Logical Operators: Chinese Checkers

Iteration three

March 20th, 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

Due to an absence, the fourth iteration of the iteration level plan has been extended by one week by unanimous decision of the group. This change adds 48 hours to the total hours allocated and extends the deadline to April 3, 2014. The number of work days has increased from 77 days to 82 days and work weeks have increased from 11 weeks to 12 weeks.

2.11 Release level plan

Start	9-Jan
Finish	3-Apr
Last day of class	4-Apr
Work days	82
Work weeks	12
# of people	8
Hours/Week/Person	6
Total hours	576

2.12 Iteration level plan

	# Weeks	**Hours**
1st iteration	4	192
2nd iteration	3	144
3rd iteration	3	144
4th iteration	2	96
	Total hours	576

2.2 Meeting minutes

2.21 February 27, 2014

The group discussed issues with the current code, including determining possible moves. In addition, the help and settings menu was not accessible from all devices. The latter issue could be solved by adding a button within the application.

The group then presented to Vlad, including a slide show, and a demonstration on two devices. Vlad made the suggestion that the Reset button be changed to an undo button, which would only revert the move by one hop.

The group then closed the remaining iteration 2 issues left open.

The group then shared ideas to streamline future progress, such as setting smaller weekly goals, posting more tasks at the beginning of the iteration, and tagging team members for individual tasks.

Testing needed to become more efficient, so Saajid volunteered to look into automation of tasks which had previously been done manually.

More ideas to streamline work were shared, such as staggering design tasks, so some could begin before the previous iteration had finished, and the group could work on fewer modules at once. However, since the project is entering its final design iteration, this suggestion has limited merit.

It was decided that documentation should be uploaded to the master branch, and all other files should be uploaded to other branches first. Tasks should always be assigned to group members. It was also decided that features should be referred to as stories to differentiate them from formally written use cases.

The group reviewed the viability of features from the original project plan. The group decided not to include user chat or friends features, and will probably not include peg animation. The group decided to include anonymous and Facebook accounts, and to ensure COPPA compliance in the EULA. Only untimed online games will be available, but they will kick a player out after one week of inactivity, replacing them with an AI player.

The group then started to establish details on stories. The group briefly discussed what sort of tables should be included in the database.

It was decided that the Facebook connect button should be displayed in the settings menu.

The server should be based on Java, communicate via Node.js, and store data using MySQL. Heroku, a Git repository platform could be used on the server, and Mocha could be used for testing JavaScript code.

2.22 March 4, 2014

The group used this meeting to expand on previously discussed features, and establish user stories.

The discussion began with a review of previously established stories, namely anonymous accounts and Facebook accounts.

There was also the discussion of COPPA compliance in the EULA. It has been taken care of, but a project task should be established and closed on GitHub in case further action is required.

Next, the help section was discussed. It was assigned to Ben. It is to be written in the style of a FAQ, in plain text and converted to XML. Topics covered should include game configuration including watching a demo game, rules including when a game ends, and help for online games including how to quit. If leader boards are included in the game, the help section should mention how to accrue points and their purpose. Ben will research the help pages of existing Android applications.

It was decided that the application webpage should be left until the end, and completed if time allows.

To toggle the display of possible moves, a toggle switch will be placed in the settings.

The discussion turned to push game turn notification. The Android device would need to run a listener service for notifications pushed by the server. The application must update the online games when a new notification is received, and notifications should be displayed outside of the application. Online game listings must also be updated. Updated games should have an icon to indicate this is the case. This should also be true of the online game button on the main menu. Group members will research creating notifications with Google Cloud Messaging.

The discussion then turned to creating multiple online games, displayed in a list. These games are to be created using a matchmaking service. The user will tap a button to start a new game, and will then be presented with a dialog box, allowing the user to select the number of players. If enough users are not found within a certain fixed length of time, the remaining slots will be filled with AI players of medium or random difficulty. Games will be stored in a database. An ER diagram will be created to model the necessary tables.

Next, the group discussed cases in which an AI player would be inserted into an online game. If a newly created game times out, the remaining slots will be filled with AI players. If a player quits an online game, they will be replaced with an AI player.

Lastly, the group discussed the possibility of a user quitting an online game. This could be accomplished with a long press on a game on the list of active games, which would open a dialog box, prompting the user to delete the game.

The group decided not to include the leader board in this iteration.

2.23 March 6, 2014

The group discussed concerns with the development of the application. The issues included AI integration, whether to include the list of online games, and the function of the game state manager.

Ideally, the game state manager should treat local, AI, and network players the same way, but doing so may be problematic.

Facebook integration was discussed next. It requires Gradle version 1.9 or higher.

AI was discussed next. Progress up to this point is stored in a GitHub branch called AI_dev. A delay should be inserted so the user can see each hop the AI player makes. James will try to integrate the easy level AI over the weekend, and get the other levels working later on.

Networking was discussed next. It was decided that Node.js need not be used, and that Java alone would suffice.

The group considered removing the online game list, but decided that it would not take much more trouble to include it.

The group also discussed Android notifications.

The database was discussed next. It will contain a users table, which should include a unique ID number as a primary key, a unique username, a device ID, and a Facebook ID. It will also contain a games table, which should contain a game ID, the number of users, the users, which are sorted by number, verified by a check constraint, as well as the current turn, and a winner ID. It will also contain a table to join these two, as well as a pieces table, which should contain a piece ID, a row index, a game ID, and a user ID.

The rotation of the game board was also discussed, and it was decided that users' turns should rotate clockwise, while the game board rotates counter-clockwise.

The group decided that individual hops do not need to be shown for network games. Also, a network game should end when the first player wins, as verified by the server.

Unique medium level AI players should be generated in the players table as needed and destroyed when the game ends.

In order to determine when the creation of a network game times out, some type of scheduling has to occur on the server. This can be accomplished using the MySQL event scheduler, polling, or a cron job.

At this point, the group switched rooms.

For server development, Vagrant can be used to test locally. The Java Play framework will be used to program it.

SQL constraints can be tested. So can the AI once it is linked up.

The possible moves are difficult to see inside the home base, so the group decided to change the colour of the border to that of the peg.

The group then set some goals for Tuesday, including AI integration and the database. Facebook linking has been put off, and will only be included if time permits at the end of the iteration.

The application must be able to check for the presence of an Internet connection. If a connection is not found, a banner will be displayed to inform the user in each activity other than offline and help.

The group then reviewed issues on GitHub, establishing time estimates.

Ben was assigned to looked into the Android REST interface, and will complete the help section text in a FAQ style without headers.

2.24 March 11, 2014

At least one level of AI should be done tonight.

Design for push notification and networking has been done.

Matchmaking can be done by the database only. A stored procedure in SQL could handle the cleanup for online games that have been populated completely by AI players. Constraints for the number of players are in place.

The help FAQ was done, but needed revision.

Ben had studied Android and REST services in order to aid with coding.

The group discussed adjusting win conditions, and decided to trigger a win if a player's goal spaces are full and they have at least one there. This decision should not affect AI.

The current rules allow for one slide movement, followed by any number of jumps if necessary. The AI and help text should be updated to reflect this rule.

If an opponent blocks a player's goal spaces with a wall two spaces wide, winning becomes impossible. This issue was put off for the time being.

The username should be generated as a forty character GUID on the device automatically without prompting the user. This should be clarified in the help section.

Dialog boxes are the same colour as a dimmed background, so a white outline should be added.

The group then discussed outstanding tasks, including the GCM notification pusher, server logic, and push notifications.

When a player's turn comes in an online game, a symbol should be shown next to the Online button. This was originally going to be an exclamation point in a circle, but the group decided to use an arrow instead.

The group discussed what the REST API should return for different requests.

When listing games, the API must return the game ID, whether there is a winner, the winner's user ID and username, the current player number, as well as the player number of the player who requested it, indicated by a star.

When starting a new game, the API should send the requester's user ID and the number of player's. A push notification should be received when the game is started, and the games list should be retrieved again.

When leaving a game, the API should send the game ID and the user ID.

When loading a game, the API should return the game ID, positions of game pieces, a list of players' names and positions, and the current player.

When sending a move by tapping the Done button, the API should send the game ID, and the original position and new position of the piece moved. The database should then increment the current player.

When another player makes a move, the API should return the game ID, as well as the original position and new position of the piece moved.

In the case that another player wins, the API should return the username of the player who won.

Network errors were also discussed. In the case that such an error occurs, the user should be notified that the task could not be completed.

2.25 March 13, 2014

The group discussed existing errors in the application. Firstly, the AI players do not return a path if no advantageous move is available. Starting a game with only AI players crashes the application as well. A stack overflow error is also produced by AI players in the Game State Manager later in a game due to circular recursion.

A delay should also be added to local AI movement so the user can see the move being made.

An object library should be created for game board and player objects on both the Android application and the server. A set of wrapper classes can be added for Android to allow the objects to be parcelable.

For this iteration, a player should be able to register on the server.

The group discussed pushing back iteration deadlines, and decided to ask for a one-week extension for iteration four. This gives the group until April 3rd. With the AI working, there should be enough to present this iteration to Vlad.

It was decided that a network activity class should be created.

The players are allowed to move game pieces out of the goal space. The group discussed preventing this.

The ER diagrams and SQL naming for the database should be adjusted.

The group discussed what should be done by the end of this iteration. The easy level AI should be working with a sleep put in. The object library with parcelable wrapper classes should also be done. The application should be able to detect an Internet connection, and display a banner when no connection is present. Help text needs to be put in. Notifications for the user's turn should be done, and an arrow informing the user should be placed on the main screen when it is their turn. Testing should also be done, but a working AI is required.

Peter will research information for a notifications badge.

Curtis will return on March 17, and the group will meet with Vlad on March 20.

2.26 March 18, 2014

The group decided that the easy level AI is too difficult, and will need adjustment.

The group discussed whether they were ready to present to Vlad on Thursday.

The easy AI was working, but the medium AI should be done by then. The hard button will still default to easy.

The group would like the ability to register players with the server to be ready, but it may not work out. The group would like connectivity to be done. The database should be up by then as well.

The group would like to demonstrate their progress by showing an AI game on the TV.

Curt planned to send an email to Vlad requesting a one-week extension for the forth iteration.

Peter should be informed of the change in win condition, and the restriction preventing movement out of the end zone.

ST238 could not be secured for Thursday, so ST237A was booked instead from 5 to 7.

Ease of use and acceptance testing needs to be done for Thursday as well.

The group then discussed open tasks on GitHub.

The online architecture is done.

The umbrella task for playing against an AI can be closed, and the hard mode can be pushed back to the forth iteration.

Board rotation will also be pushed back, possibly for the forth iteration, or later.

Push notifications will also be pushed back to the forth iteration.

The database is created, and just needs to be tested.

Making the AI code runnable on the server can also be pushed back to the forth iteration.

Registering a user with the server should be possible by Thursday.

The presentation on Thursday should follow a format similar to those for the previous iterations. The group should explain the breakdown of iterations into sprints, explain modifications of the game rules, and explain their goals for iteration four. The group should also provide a demonstration, including a six-player game and the online list.

A bug was brought up. If a user quickly switches between an active Chinese checkers game and another application may cause the board to return to the screen, but not populated. This was not considered a priority.

Another bug was discussed. When an offline game is resumed, the AI players continued to take their turns while the resume dialog was displayed. This was fixed by populating the board after the user chooses to resume.

The group discussed publishing the application, and decided it would take too long to do before the end of the class, but it could be done later, if it were more polished.

2.3 Use cases

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

2.31 List online games (IT-3-UC-12)

Revision history

Date	Author	Description of change
2014/03/12	Ben Stitt	Authored use case.

Use case

List online games

Id

IT-3-UC-12

Description

The user wishes to check the status of any online games in which they are involved. The user taps the Online button, starting the Online List Activity.

Level

User goal

Primary actor

The user wishes to see their active games or start a new one.

Stakeholders and interests

The user requires information on online games, and the other players wish the user to take their turn when it comes in order to continue game play.

Pre-conditions

The user must have the application installed and running, and the user's device must be connected to the Internet.

Post conditions

Success end condition

The Online List Activity starts and the list of active games is retrieved from the server.

Failure end condition

The Online List Activity fails to load, or the list cannot be retrieved from the server.

Minimal Guarantee

The application does not crash, and data on the device remains unaffected. Existing online games remain unaffected as well.

Trigger

The user taps the Online button on the main menu.

Main success scenario

1. The Online List Activity is started.
2. A request is sent to the server for games in which the user is participating.
3. The application receives a response from the server containing a list of games.
4. The screen is updated to list each game.

Extensions

3. No response is retrieved from the server.
4. The user is informed of the error.

Frequency

This use case is triggered each time the Online button is tapped. As such, frequency depends on the user, but it may occur as often as several times daily, minutes apart.

Assumptions

The user wishes to play online.

Special requirements

The device must run Android and have the necessary space to install the game. The device must be capable of accessing the Internet.

Issues

1. Is the network service operational enough to test it successfully?

To do

1. Have the network service running.

2.32 Start an online game (IT-3-UC-13)

Revision history

Date	Author	Description of change
2014/03/12	Ben Stitt	Authored use case.

Use case

Start an online game

Id

IT-3-UC-13

Description

The user wishes to start an online game. They do so by tapping the New Game button in the Online List Activity.

Level

User goal

Primary actor

The user wishes to play against online opponents.

Supporting actors

Other users may be completing this action at the same time, and may be matched with this user.

Stakeholders and interests

The users who wish to play online would like to be matched with each other, and if necessary, AI players.

Pre-conditions

The user must have the application installed and running, and the user’s device must be connected to the Internet.

Post conditions

Success end condition

The user is added to a current matchmaking session, and a game is started.

Failure end condition

The user is not added to the new game.

Minimal Guarantee

The application does not crash, and data on the device remains unaffected. Existing online games remain unaffected as well.

Trigger

The user taps the New Game button in the Online List Activity.

Main success scenario

1. The user is prompted for the number of players they wish to include in their game.
2. The user taps the number of players they wish to include.
3. A request is sent to the server to include the user in a new game with the selected number of players.
4. Once the selected number of players has been reached, the game is started, and notifications are sent to the participating players.
5. The online games list is updated to include the new game.

Extensions

4. If not enough players are found after a brief length of time, the remaining player slots are filled by AI players, the game is started, and notifications are sent to the participating players.
5. The online games list is updated to include the new game.

Frequency

This use case occurs each time a player chooses to start a new online game. As such, frequency depends on the user, but it may occur as often as several times daily, minutes apart.

Assumptions

The user wishes to play online.

Special requirements

The device must run Android and have the necessary space to install the game. The device must be capable of accessing the Internet.

Issues

1. Is the network service operational enough to test it successfully?

To do

1. Have the network service running.

2.33 Leave an online game (IT-3-UC-14)

Revision history

Date	Author	Description of change
2014/03/12	Ben Stitt	Authored use case.

Use case

Leave an online game

Id

IT-3-UC-14

Description

The user is removed from an online game.

Level

User goal

Primary actor

The user wishes to be removed from an online game.

