 Appendix E

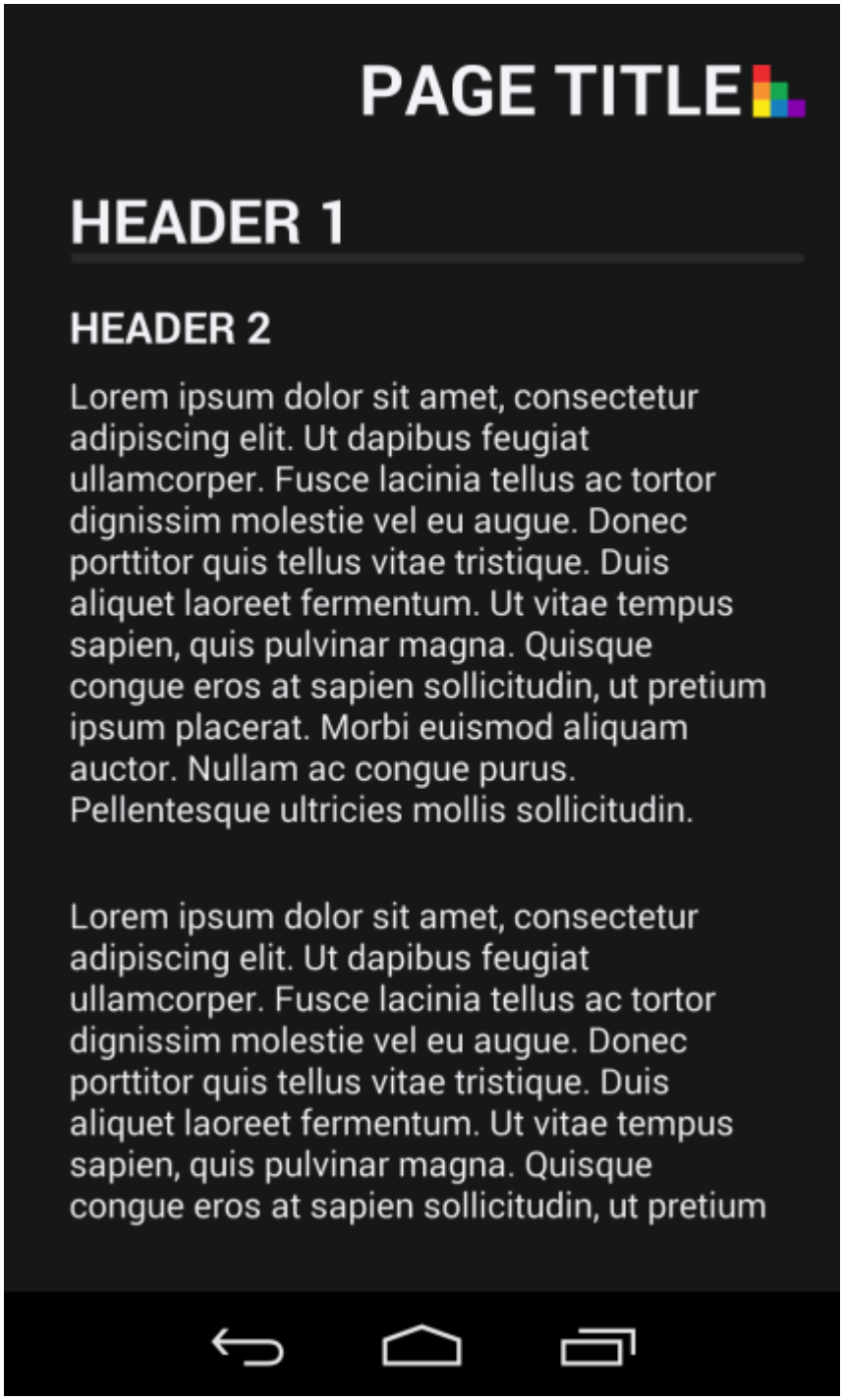


Figure 5.5 Static page mock-up

5.6 Appendix F

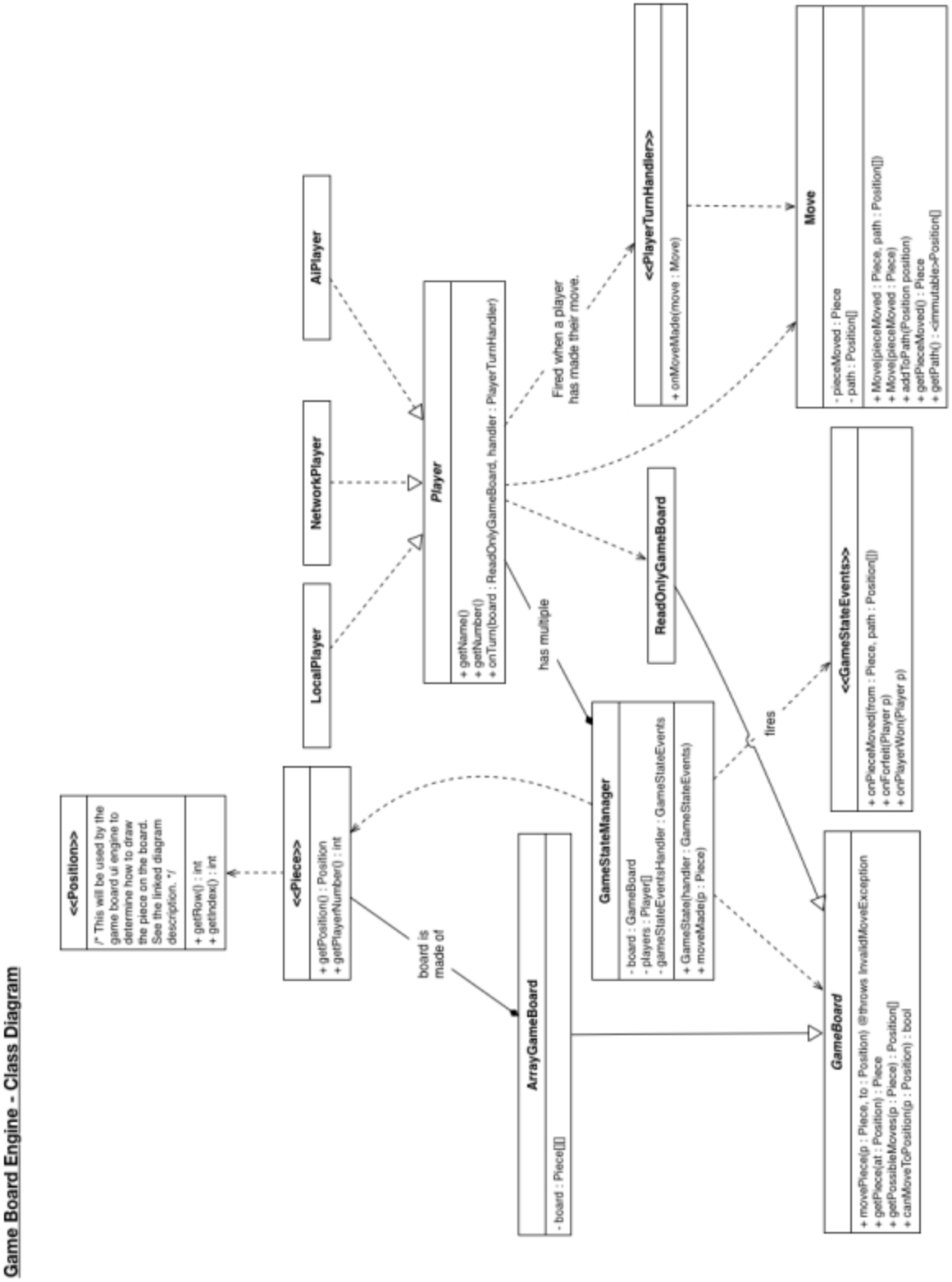


Figure 5.6 Game board engine class diagram

5.7 Appendix G