Stakeholders and interests

Other players in the game may wish to continue play with or without the user.

Pre-conditions

The user must have the application installed and running, and the user’s device must be connected to the Internet.

Post conditions

Success end condition

The user is removed from the game, and replaced by an AI player. The game is also removed from the user’s list of online games.

Failure end condition

The user is not removed from the game, or the game is not removed from the user’s list of online games.

Minimal Guarantee

The application does not crash, and data on the device remains unaffected. Existing online games remain unaffected as well.

Trigger

The user presses and holds the game in the Online List Activity.

Main success scenario

1. The user is prompted whether to leave the game.
2. The user taps to confirm.
3. A request is sent to the server to remove the user from the selected game.
4. The server removes the player from the records associated with the game.
5. The game is removed from the user’s list of online games.

Extensions

4. If there are no human players remaining in the game, the server removes the game record.

Frequency

This use case occurs whenever a user leaves a game. As such, it should occur rarely, most likely less than once a week.

Assumptions

The user wishes to leave an online game.

Special requirements

The device must run Android and have the necessary space to install the game. The device must be capable of accessing the Internet.

Issues

1. Is the network service operational enough to test it successfully?

To do

1. Have the network service running.
2. Have the games list operational.

2.34 View settings (IT-3-UC-15)

Revision history

Date	Author	Description of change
2014/03/13	Ben Stitt	Authored use case.

Use case

View settings

Id

IT-3-UC-15

Description

The settings screen is displayed for the user.

Level

User goal

Primary actor

The user wishes to view the current settings for Chinese checkers on their device.

Pre-conditions

The user must have the application installed and running.

Post conditions

Success end condition
The settings screen and the individual controls are displayed.

Failure end condition
The settings screen is not displayed or control widgets do not appear or are not responsive.

Minimal Guarantee

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

Trigger

The user taps the Settings button on the main menu.

Main success scenario

1. The Settings Activity is started.
2. The settings screen and the associated widgets are displayed.

Extensions

2. If no Internet connection is present, the username widget does not allow interaction.

Frequency

This use case occurs only when a user wishes to view or change settings. As such, a user will access it less and less as they become more familiar with the application, likely months apart.

Assumptions

The user is capable of deciphering the purpose of each widget.

Special requirements

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

Issues

1. Are the individual widgets operational?

To do

1. Create toggle for possible moves.
2. Create text entry and network service for changing username.

2.35 Toggle hints (IT-3-UC-15-SF-1)

Revision history

Date	Author	Description of change
2014/03/13	Ben Stitt	Authored use case.

Use case

Toggle hints

Id

IT-3-UC-15-SF-1

Description

Possible move hints are switched off or on.

Level

Sub-function

Primary actor

The user has decided to switch the possible moves hints on or off.

Pre-conditions

The user must have the application installed and running, and must have the Settings Activity running.

Post conditions

Success end condition

A value determining whether to show possible moves is updated on the device, and the widget is updated.

Failure end condition

The value is not updated, or the widget is not updated.

Minimal Guarantee

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

Trigger

The user taps the inactive portion of the hints widget.

Main success scenario

1. A value determining whether to show possible moves is updated on the device.
2. The widget is updated.

Frequency

This use case occurs only when this setting is changed. As such, it should only occur a few times, months apart.

Assumptions

The user is aware of their skill level, and is able to determine whether or not they require hints.

Special requirements

The device must run Android and have the necessary space to install the game. The device must allow the application to write the value.

To do

1. Implement a value to determine this setting

2.36 Update username (IT-3-UC-15-SF-2)

Revision history

Date	Author	Description of change
2014/03/13	Ben Stitt	Authored use case.

Use case

Update username

Id

IT-3-UC-15-SF-2

Description

The username is updated in the text field and on the server.

Level

Sub-function

Primary actor

The user wishes to change their username.

Pre-conditions

The user must have the application installed and running, and must have the Settings Activity running. The device must also be connected to the Internet.

Post conditions

Success end condition

The username is updated on the device and on the server.

Failure end condition
The username is not updated on the device or the server.

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

Trigger

The user changes the text in the username field.

Main success scenario

1. A request is sent to the server to change the user’s username.
2. The user’s username is updated on the server.
3. The text in the username field is updated.

Frequency

This use case occurs only when the user wishes to change their username. As such, it may only occur a few times, months apart.

Assumptions

The user wishes to change their username, and has decided what to change it to.

Special requirements

The device must run Android and have the necessary space to install the game. The device must also be connected to the Internet.

Issues

1. Is the network service operational enough to test it successfully?

To do

1. Have the network service running.

2.37 Load an online game (IT-3-UC-16)

Revision history

Date	Author	Description of change
2014/03/16	Ben Stitt	Authored use case.

Use case

Load an online game

Id

IT-3-UC-16

Description

A complete game state is retrieved from the server, and the user is able to see the current game board.

Level

User goal

Primary actor

The user wishes to see the current state of an online game, and be able to play a move if it is their turn.

Stakeholders and interests

Other players may be waiting on the user to take their turn.

Pre-conditions

The user must have the application installed and running, and the user’s device must be connected to the Internet.

Post conditions

Success end condition

The game board is displayed, and if it is the user's turn, they may make a move.

Failure end condition

The game state is not retrieved, the game board is not displayed, or the user is unable to make a move on their turn.

Minimal Guarantee

The application does not crash, and data on the device remains unaffected. Existing online games remain unaffected as well.

Trigger

The user taps an active game in their list of online games.

Main success scenario

1. A request is sent to the server to retrieve to game state.
2. The server retrieves the game state from the database, and returns it.
3. The application displays the current state of the game board.

Extensions

3. If it is the user's turn, they may also submit a move.

Frequency

This use case occurs whenever a user checks the state of an online game. as such, it may occur several times daily, minutes apart.

Assumptions

The user is able to tell whether or not it is their turn.

Special requirements

The device must run Android and have the necessary space to install the game. The device must be capable of accessing the Internet.

Issues

1. Is the network service operational enough to test it successfully?

To do

1. Have the network service running.
2. Have the games list operational.

2.38 Play a move online (IT-3-UC-17)

Revision history

Date	Author	Description of change
2014/03/17	Ben Stitt	Authored use case.

Use case

Play a move online

Id

IT-3-UC-17

Description

A move is sent to an online game.

Level

User goal

Primary actor

The user wishes to advance game play in their favour.

Stakeholders and interests

Other players wish game play to advance.

Pre-conditions

The user must have the application installed and running, and the user's device must be connected to the Internet. The Online Game Activity must also be active, and it must be the user's turn.

Post conditions

Success end condition

The user’s move is sent to the server, and the game state is updated in the server’s database.

Failure end condition

The game state remains in its current state.

Minimal Guarantee

The application does not crash, and data on the device remains unaffected. Existing online games remain unaffected as well.

Trigger

The user taps the Done button on their turn in the Online Game Activity.

Main success scenario

1. A request is sent to the server to update the game state.
2. The game state is updated in the server’s database.
3. The user is returned to the Online List Activity.

Frequency

This use case occurs whenever a user wishes to make a move in an active game

Assumptions

The user is ready to send a move to the server.

Special requirements

The device must run Android and have the necessary space to install the game. The device must be capable of accessing the Internet.

Issues

1. Is the network service operational enough to test it successfully?

To do

1. Have the network service running.
2. Have the games list operational.

2.4 Burn downs and ups

2.41 Iteration three progress

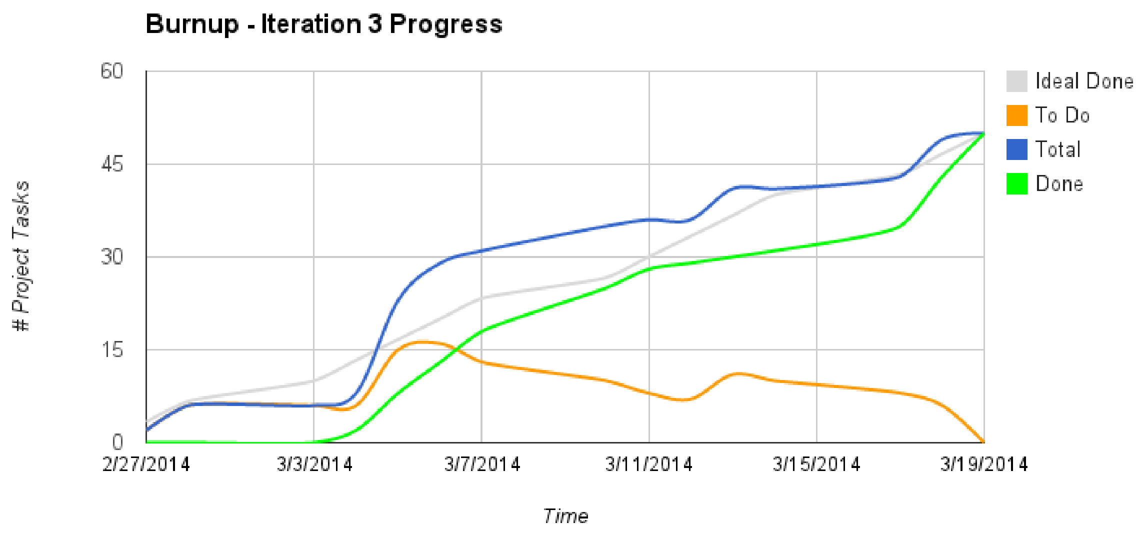


Figure 2.1 Iteration three progress burn down and up chart

2.42 Iteration two versus iteration three

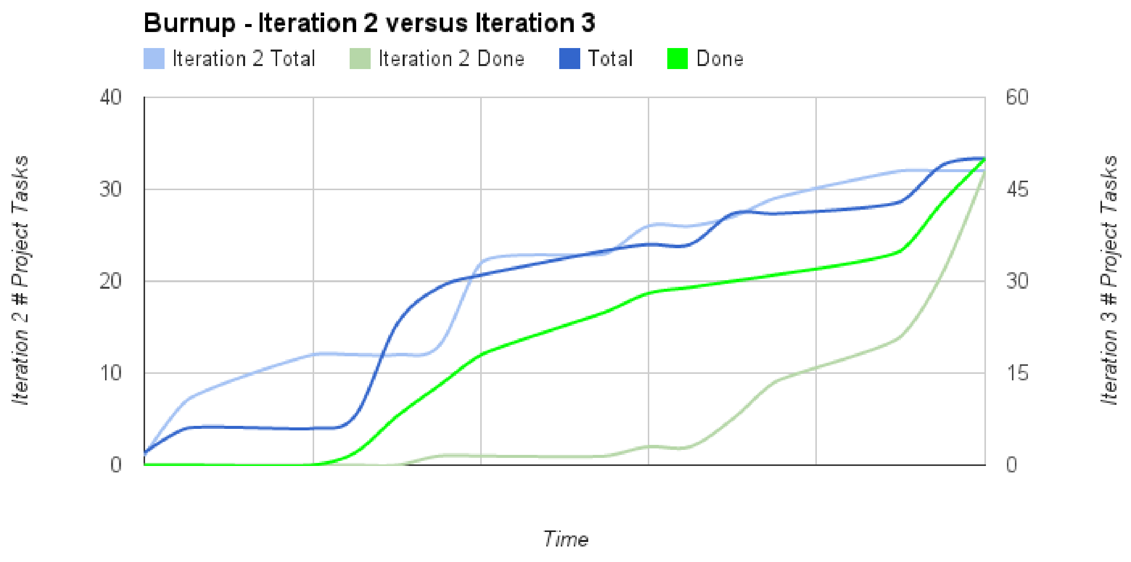


Figure 2.2 Iteration two versus iteration three burn up chart

2.5 Presentation

The following presentation will be presented during the retrospective meeting on March 20, 2014. Order follows: left to right, top to bottom.

Review

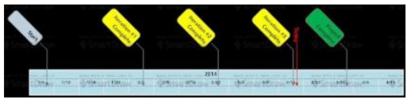
Using Agile principles

Split project into time-boxed iterations

Created and prioritized the list of features

Pulled highest priority features into the current iteration

Project Timeline



Iteration #3 Goals

Pulled the next set of features from the backlog

- Artificial intelligence integration
- Initiate some architecture and database work to support online play
- Help section
- Refactoring to reduce technical debt

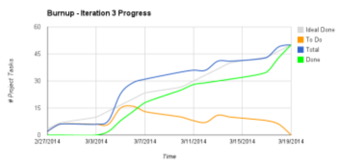
Improve project progress

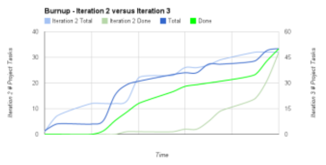
Planning

All tasks are tracked in GitHub

Split the 3 week iteration into 1 week sprints stressing completion of functionality each week

Analysis, design, construction and testing spawned many subtasks eventually leading to a total of 50 project tasks



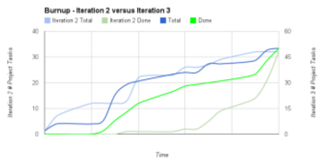


Observations

Iteration 3 was same duration as iteration 2

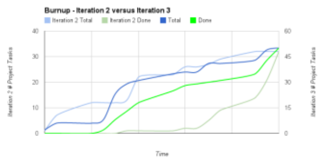
Observable improvement progress consistency which made for a smoother iteration

Goal for next iteration will be to continue this trend



Next Iteration Goals

Finalize network capabilities for online features and game play over the internet



Demonstration

Feedback?
Q&A

3 Design

3.1 Guidelines

3.11 Game rules

For our version of Chinese checkers, we have adapted the rules from Masters Traditional Games ("The Rules of Chinese Checkers", 2012) with modifications to better suit a digital environment.

Components

All equipment needed to play within the application is digital. A moving piece is referred to as a “peg” and the holes in the board where a peg can move are referred to as “slots”. The board is a six-pointed star with six “home-“ and “end-“ zones each in a different colour (red, orange, yellow, green, blue and purple). Each zone is a triangle with four peg slots on each edge and the interior of the board is a hexagon with five peg slots on each edge.

Setup

A game of Chinese Checkers can be played with two, three, four or six players who each get assigned ten pegs of the chosen colour that are placed in their home-zone. In two, four and six player games, each player starts in opposing triangles. In three player games, the players will start equidistant from each other.

Objective

To be the first player to move all ten of their pegs into the opposing end-zone or move enough of their own pegs into the opposing end-zone to trigger a win-condition.

Play

The player who chooses to be Player One will always make the first move at the start of a round. Each round is over when each player has finished a move. A move may consist of:

Sliding

In one turn, a player may move one of their pegs into an adjacent slot that is empty.

Hopping

In one turn, a player may hop over any other peg with one of their pegs. A hop must be over an adjacent peg and into an empty slot directly beyond it. A player may choose to continue hopping if the setup allows for it or finish their turn after any number of hops. It is legal to move by hopping from a home-zone and into an end-zone in one turn.

Combo

To allow for a quicker game, in one turn, a player may slide one of their pegs and then proceed to hop over another peg with the peg that was just slid if a hop is available.

Pegs are never removed from the board and any peg can be placed in any empty slot including the home-zone of any player or non-player. Once one of your pegs has been moved outside of your end-zone, it cannot move to the interior of the board; it can only within the slots of that end-zone.

Winning

The game ends when an end-zone is filled with ten pegs and at least one of those pegs belong to the opposing player. For example, if there are nine Green pegs inside of Green Player's home-zone and on Red Player's (opposing) next turn they move one Red peg inside of this zone then Red is the winner. This rule has been added to prevent an opposing player from preventing another player from winning by blocking the end-zone slots.

3.12 FAQ-style help

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

A: 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.

Q: How do I play?

A: 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.

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

A: 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.

Q: Why do I have a username?

A: 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.

Q: How do I start a game over the Internet?

A: 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 time, the game will begin. If enough players are not found, AI players will be used to fill the game.

Q: I got a notification telling me it's my turn. How do I take my turn?

A: Tap the Online button, and then tap the game with an arrow next to it on the list.

Q: How do I leave an online game?

A: Tap the Online button, then press and hold the game you want to leave. You will be asked if you want to quit it. Tap Yes. You will be replaced by an AI, so the other players can keep playing.

3.13 User interface control elements

Main activity

Component ID	**Control type**	**Effect**
offlineActivityConfigurationButton	Button	• Starts the OfflineConfigurationActivity OR OfflineGameActivity (with resume dialog) if game is saved
onlineListActivityButton	Button	• Starts the OnlineListActivity
helpActivityButton	Button	• Starts the HelpActivity
settingsActivityButton	Button	• Starts the SettingsActivity

Offline Configuration activity

Component ID	**Control type**	**Effect**
twoPlayerButton	ToggleButton	• Displays two input fields: ○ offlineRedPlayerNameEditText ○ offlineBluePlayerNameEditText
threePlayerButton	ToggleButton	• Displays three input fields: ○ offlineRedPlayerNameEditText ○ offlinePurplePlayerNameEditText ○ offlineYellowPlayerNameEditText
fourPlayerButton	ToggleButton	• Displays four input fields: ○ offlineRedPlayerNameEditText ○ offlineGreenPlayerNameEditText ○ offlinePurplePlayerNameEditText ○ offlineYellowPlayerNameEditText ○ offlineOrangePlayerNameEditText
sixPlayerButton	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>PlayerMediumButto n	ToggleButton	
Offline<COLOUR>PlayerHardButton	ToggleButton	
offlineRedPlayerNameEditText	EditText	• Allows for textual entry
offlineGreenPlayerNameEditText	EditText	

offlinePurplePlayerNameEditText	EditText	*Validation*: • Can't be blank depending on number of human players selected
offlineBluePlayerNameEditText	EditText	
offlineYellowPlayerNameEditText	EditText	
offlineOrangePlayerNameEditText	EditText	
offlineGameActivityButton	Button	• Finishes this activity and starts • OfflineGameActivity

Offline Game activity

Component ID	**Control type**	**Effect**
gameMoveResetButton	Button	• Undoes any peg movements since start of current turn
gameMoveDoneButton	Button	• Updates game with player's move • Rotates the board • Changes the current player label at the top of the screen
gamePlayerListButton	Button	• Creates/displays a dialog which displays a list of the players

Offline Game activity: resume dialog

Component ID	**Control type**	**Effect**
dialogAcceptButton	Button	• Dialog disappears
dialogCancelButton	Button	• This activity finishes and OfflineConfigurationActivity starts • Saved game is deleted

Offline Game activity: end of game dialog

Component ID	**Control type**	**Effect**
dialogCancelButton	Button	• This activity finishes and MainActivity starts
dialogAcceptButton	Button	• This activity finishes and OfflineConfigurationActivity starts

Online List activity

Component ID	**Control type**	**Effect**
onlineNewGameButton	Button	• Creates/displays a dialog to create a new online game

onlineGameActivityButton	LinearLayout	• Hides the notification icon on the game's list entry (if it is visible) • Starts OnlineGameActivity and loads any updates to that game's state

Online List activity: new game dialog

Component ID	Control type	Effect
dialogCancelButton	Button	• Dialog disappears
dialogAcceptButton	Button	• Dialog disappears; new game is added to the OnlineList's game list
numberOfPlayersRadioGroup	RadioGroup	• Contains the four below options
twoPlayerButton	RadioButton	• Will be used to create/join a game of that many players when the dialogAcceptButton is clicked
threePlayerButton	RadioButton	
fourPlayerButton	RadioButton	
sixPlayerButton	RadioButton	

Online Game activity

Component ID	Control type	Effect
gameMoveResetButton	Button	• Undoes any peg movements since start of current turn
gameMoveDoneButton	Button	• Updates game with player's move • Rotates the board • Changes the current player label at the top of the screen
gamePlayerListButton	Button	• Creates/displays a dialog which displays a list of the players

Online Game activity: end of game dialog

Component ID	Control type	Effect
dialogAcceptButton	Button	• This activity finishes and starts OnlineListActivity

Settings activity

Component ID	Control type	Effect
settingsShowMovesRadioGroup	RadioGroup	• Container for the two radio buttons below

settingsShowMoveOnButton	RadioButton	• Sets a SharedPreference flag to show possible moves in-game
settingsShowMoveOffButton	RadioButton	• Sets a SharedPreference flag to hide possible moves in-game
settingsUsernameEditText	EditText	• Allows for textual entry • Makes request to server to change name when focus is lost
settingsFacebookConnectButton	Button	• Goes to the facebook site to authenticate the user
settingsFacebookUnlinkButton	Button	• Starts a dialog to confirm unlinking of the account

Settings activity: unlink Facebook dialog

Component ID	Control type	Effect
dialogCancelButton	Button	• Dialog disappears
dialogAcceptButton	Button	• Validates password with server. If: ○ Correct → dialog disappears; account is deleted ○ Incorrect → error icon beside password field
dialogPasswordEditText	EditText	• Allows for textual entry • Used to confirm identity when unlinking facebook account

3.14 MySQL stored procedures

All stored procedures follow the form **<heading>.<procedure name>**

For example: GU.userJoinGame(1, 3);

If a procedure is returning a value, include a **@** symbol with a variable name after the parameters for a return statement.

For example: users.createUser(@id); SELECT @id;

Users

Procedure	Description
createUser()	• returns userID of new user created
createAI()	• returns userID of new AI created

getAi(userID)	• returns isAi field of users
getUserID(username)	• returns userID of username
getUsername(userID)	• returns username of userID
setUsername(String newName, userID)	• updates userID with new username
deleteUser()	• Designed to be used on human users, replaces user with AI in every game they are currently in

Games

Procedure	Description
createGame(numberOfPlayers, createdTime)	• creates a new game
getCreated(gameID)	• returns DATETIME game was created at
getCurrentTurn(gameID)	• returns INT what current turn of the game is
getNotReadyList()	• returns list of all games where isReady is 0
getNumPlayer(gameID)	• returns number of players allowed for a game
getReady(gameID)	• returns isReady for a gameID
getWinner(gameID)	• returns winner of a game, NULL if no winner set
checkWinner(userID, gameID)	• returns a BOOL representing whether a player has the win condition for the game. 1 = true, 0 = false.
setWinner(userID, gameID)	• sets winner of a game
nextTurn(gameID)	• iterates currentTurn according to number of players

GamesUsers (GU)

Procedure	Description
getGameID(guID)	• returns gameID of some gamesusers record
getGuID(userID, gameID)	• returns gamesusersID
getPlayerNumber(guID)	• returns player number of that gamesusers session
getUsername(guID)	• returns username of that gamesusers session
getUserID(guID)	• returns userID of that gamesusers session
userJoinGame(userID, gameID)	• creates a new relation between user and game, automatically fills playerNumber appropriately according to how many players that game is for
userLeaveGame(userID, gameID)	• replaces the user with an dynamically created AI, checks to see if all other players are AI. If so the game is deleted, also automatically clears TABLE users of any unassociated AI. Once a game is deleted, pieces of the associated game are also deleted.
getUsersGames(userID)	• returns a list of gameIDs the userID belongs to

Games

Procedure	**Description**
createPiece(guID, Row, Index)	• returns pieceID of the piece created
getPieceID(guID, Row, Index)	• returns pieceID
updatePiece(guID, newRow, newIndex)	• updates the row and index of piece in question
getGamePieces(gameID)	• returns list of playerNumber, Row, Index of pieces belonging to the gameID.

3.2 Component diagrams

3.21 Server Architecture

- See Appendix U

3.22 Push Notification

- See Appendix V

3.3 Class diagrams

3.31 Push Notifications

- See Appendix W

3.32 Network API

- See Appendix X

3.4 User interface control flow

3.41 Overview

- See Appendix A

3.42 Main activity

- See Appendix B

3.43 Offline configuration activity

- See Appendix C

3.44 Offline game activity

- See Appendix D

End of game dialog

- See Appendix E

Resume dialog

- See Appendix F

3.45 Online list activity

- See Appendix G

3.46 Online game activity

- See Appendix H

End of game dialog

- See Appendix I

New game dialog

- See Appendix J

3.47 Settings activity

- See Appendix K

Unlink Facebook dialog

- See Appendix L

3.5 Entity relationship diagrams

3.51 Relational database

- See Appendix N

3.6 Visual design

3.61 Introduction

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

3.62 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)

Very Dark Primary Palette

🟥	🟫	🟨	🟩	🟦	🟪
rgb(76, 9, 10)	rgb(83, 44, 0)	rgb(78, 74, 0)	rgb(19, 62, 28)	rgb(13, 39, 62)	rgb(52, 0, 68)

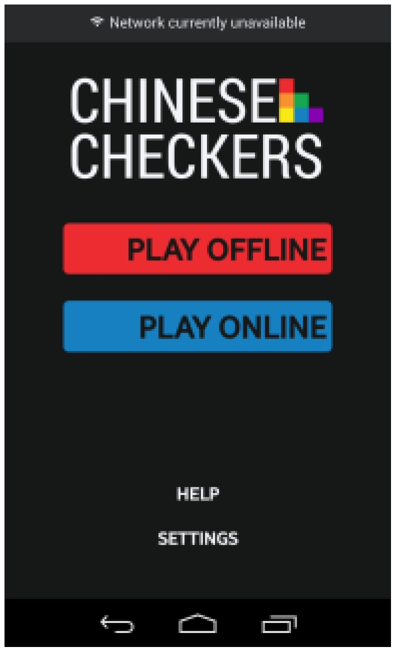
Chosen swatches

One new palette was added to the colour scheme to better identify empty peg slots in the end- or home-zones. Previous, the Light Primary Palette colours were used to identify these empty pieces, but we found through testing that it was hard to see the hints when they showed up on these spaces. As such, the Light Primary Palette and the Dark Primary Palette schemes have swapped purposes in relation to the game board: the colours in the Light Primary Palette now reflect which piece is selected. The Dark Primary Palette is still used for accent colours on pages such as the Offline Configuration Activity to show which AI difficulty level has been selected.

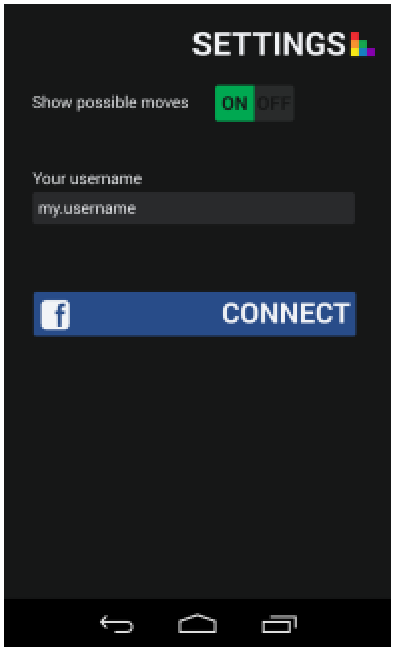
Rejected swatches

Previously rejected swatches will no longer be considered for implementation in following iterations due to proven position reception of the chosen themes.

3.63 Mock-ups



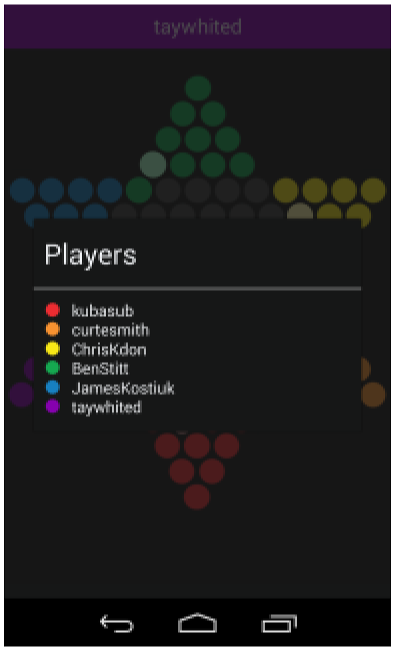
See Appendix O for full-scale mock-up image



Show possible moves: ON OFF

Your username: my.username

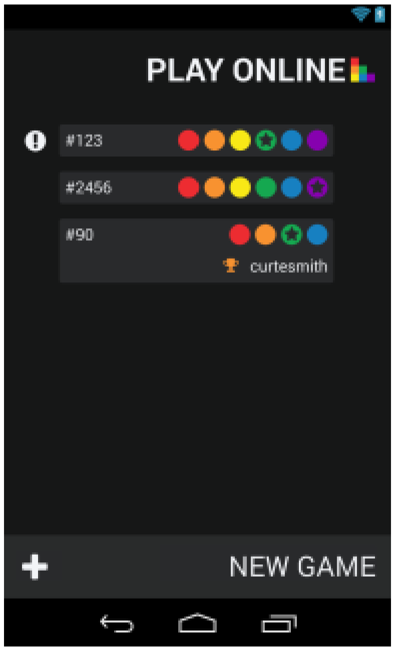
See Appendix Q for full-scale mock-up image


taywhited

Players

- kubasub
- curtesmith
- ChrisKdon
- BenStitt
- JamesKostiuk
- taywhited

See Appendix P for full-scale mock-up image


PLAY ONLINE

#123

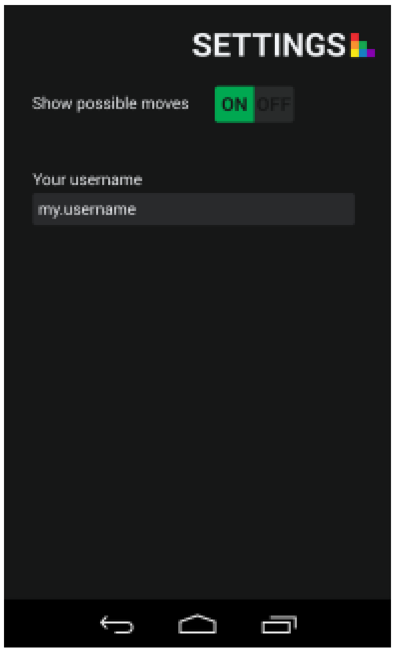
#2456

#90

curtesmith

NEW GAME

See Appendix S for full-scale mock-up image

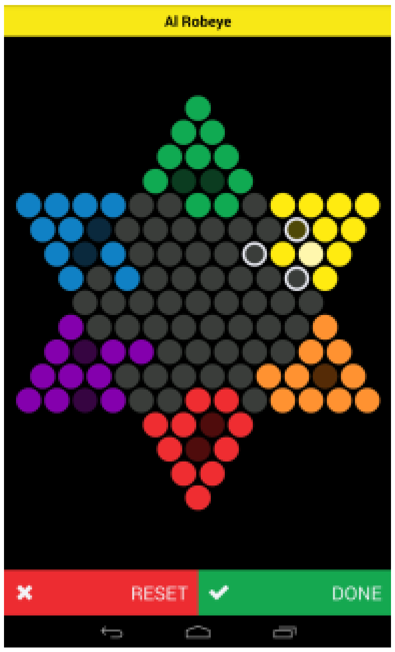

SETTINGS

Show possible moves ON OFF

Your username

my.username

See Appendix R for full-scale mock-up image



See Appendix T for full-scale mock-up image

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

- Links to Help and Settings have been added to the main activity. On pages where having an internet connection may be important, a “no connectivity” banner will be show at the top of that activity. An example of this is shown in the main activity mock-up.
- A player may now see a list of players in their game by tapping on the banner at the top of the screen to display a dialog box with the information.
- Initially the application was going to have Facebook integration and the settings page was designed accordingly. Facebook integration was dropped in favour of an in-house users system. The initial wireframe sketch of the settings page can be seen in Appendix M.
- The Play Online game list activity shows which games you are currently in. If they have ended, the winner will be displayed. The star signifies that the colour it is on is your peg colour in that game. A notification icon will appear on the left side of a game item if it is your turn to make a move.
- The game board has been updated to showcase all of the accepted palettes: Monochrome, Primary, Light Primary, Dark Primary and Very Dark Primary. It is currently Yellow’s turn in the mock-up.