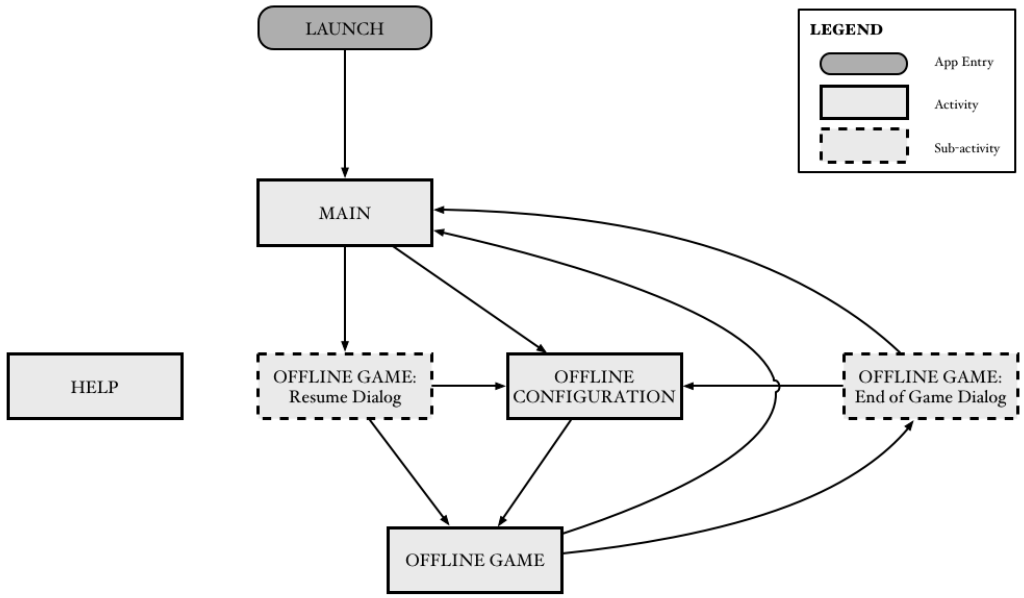


Figure 5.7 Overview of user interface control flow

5.8 Appendix H

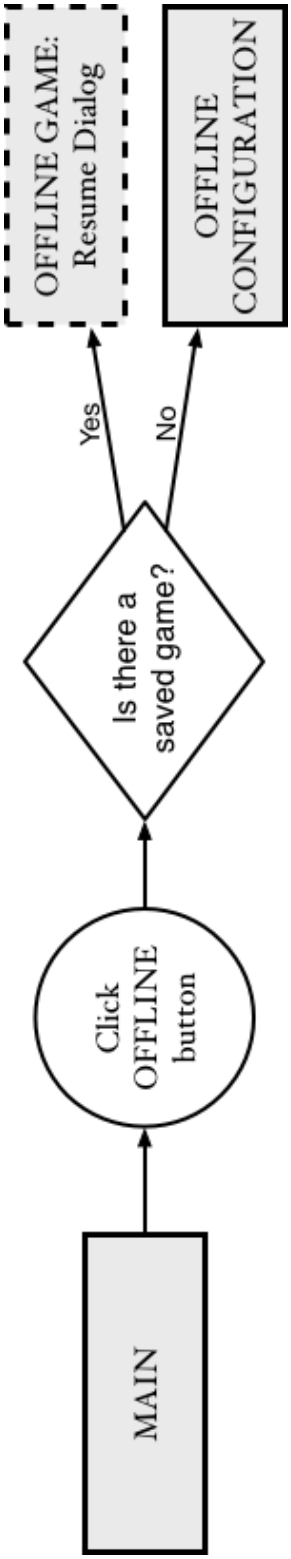


Figure 5.8 Main activity user interface control flow