4 Testing

4.1 Iteration three testing plan

4.11 Revision history

Date	Author	Description of change
2014/03/07	Saajid Mohammed	Authored testing plan
2014/03/11	Ben Stitt	General revision and editing

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 the offline configuration, online configuration, settings and 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.

4.13 Objectives

Verify the current phase programming is functional and works consistently.

4.14 Testing phases

Automated testing

User interface unit

- MainActivity
  - offlineActivityConfigurationButton
    - Verify clickabilty
    - Verify visibility
    - Verify proper methods are being called
  - onlineListActivityButton
    - Verify clickabilty
    - Verify visibility
    - Verify proper methods are being called
  - helpActivityButton
    - Verify clickabilty
    - Verify visibility
    - Verify proper methods are being called
  - settingsActivityButton
    - Verify clickabilty
    - Verify visibility
    - Verify proper methods are being called
- OfflineConfigurationActivity
  - twoPlayerButton
    - Verify clickabilty
    - Verify visibility
    - Verify displays 2 input fields are shown
  - threePlayerButton
    - Verify clickabilty
    - Verify visibility
    - Verify displays 3 input fields are shown
  - fourPlayerButton
    - Verify clickabilty
    - Verify visibility
    - Verify displays 4 input fields are shown
  - sixPlayerButton
    - 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 selection
        - 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 selection

        - 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 selection
        - 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 selection
        - 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 selection
        - 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 selection
        - 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
    - moveResetButton
        - Verify clickabilty

        - Verify visibility
        - Verify proper methods are being called
    - moveDoneButton
        - 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
    - gamePlayerListButton
        - Verify clickabilty
        - Verify visibility
        - Verify proper methods are being called
        - Verify game is playable
    - HelpActivity
        - Verify text is visible
- SettingsActivity
    - settingsShowMovesRadioGroup
        - Verify only one of the group members are selectable at a time
    - settingsShowMoveOnButton
        - Verify visibility
        - Verify clickability
        - Verify change is made in SharedPreference
    - settingsShowMoveOffButton
        - Verify visibility
        - Verify clickability
        - Verify change is made in SharedPreference
    - settingsUsernameEditText
        - Verify visibility
        - Verify accepts text or doesn’t depending on case
        - Verify proper error on no input

User interface integration

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

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

Manual testing

- Verify game play is complaint with the rules of Chinese Checkers as stated in Game rules
- Verify AI players are obeying game rules and functions with noticeable difference in skill levels.
- Verify dialog functionality
    - ResumeDialog – Offline & Online Game Activity
        - Verify is called at proper time
        - Verify visibility
        - dialogAcceptButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called
            - Verify proper transition
        - dialogCancelButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called
            - Verify proper transition
    - EndofGameDialog– Offline & Online Game Activity
        - Verify is called at proper time
        - Verify visibility
        - dialogCancelButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called
            - Verify proper transition
        - dialogAcceptButton
            - Verify clickabilty
            - Verify visibility
            - Verify proper methods are being called
            - Verify proper transition

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

4.16 Features to be tested

- MainActivity
    - Button functionality
    - Transition to OfflineConfigurationActivity
    - Transition to HelpActivity
    - Transition to SettingsActivity
- OfflineConfigurationActivity
    - Button functionality
    - Transition to OfflineGameActivity
    - Transition to HelpActivity

    - Transition to SettingsActivity
    - Transition to MainActivity
- OfflineGameActivity
    - Button functionality
    - Game playability
    - Popup Dialogs
    - Transition to HelpActivity
    - Transition to SettingsActivity
- OnlineListActivity
    - Button functionality
    - Player lobby tracking
    - Transition into proper OnlineGameActivty with updated state
    - Transition to HelpActivity
    - Transition to SettingsActivity
- OnlineGameActivity
    - Button functionality
    - Game playability
    - Popup Dialogs
    - Transition to HelpActivity
    - Transition to SettingsActivity
- SettingsActivity
    - Toggle functionality
    - Button functionality
    - Popup Dialogs
    - Transition back to activity called from
- HelpActivity
    - Button functionality
    - Transition back to activity called from
- Game adheres to game rules
- 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 Pobojewski	Deputy Lead	___________________________	_____________
Ben Stitt	Documentation Lead	___________________________	_____________
Saajid Mohammed	Test Lead	___________________________	_____________

4.2 Unit testing

4.21 Main tests

- See Appendix AD

GridUnitTest

- See Appendix AE

MainActivityUnitTest

- See Appendix AF

MainAndConfigAndGameIntegrationTest

- See Appendix AG

MainAndConfigIntegrationTest

- See Appendix AH

OfflineConfigurationActivityUnitTest

- See Appendix AI

OfflineGameActivityUnitTest

- See Appendix AJ

ActivityAITest

- See Appendix AK

4.22 UI engine tests

- See Appendix AL

DimensionsTest

- See Appendix AM

PieceDrawingDetailsTest

- See Appendix AN

PiecePositionSystemTest

- See Appendix AO

PixelPositionTest

- See Appendix AP

4.23 Summary

- See Appendix AQ

4.3 Integration testing

- See Appendix AC

4.4 Manual testing

- See Appendix AB

4.5 Regression testing

- See Appendix AA

4.6 Ease of use testing

4.61 Ease of use testing tasks

- Task 1: Configure a 2 player game against Easy AI and play till the end. Rate on difficulty of opponent.
- Task 2: Configure a 2 player game against Medium AI and play till the end. Rate on difficulty of opponent.
- Task 3: Change your user name to a name of your choice.
- Task 4: Get to the help page from within the game, and rate on how helpful it appeared.

4.62 Results

- See Appendix AR

4.7 Acceptance testing

4.71 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.

4.72 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 4.0, 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 Android Canvas.

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

4.73 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

4.74 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 Settings button	The Settings Activity launches, displaying the Show Hints and Username widgets, populated with their current states.	X	
2	Tap the Show Hints widget to the off position on the Settings Activity screen.	The widget appears in the off position.	X	
3	After completing task 2, start an offline game with at least one human player, and tap a peg.	Possible moves are not shown on the board.	X	
4	Tap the Show Hints widget to the on position on the Settings Activity screen.	The widget appears in the on position.	X	
5	After completing task 4, start an offline game with at least one human player, and tap a peg.	Possible moves are shown on the board.	X	
6	Change the Username field on the Settings Activity screen.	The new username is displayed in the field.	X	
7	After completing task 6, exit the application, and clear it from memory. Open it again, and return to the Settings Activity.	The new username is displayed in the field.		X
8	In the Offline Game Configuration Activity, select a two-player game, with one human player and one AI. Make sure the difficulty selector is set to Easy.	A game begins against an AI of easy difficulty. The game can be completed normally.	X	
9	In the Offline Game Configuration Activity, select a two-player game, with one human player and one AI. Make sure the difficulty selector is set to Medium.	A game begins against an AI of medium difficulty. The game can be completed normally. This AI is more difficult than on the easy level.	X	
10	In the Offline Game Configuration Activity, select a two-player game, with two AI players. Make sure the difficulty selector is set to Easy for one player and Medium for the other.	A game begins between the two AI players, and the medium level AI is more likely to win.	X	
11	Repeat task 10 for each number of players. At least one AI player should be set to Medium each time.	The medium level AI is more likely to win.		X
12	Tap the Online Game button on the main menu.	The Online List Activity is launched, and the sample list of online games is displayed.	X	
13	Press and hold a game in the Online List Activity.	A dialog is displayed asking the user if they wish to remove themselves from the game.	X	
14	After completing task 13, accept the prompt.	The game is removed from the online list.	X	

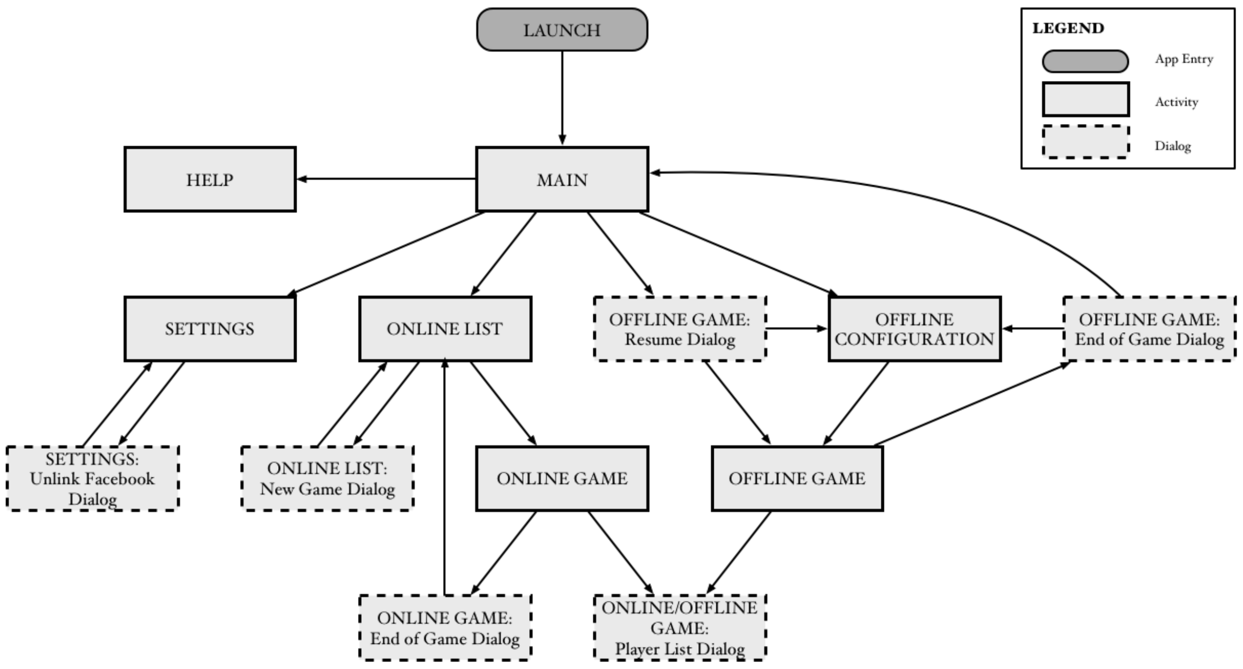
15	In an offline game with all AI players, leave the Activity.	A prompt to remove the game is displayed, and the game doesn’t resume until the user taps the Resume button.	X	
16	Tap the Help button on the main menu.	A scrollable help page is displayed.	X	

4.75 Notes

In 3-player AI-only game, AI players move the same pegs back and forth at endgame, resulting in no winner.

5 Appendices

5.1 Appendix A



LAUNCH
LEGEND
App Entry
Activity
Dialog
HELP
MAIN
SETTINGS
ONLINE LIST
OFFLINE GAME: Resume Dialog
OFFLINE CONFIGURATION
OFFLINE GAME: End of Game Dialog
SETTINGS: Unlink Facebook Dialog
ONLINE LIST: New Game Dialog
ONLINE GAME
OFFLINE GAME
ONLINE GAME: End of Game Dialog
ONLINE/OFFLINE GAME: Player List Dialog

NOTES
- The Help and Settings Activities are accessible from the context menu of every other Activity.
- The Android back button functions as usual. Popping from the Activity stack is not displayed above.