5.9 Appendix I

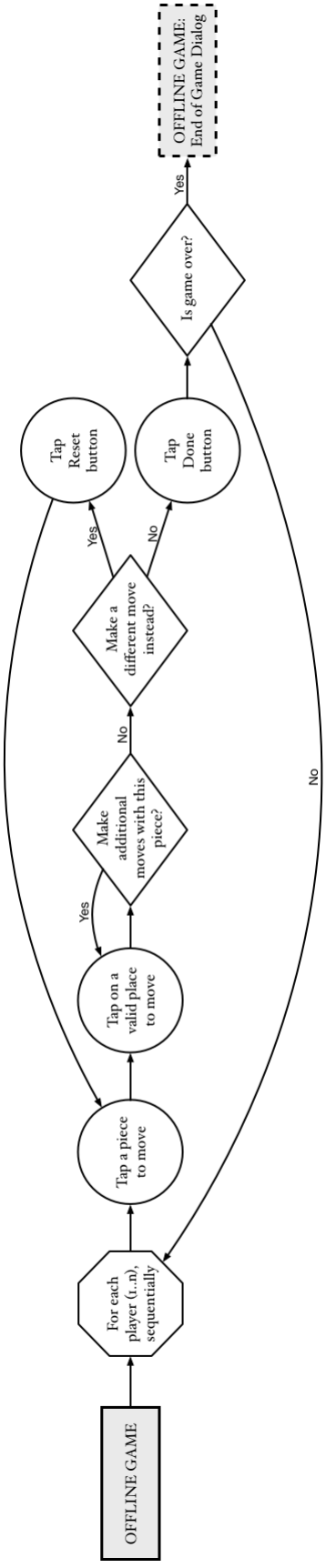


Figure 5.9 Offline game activity user interface control flow

5.10 Appendix J

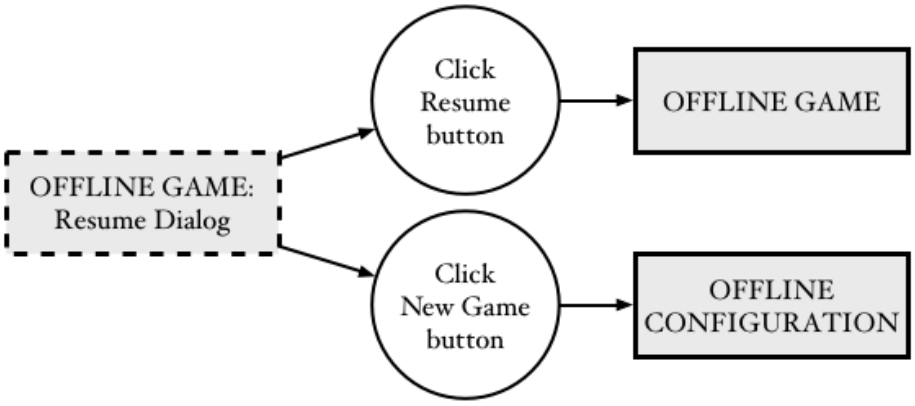


Figure 5.10 Offline game activity: resume dialog user interface control flow

5.11 Appendix K

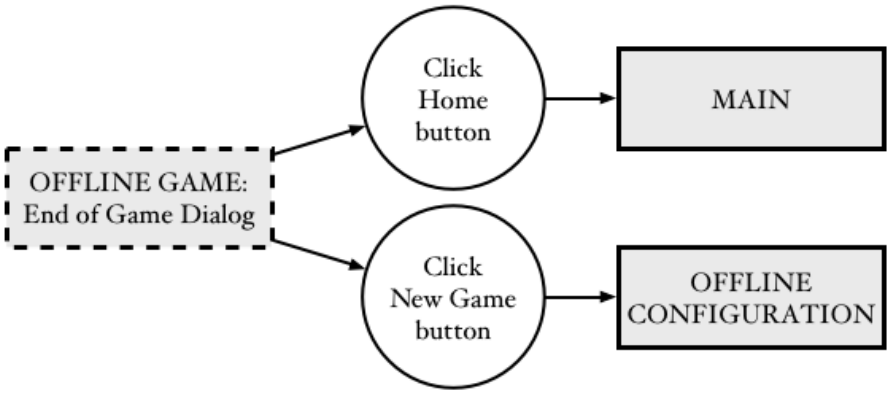


Figure 5.11 Offline game activity: end of game dialog user interface control flow