Figure 5.1 Overview of user interface control flow

5.2 Appendix B

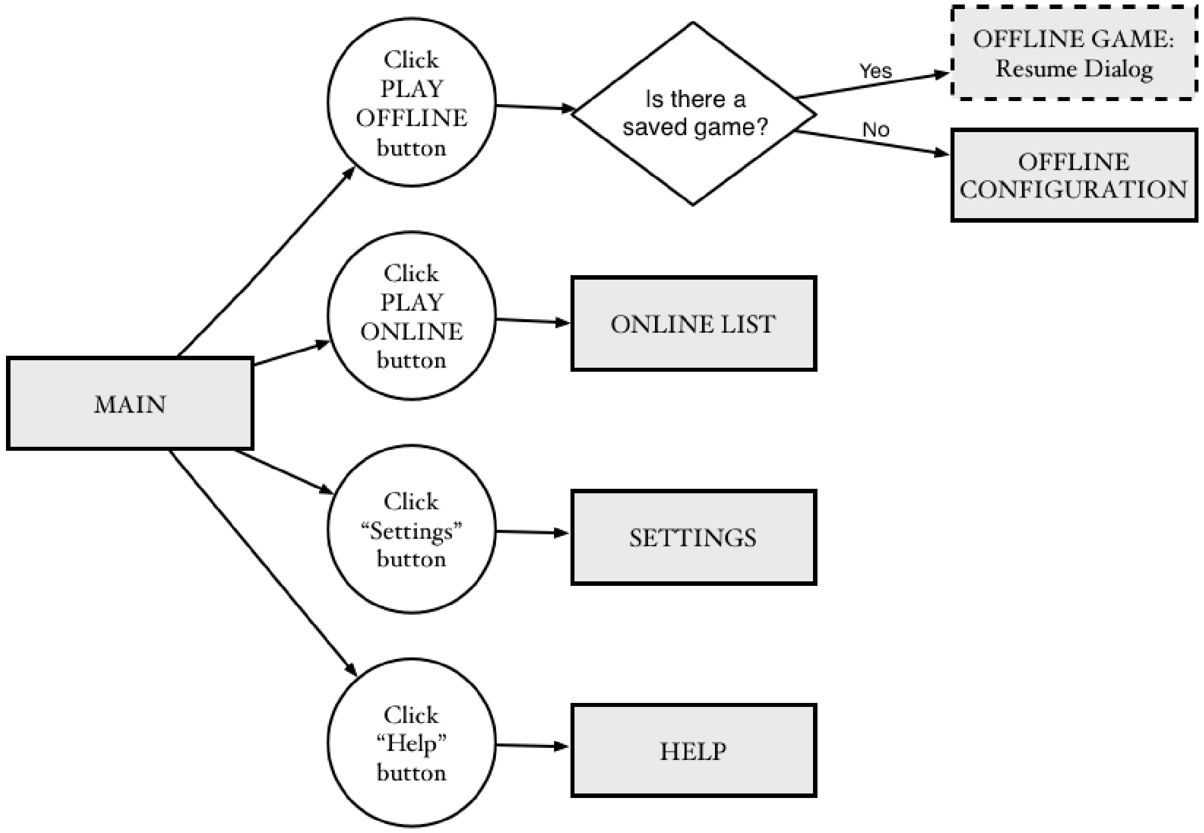


Figure 5.2 Main activity user interface control flow

5.3 Appendix C

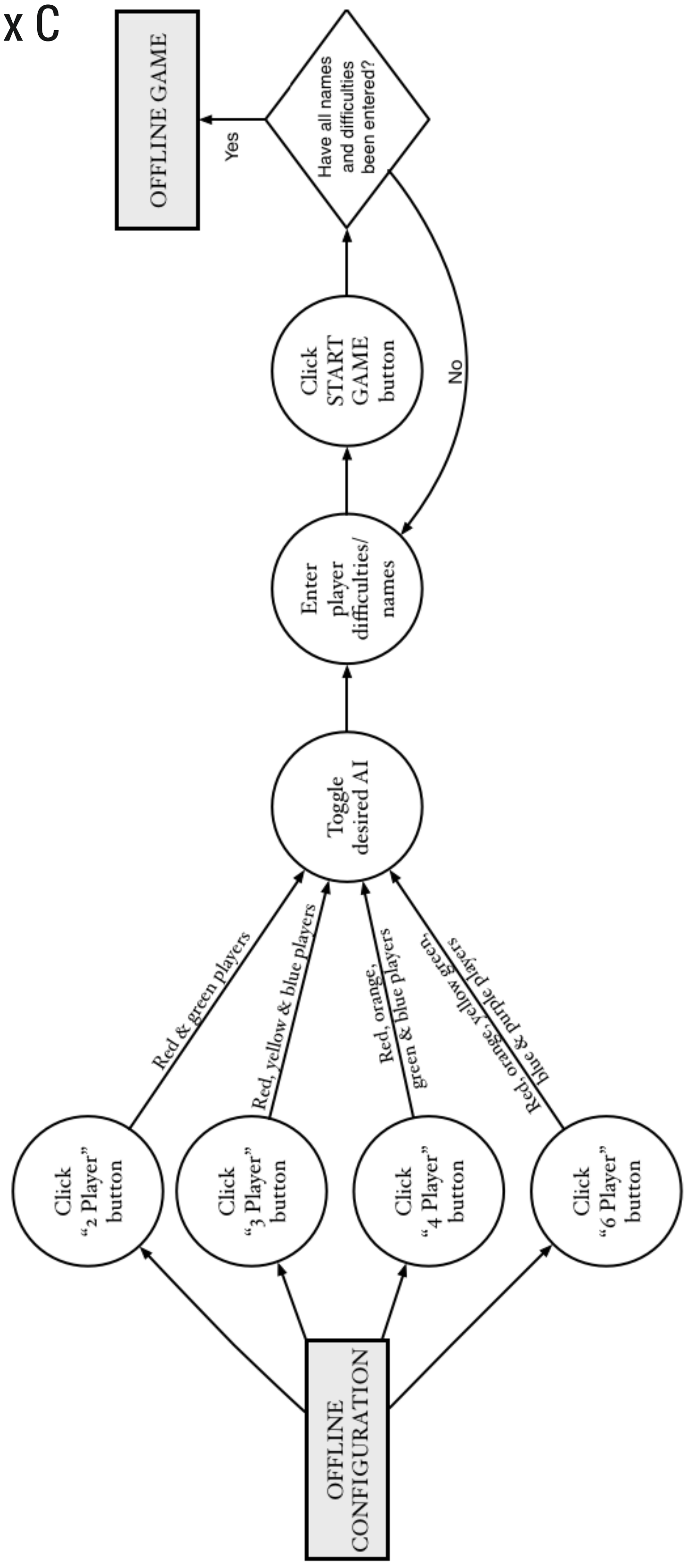


Figure 5.3 Offline game configuration activity user interface control flow

5.4 Appendix D

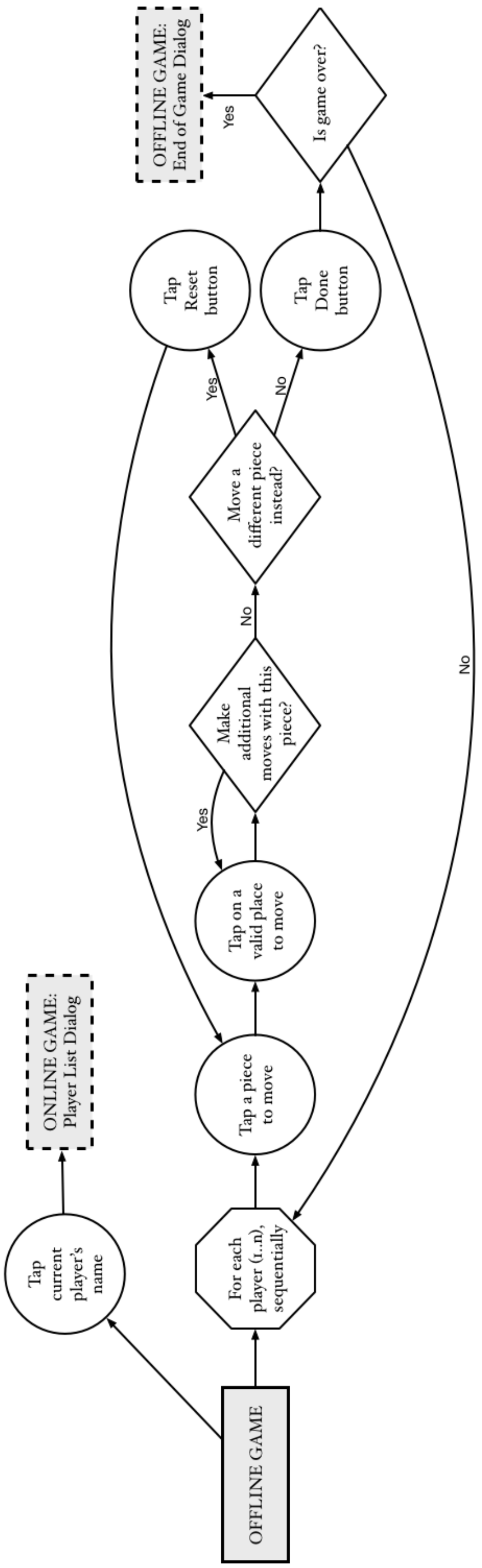

OFFLINE GAME
Tap current player's name
ONLINE GAME: Player List Dialog
For each player (1..n), sequentially
Tap a piece to move
Tap on a valid place to move
Make additional moves with this piece?
Yes
No
Move a different piece instead?
Yes
No
Tap Reset button
Tap Done button
Is game over?
Yes
No
OFFLINE GAME: End of Game Dialog

Figure 5.4 Offline game activity user interface control flow

5.5 Appendix E

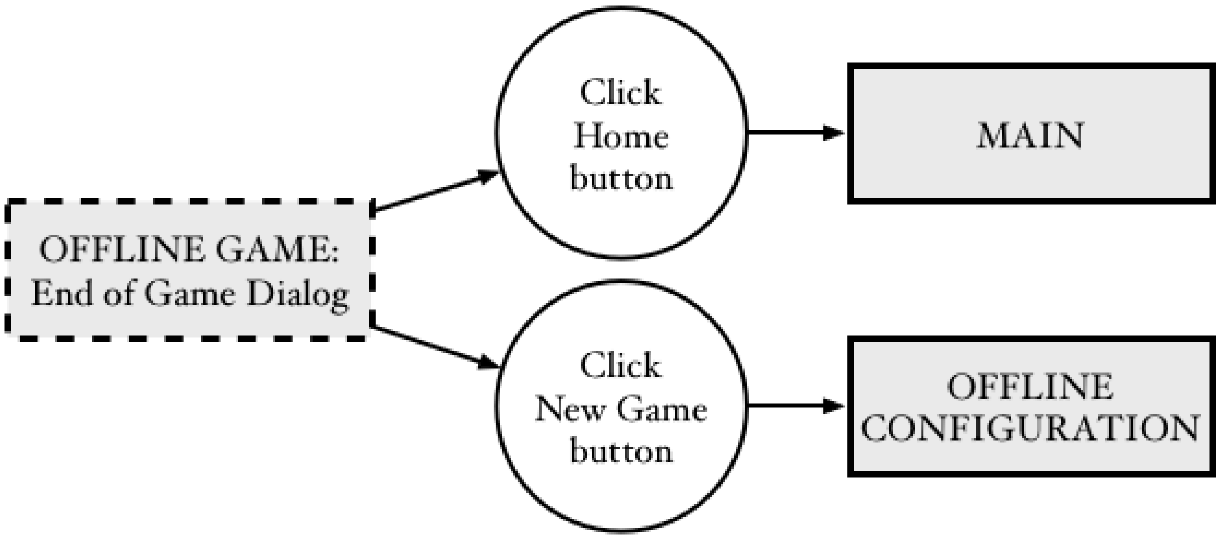


Figure 5.5 Offline game activity: end of game dialog user interface control flow

5.6 Appendix F

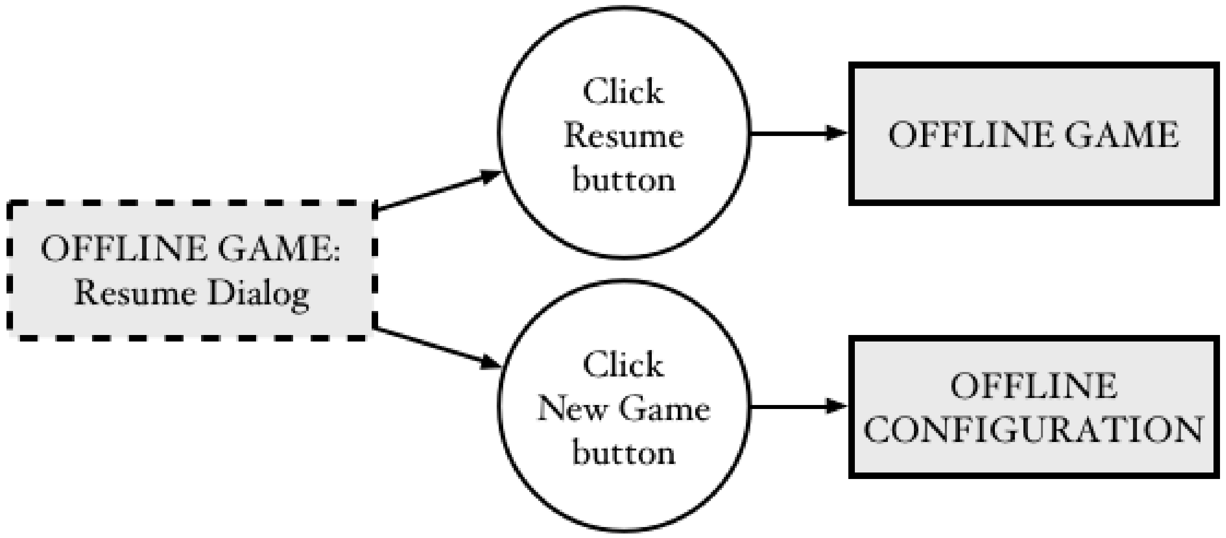


Figure 5.6 Offline game activity: resume dialog user interface control flow

5.7 Appendix G

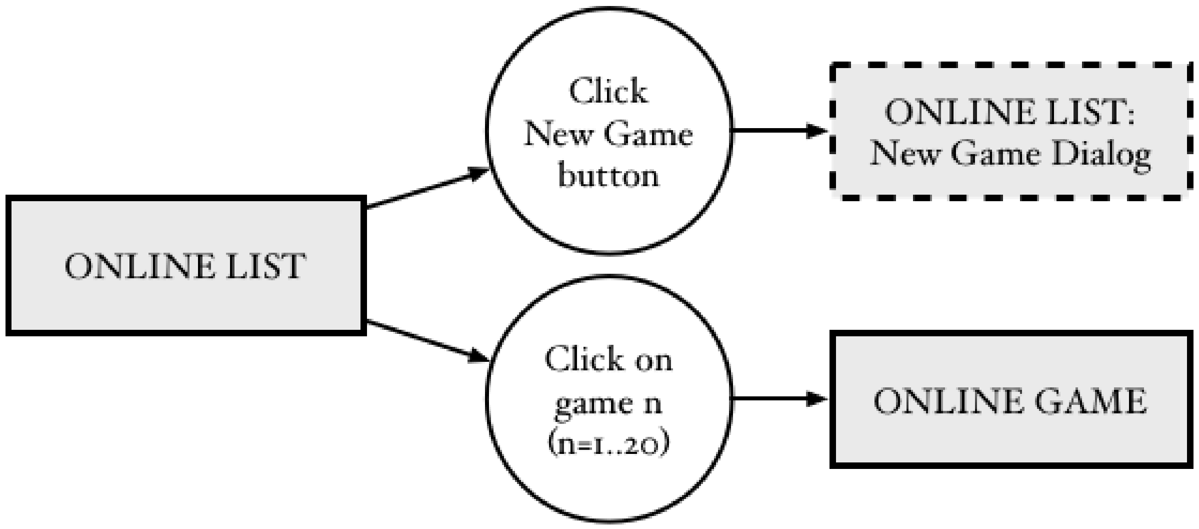
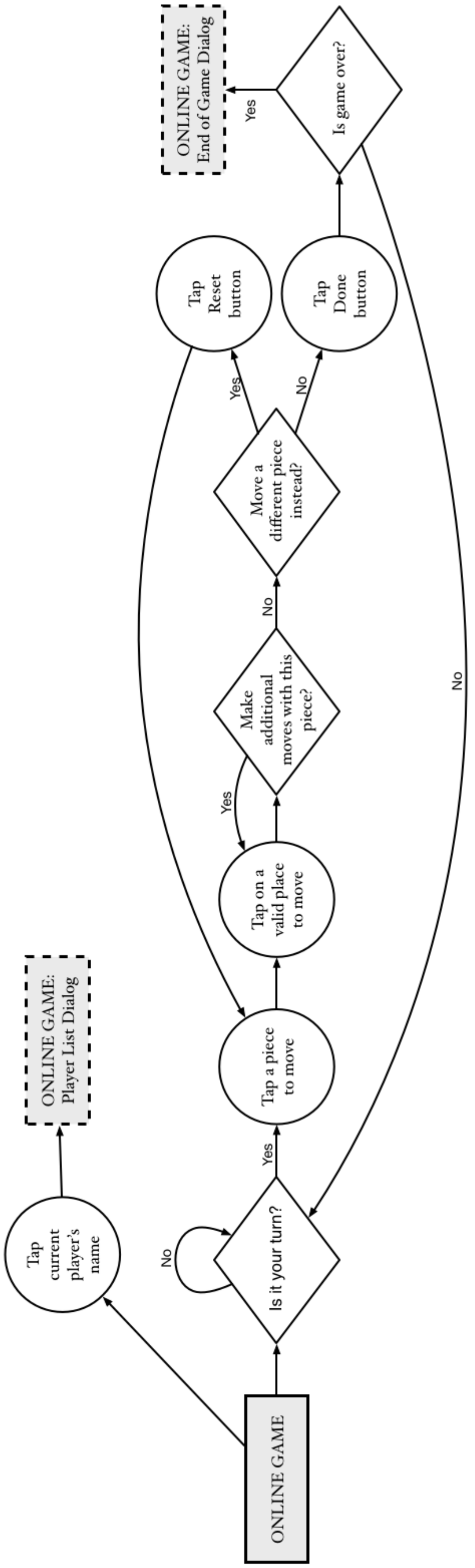