5.12 Appendix L

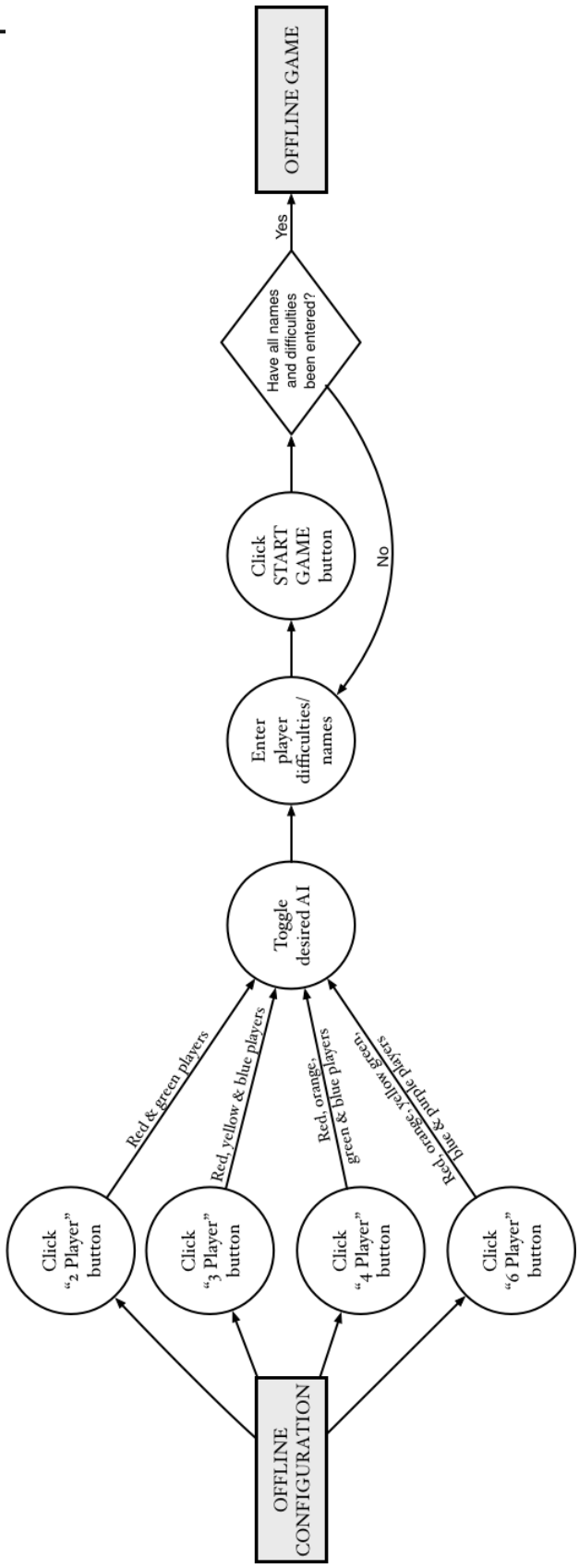


Figure 5.12 Offline configuration activity user interface control flow

5.13 Appendix M

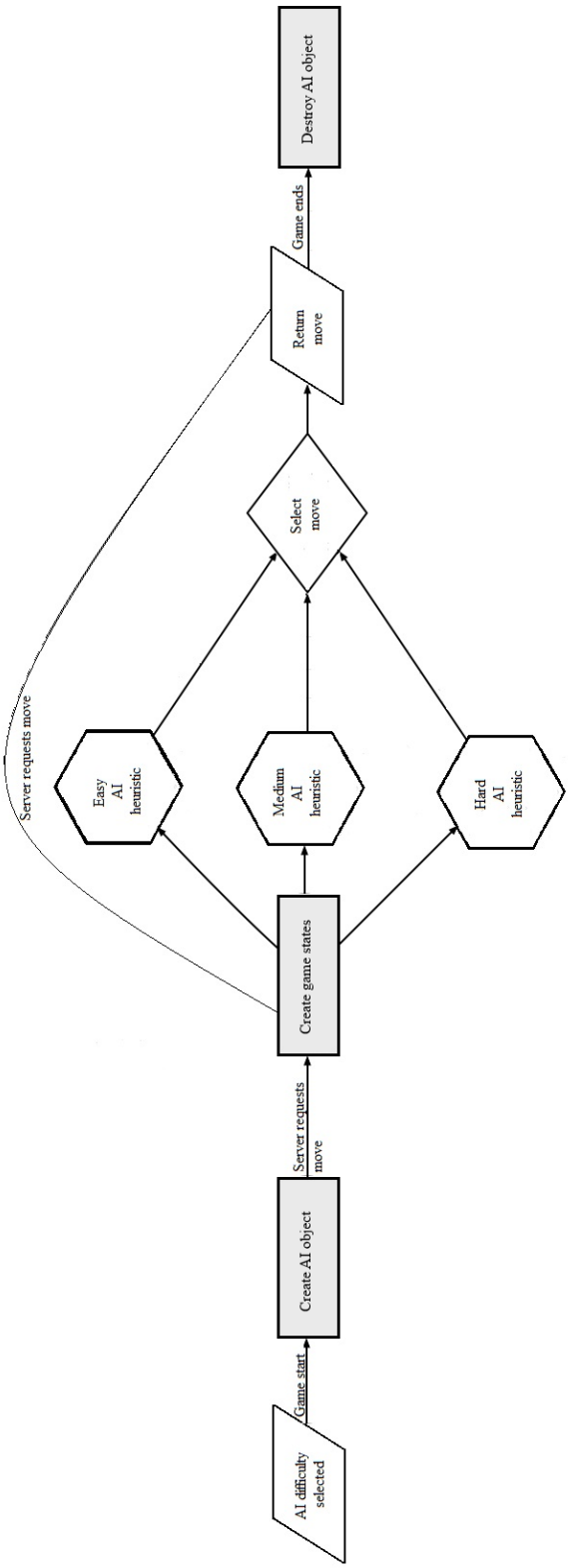


Figure 5.13 AI internal flow diagram

5.14 Appendix N

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: _____

_____

Figure 5.14 Ease of use testing feedback form

5.15 Appendix O

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: _____

Figure 5.15 Test feedback form

5.16 Appendix P

Test feedback form

Project: Chinese Checkers for Android

Project Phase: 2 Test Phase: Complete Application Testing Date: 2/27/2014

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Failed

Remarks (If fail what caused failure):

- Peg Movements are not in line with rules as specified at http://www.mastersgames.com/rules/chinese-checkers-rules.htm
- Errors found include but may not be limited to:
    - Multiple pegs can be moved in a single turn.
    - Player pegs may be moved multiple times even when no hop taken.
    - Player pegs may be moved out of opposite triangle
    - Player hints sometimes give improper options.

Test Output (If Applicable): N/A

Action Taken: To be determined by team .

Tester Signature: _____ Project Lead Signature: _____

Test Lead Signature: _____ Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

Figure 5.16 Manual testing feedback

5.17 Appendix Q

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: 3

Task Remarks: Switching between the computer and real players was confusing.

Task 3.1 Score: 0

Task Remarks: Easier once the controls were figured out

Task 3.2 Score:0

Task Remarks:

Task 3.3 Score: 0

Task Remarks:

Task 3.4 Score: 0

Task Remarks:

Task 3.5 Score: 0

Task Remarks:

Task 3 Overall Score: 0

Task Remarks:

Task 4 Score: 10

Task Remarks: Could not locate the help screen.

Task 5 Score:0

Task Remarks:

Task 6 Score: 10

Task Remarks: Did not ask to resume play

Task 7 Score: 8

Task Remarks: Game worked well. At the end of the game the program crashed though