Figure 5.7 Online list activity user interface control flow

5.8 Appendix H



ONLINE GAME
Is it your turn?
No
Yes
Tap current player's name
ONLINE GAME: Player List Dialog
Tap a piece to move
Tap on a valid place to move
Make additional moves with this piece?
Yes
No
Move a different piece instead?
Yes
No
Tap Reset button
Tap Done button
Is game over?
Yes
No
ONLINE GAME: End of Game Dialog

Figure 5.8 Online game activity user interface control flow

5.9 Appendix I

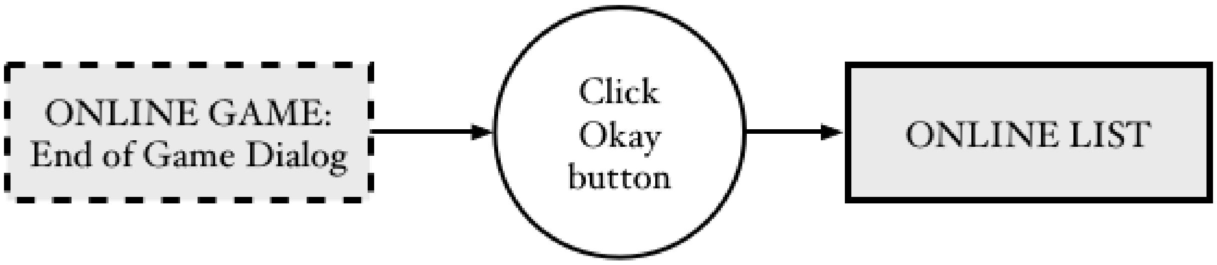


Figure 5.9 Online game activity: end of game dialog user interface control flow

5.10 Appendix J

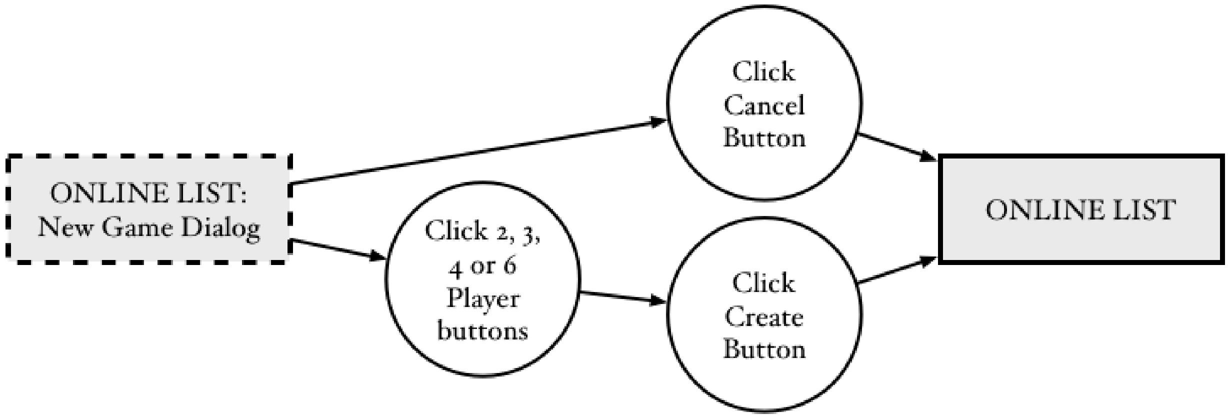


Figure 5.10 Online game activity: new game dialog user interface control flow

5.11 Appendix K

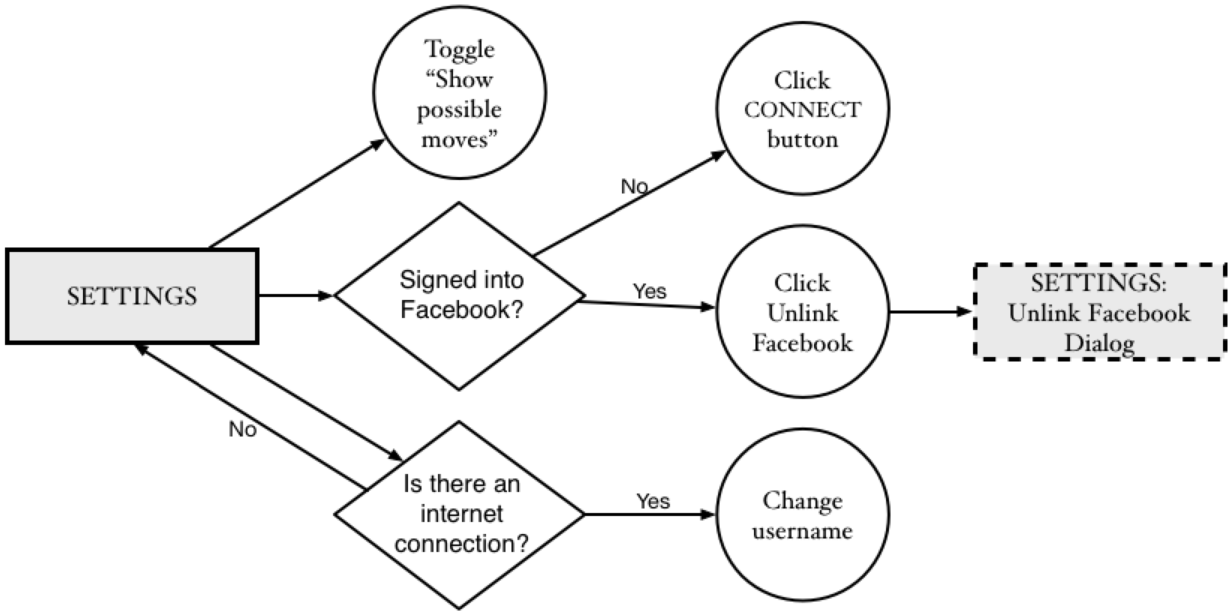


Figure 5.11 Settings activity user interface control flow

5.12 Appendix L

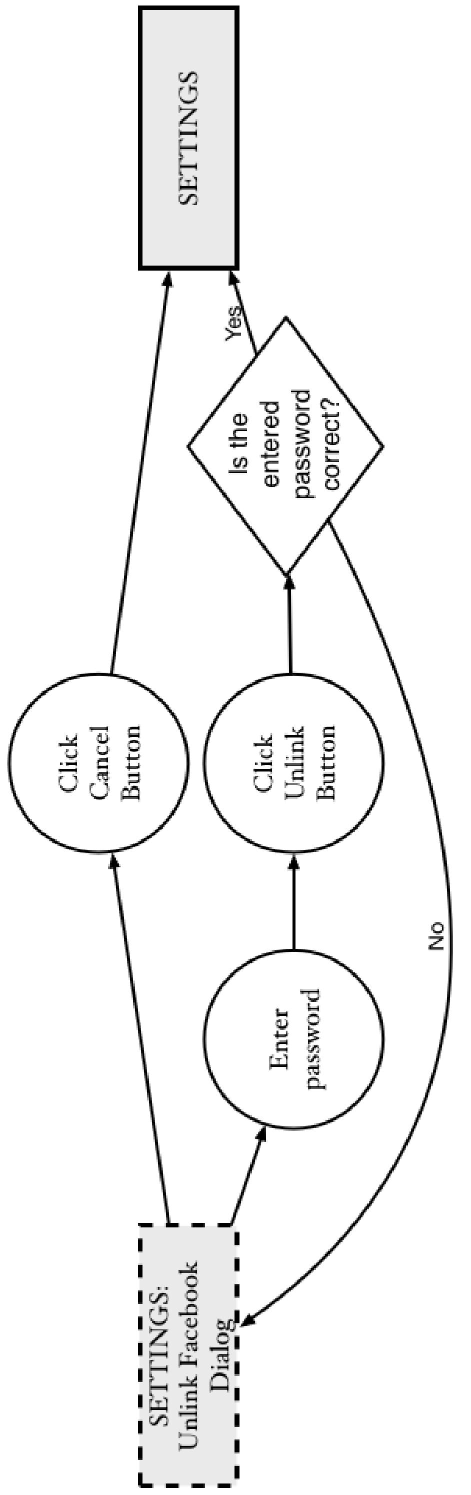


Figure 5.12 Settings activity: unlink Facebook dialog user interface control flow

5.13 Appendix M

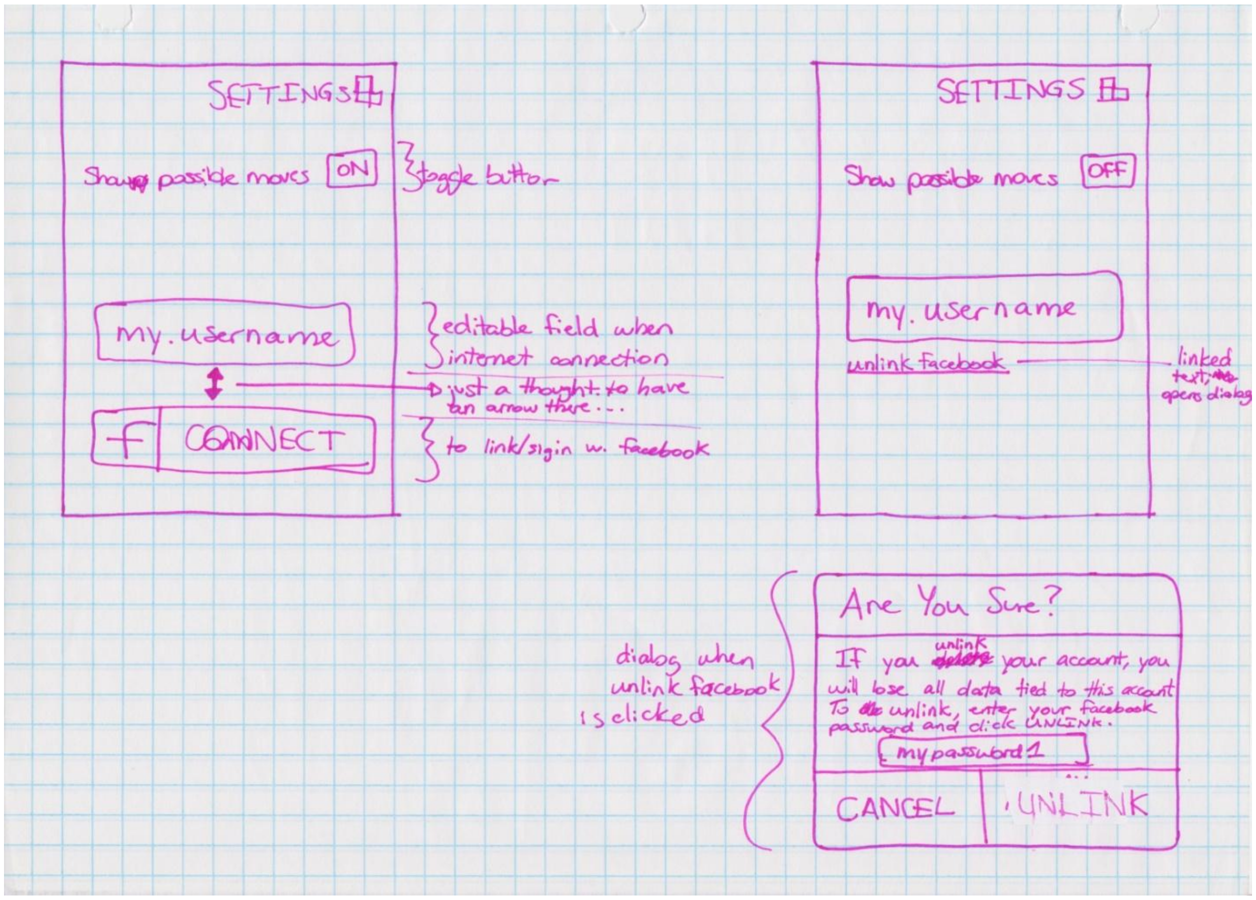

SETTINGS
Show possible moves ON
toggle button
my.username
editable field when internet connection
just a thought: to have an arrow there...
f CONNECT
to link/sign in w. facebook
SETTINGS
Show possible moves OFF
my.username
unlink facebook
linked text, opens dialog
dialog when unlink facebook is clicked
Are You Sure?
If you ~~delete~~ unlink your account, you will lose all data tied to this account. To unlink, enter your facebook password and click UNLINK.
mypassword1
CANCEL
UNLINK