General Remarks: I liked that the pieces predicted possible movements. It allows for less experienced players to enjoy the app.

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: 5

Task Remarks: Switching between computer and human players not instinctive

Task 3.1 Score: 0

Task Remarks:

Task 3.2 Score: 0

Task Remarks:

Task 3.3 Score: 0

Task Remarks:

Task 3.4 Score: 0

Task Remarks:

Task 3.5 Score:0

Task Remarks:

Task 3 Overall Score: 5

Task Remarks: Same as 2

Task 4 Score: 10

Task Remarks: Could not find

Task 5 Score: 0

Task Remarks:

Task 6 Score: 10

Task Remarks: Could not find

Task 7 Score: 5

Task Remarks: Easy play but game crashed on win.

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: touching an application icon is not hard

_____

____

_____

Task 2 Score: __0___

Task Remarks: _easy to see which game setting option was for the 4 players version of the game. _____

_____

Task 3.1 Score: _1____

Task Remarks: __a bit harder just because I had to type all the letters of my name.

_____

_____

Task 3.2 Score: _1____

Task Remarks: __one click

_____

_____

Task 3.3 Score: _0____

Task Remarks: __no clicks

_____

_____

Task 3.4 Score: _1____

Task Remarks: _one click

_____

_____

Task 3.5 Score:_2____

Task Remarks: _couple extra clicks compared to just picking the robot difficulties

_____

Task 3 Overall Score:__2___

Task Remarks: __6 players required a bit of effort simply because there were more player details to enter. _____

_____

Task 4 Score:__n/a___

Task Remarks: __Could not see the Help menu on our tablet

_____

Task 5 Score:_3____

Task Remarks: _easy to understand what to do

_____

Task 6 Score:_2____

Task Remarks: _easy to understand and not too many clicks

_____

Task 7 Score:__4___

Task Remarks: _pretty easy to play and the hints were helpful to see what moves you could make. Nearing the end of the game when my pegs were in the opposing players end it was difficult to see the hints on our tablet because the colouring was similar.

_____

_____

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: _1____

Task Remarks: _Not hard to touch the screen one time to launch the game

_____

_____

Task 2 Score: _2____

Task Remarks: __Was pretty easy to pick 4 player game and set them up

_____

_____

Task 3.1 Score: _3____

Task Remarks: __I just had to type my name

_____

_____

Task 3.2 Score: _1____

Task Remarks: ___easy to click medium

_____

_____

Task 3.3 Score: _0____

Task Remarks: __didn't have to do anything

_____

_____

Task 3.4 Score: __1___

Task Remarks: __just one click to make the robot smart

_____

_____

Task 3.5 Score:_4____

Task Remarks: __just had to click a couple extra times to set up real people instead of robots

_____

Task 3 Overall Score:_3____

Task Remarks: _I liked all the different colours.

_____

_____

Task 4 Score:_n/a____

Task Remarks: __Cannot see help on my tablet

_____

Task 5 Score:__4___

Task Remarks: __Pretty easy and cool. I like how it showed me how to play and when I could hop over more than one time

_____

Task 6 Score:_2____

Task Remarks: _One touch to pick that you want to continue is not hard

_____

Task 7 Score:_4____

Task Remarks: _The game takes kind of long but it was fun

_____

_____

General Remarks: _____

_____

5.18 Appendix R

Test feedback form

Project: Chinese Checkers for Android

Project Phase: 2 Test Phase: UI Unit and Integration Date: 2/27/2014

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Passed

Remarks (If fail what caused failure): None

Test Output (If Applicable): Attached

Action Taken: N/A

Tester Signature: _____ Project Lead Signature: _____

Test Lead Signature: _____ Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

5.19 Appendix S

Package ca.brocku.chinesecheckers.tests

all > ca.brocku.chinesecheckers.tests

tests	failures	duration	successful
23	0	1m42.51s	100%

- Classes

Classes

Class	Tests	Failures	Duration	Success rate
GridUnitTest	18	0	0.725s	100%
MainActivityUnitTest	1	0	7.351s	100%
MainAndConfigAndGameIntegrationTest	1	0	43.402s	100%
MainAndConfigIntegrationTest	1	0	29.152s	100%
OfflineConfigurationActivityUnitTest	1	0	15.201s	100%
OfflineGameActivityUnitTest	1	0	6.675s	100%

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.20 Appendix T

Class ca.brocku.chinesecheckers.tests.GridUnitTest

all > ca.brocku.chinesecheckers.tests > GridUnitTest

tests	failures	duration	successful
18	0	0.725s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testAccuracyOfGetPossibleMoves	passed (0.025s)
testAndroidTestCaseSetupProperly	passed (0.050s)
testBoardCreate	passed (0s)
testCorrectNumberOfPiecesReturend	passed (0.050s)
testCreateBoardWithValidNumberOfPlayers	passed (0.050s)
testDoubleSet	passed (0s)
testGetAllPiecesNotNull	passed (0.025s)
testGetPieceEmpty	passed (0.025s)
testGetPiecesValid	passed (0.025s)
testInvalidMove	passed (0.050s)
testMoveOnExistingPiece	passed (0.025s)
testMovePieceValid	passed (0.025s)
testOutOfBoundsPiece	passed (0.025s)
testPossibleMovesCountCorners	passed (0.275s)
testPossibleMovesForCenterPiece	passed (0.025s)
testSetOutOfBounds	passed (0.050s)
testSetandGetPiece	passed (0s)
testValidMoves	passed (0s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.21 Appendix U

Class ca.brocku.chinesecheckers.tests.MainActivityUnitTest

all > ca.brocku.chinesecheckers.tests > MainActivityUnitTest

tests	failures	duration	successful
1	0	7.351s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testActivity	passed (7.351s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.22 Appendix V

Class ca.brocku.chinesecheckers.tests.MainAndConfigAndGameIntegrationTest

all > ca.brocku.chinesecheckers.tests > MainAndConfigAndGameIntegrationTest

tests	failures	duration	successful
1	0	43.402s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testActivity	passed (43.402s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.23 Appendix W

Class ca.brocku.chinesecheckers.tests.MainAndConfigIntegrationTest

all > ca.brocku.chinesecheckers.tests > MainAndConfigIntegrationTest

tests	failures	duration	successful
1	0	29.152s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testActivity	passed (29.152s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.24 Appendix X

Class ca.brocku.chinesecheckers.tests.OfflineConfigurationActivityUnitTest

all > ca.brocku.chinesecheckers.tests > OfflineConfigurationActivityUnitTest

tests	failures	duration	successful
1	0	15.201s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testActivity	passed (15.201s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.25 Appendix Y