Figure 5.13 Settings activity wireframe sketch

5.14 Appendix N

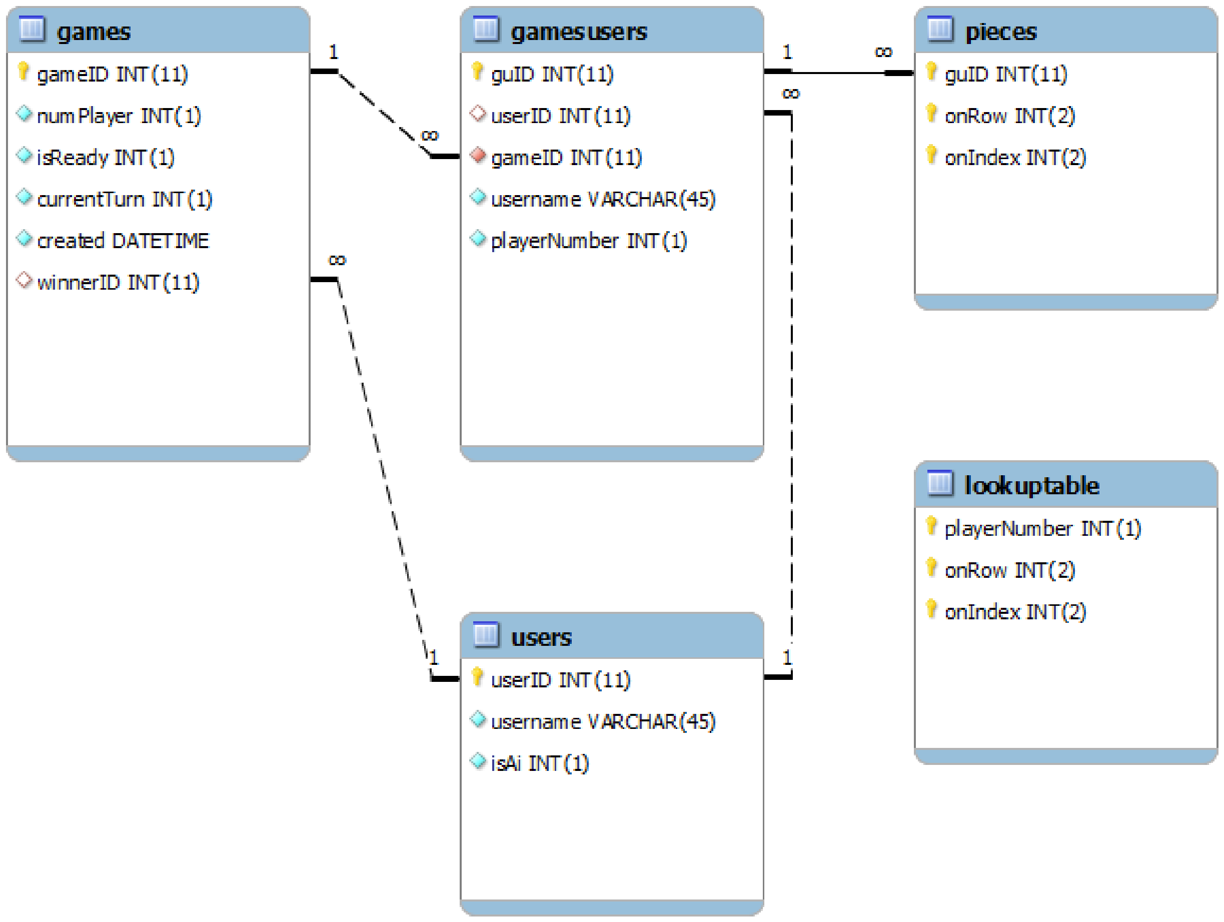


Figure 5.14 Database entity relationship diagram

5.15 Appendix O

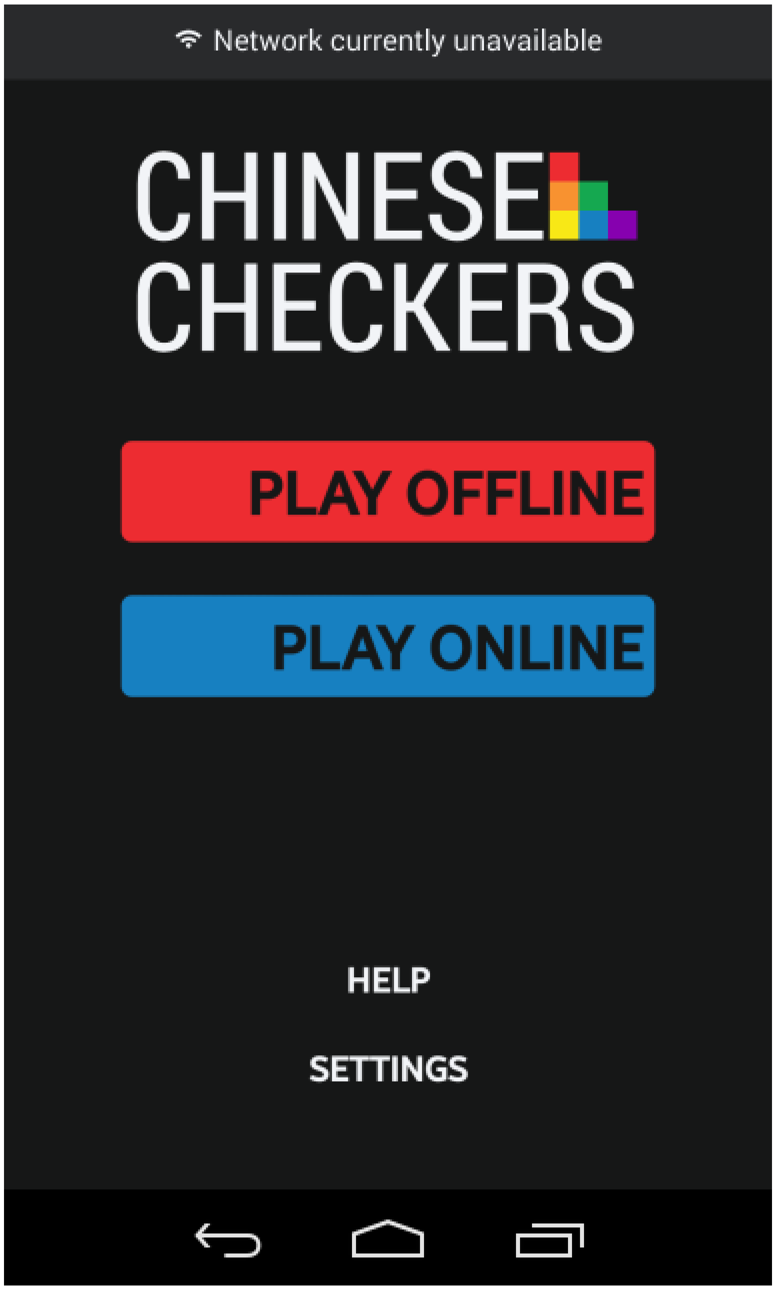
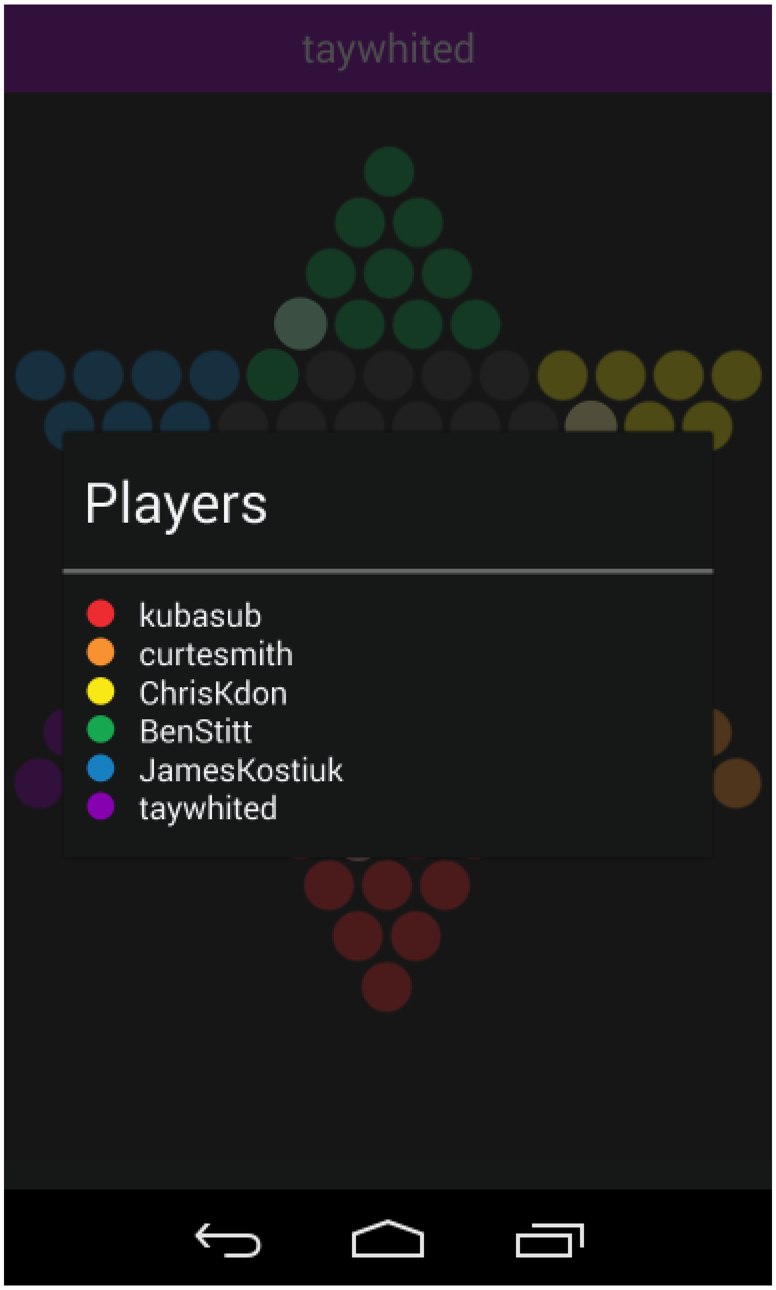


Figure 5.15 Main activity and no connectivity mock-up

5.16 Appendix P


taywhited

Players

● kubasub
● curtesmith
● ChrisKdon
● BenStitt
● JamesKostiuk
● taywhited

Figure 5.16 Player list mock-up

5.17 Appendix Q

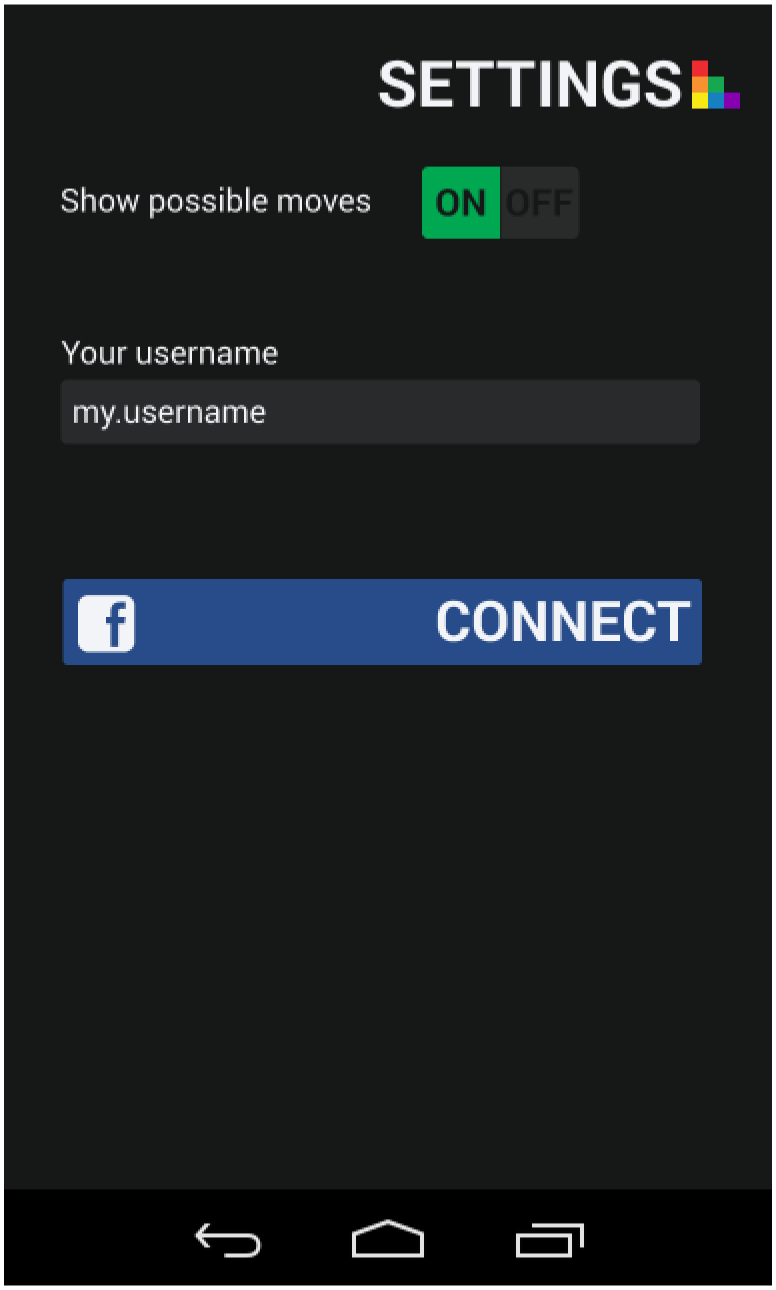
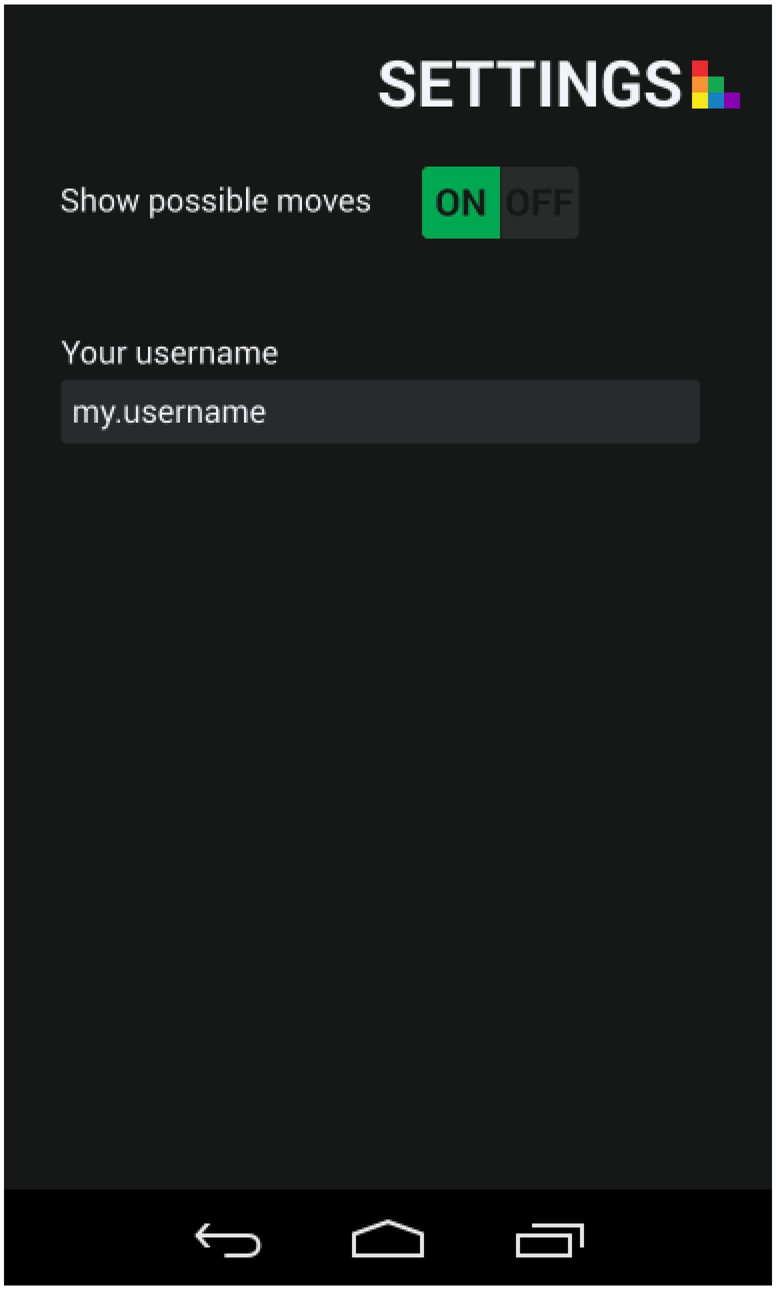