Class ca.brocku.chinesecheckers.tests.OfflineGameActivityUnitTest

all > ca.brocku.chinesecheckers.tests > OfflineGameActivityUnitTest

tests	failures	duration	successful
1	0	6.675s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testActivity	passed (6.675s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.26 Appendix Z

Package ca.brocku.chinesecheckers.tests.uiengine

all > ca.brocku.chinesecheckers.tests.uiengine

tests	failures	duration	successful
10	0	0.450s	100%

- Classes

Classes

Class	Tests	Failures	Duration	Success rate
DimensionsTest	2	0	0.075s	100%
PieceDrawingDetailsTest	2	0	0.100s	100%
PiecePositionSystemTest	3	0	0.150s	100%
PixelPositionTest	3	0	0.125s	100%

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.27 Appendix AA

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

all > ca.brocku.chinesecheckers.tests.uiengine > DimensionsTest

tests	failures	duration	successful
2	0	0.075s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testGetters	passed (0.025s)
testSetters	passed (0.050s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.28 Appendix AB

Class ca.brocku.chinesecheckers.tests.uiengine.PieceDrawingDetailsTest

all > ca.brocku.chinesecheckers.tests.uiengine > PieceDrawingDetailsTest

tests	failures	duration	successful
2	0	0.100s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testGetterPosition	passed (0.050s)
testGetterRadius	passed (0.050s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.29 Appendix AC

Class ca.brocku.chinesecheckers.tests.uiengine.PiecePositionSystemTest

all > ca.brocku.chinesecheckers.tests.uiengine > PiecePositionSystemTest

tests	failures	duration	successful
3	0	0.150s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testPositionXyCalculation	passed (0.075s)
testUnmodifiablePositionList	passed (0.025s)
testUnmodifiablePositionMapping	passed (0.050s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.30 Appendix AD

Class ca.brocku.chinesecheckers.tests.uiengine.PixelPositionTest

all > ca.brocku.chinesecheckers.tests.uiengine > PixelPositionTest

tests	failures	duration	successful
3	0	0.125s	100%

- Tests

Tests

Test	AVD_for_Nexus_S_by_Google(AVD) - 4.4.2
testGettersXY	passed (0.025s)
testOffset	passed (0.050s)
testSettersXY	passed (0.050s)

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

5.31 Appendix AE

Test Summary

tests	failures	duration	successful
33	0	1m42.96s	100%

- Packages
- Classes

Packages

Package	Tests	Failures	Duration	Success rate
ca.brocku.chinesecheckers.tests	23	0	1m42.51s	100%
ca.brocku.chinesecheckers.tests.uiengine	10	0	0.450s	100%

Classes

Class	Tests	Failures	Duration	Success rate
ca.brocku.chinesecheckers.tests.GridUnitTest	18	0	0.725s	100%
ca.brocku.chinesecheckers.tests.MainActivityUnitTest	1	0	7.351s	100%
ca.brocku.chinesecheckers.tests.MainAndConfigAndGameIntegrationTest	1	0	43.402s	100%
ca.brocku.chinesecheckers.tests.MainAndConfigIntegrationTest	1	0	29.152s	100%
ca.brocku.chinesecheckers.tests.OfflineConfigurationActivityUnitTest	1	0	15.201s	100%
ca.brocku.chinesecheckers.tests.OfflineGameActivityUnitTest	1	0	6.675s	100%
ca.brocku.chinesecheckers.tests.uiengine.DimensionsTest	2	0	0.075s	100%
ca.brocku.chinesecheckers.tests.uiengine.PieceDrawingDetailsTest	2	0	0.100s	100%
ca.brocku.chinesecheckers.tests.uiengine.PiecePositionSystemTest	3	0	0.150s	100%
ca.brocku.chinesecheckers.tests.uiengine.PixelPositionTest	3	0	0.125s	100%

Generated by Gradle 1.10 at Feb 27, 2014 10:31:04 AM

6 References

Sample Template for Acceptance Testing. (2009, October 21). *myLRIG*. Retrieved February 1, 2014, from http://my.lrig.org/Communities/Resources/ViewDocument/?DocumentKey=81f2ae1f-8a4a-4276-8751-1ecff0fabe0a

Use Case Templates. (n.d.). *TechnoSolutions*. Retrieved January 18, 2014, from http://www.technosolutions.com/use_case_template.html