Figure 5.17 Settings activity with Facebook integration mock-up

5.18 Appendix R



Show possible moves ON

Your username

my.username

Figure 5.18 Settings activity without Facebook integration mock-up

5.19 Appendix S

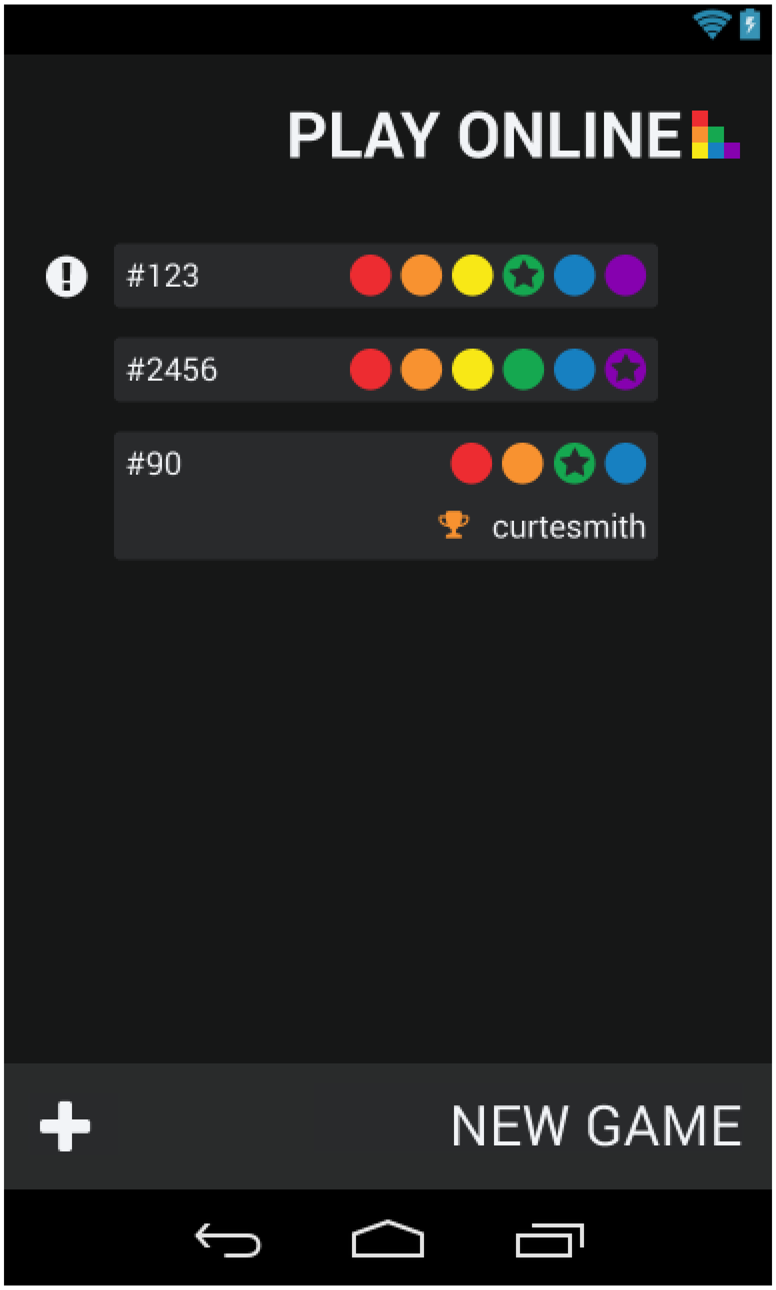


Figure 5.19 Online games list mock-up

5.20 Appendix T

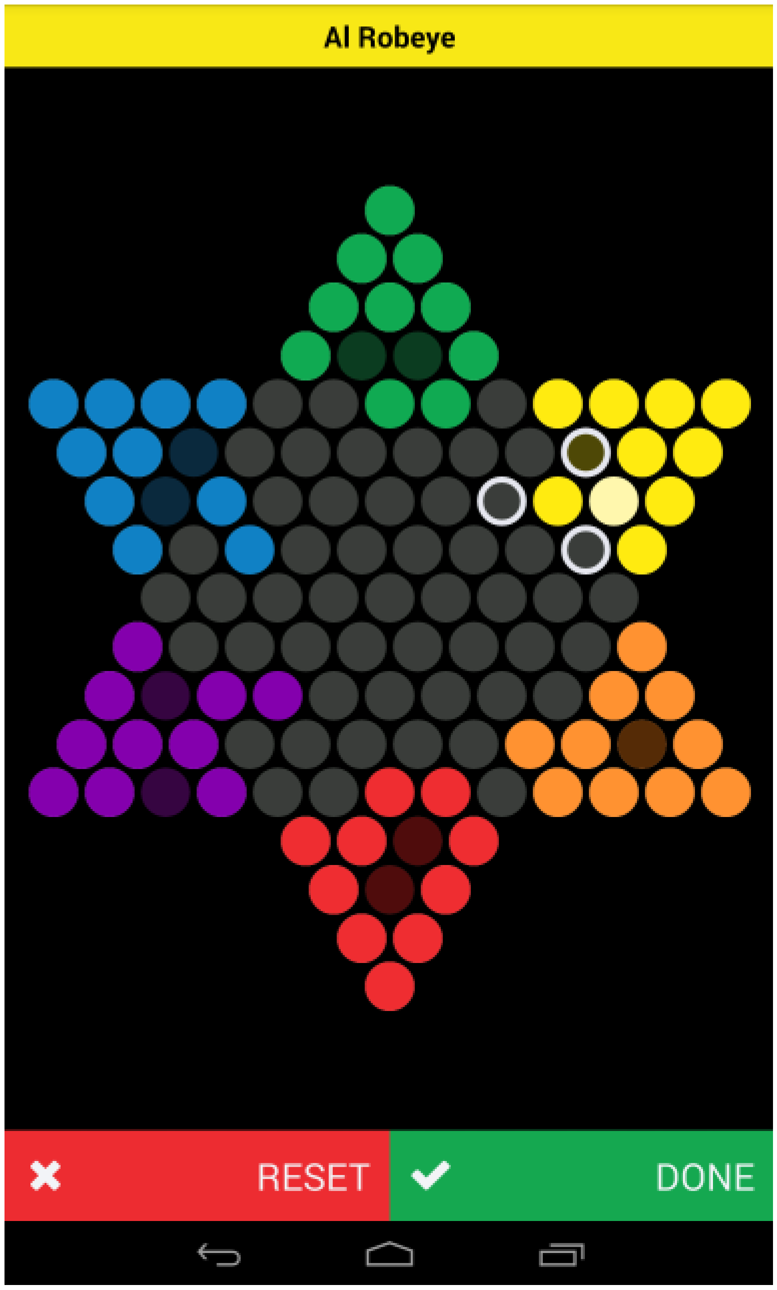


Figure 5.20 Game board mock-up

5.21 Appendix U

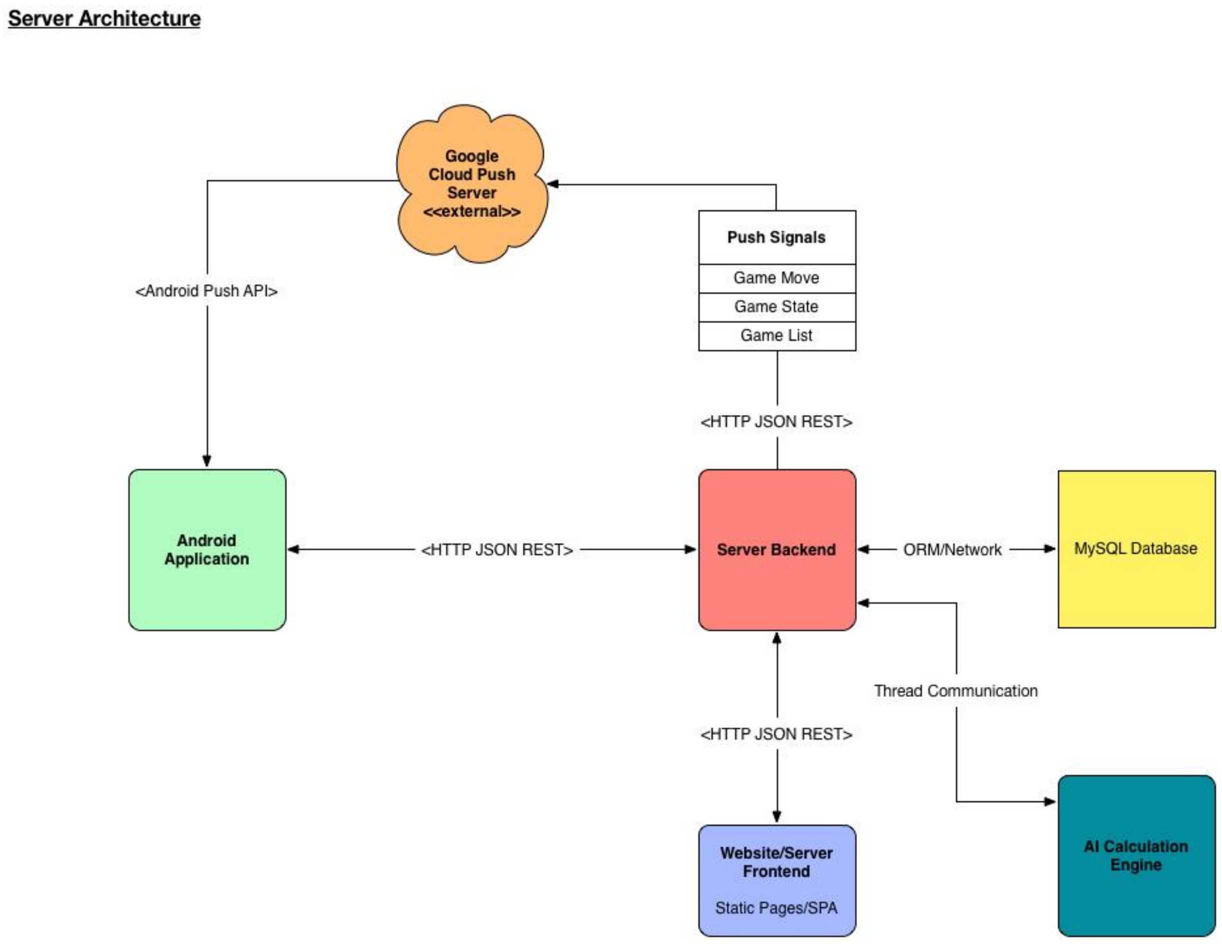


Figure 5.21 Server architecture component diagram

5.22 Appendix V

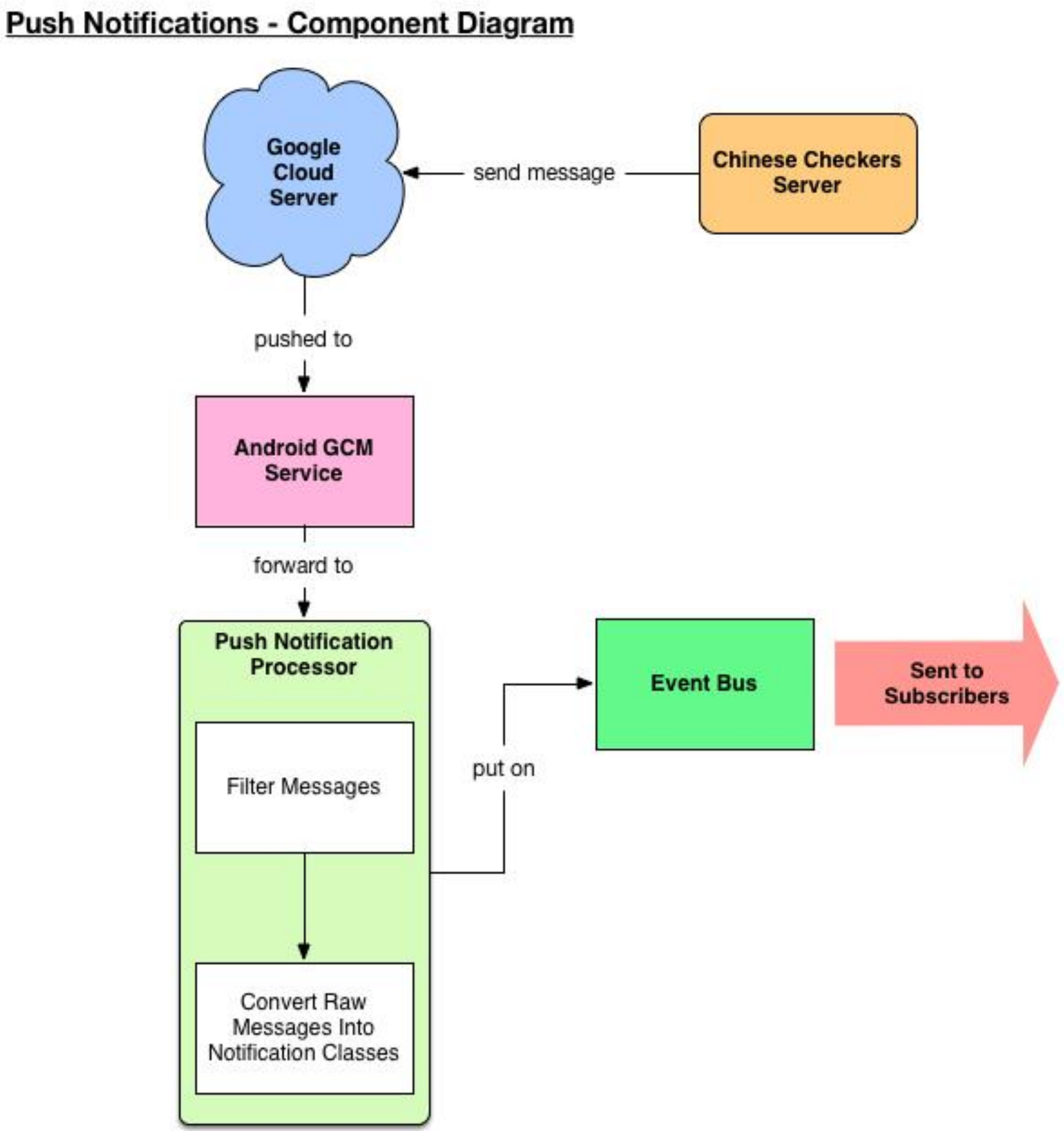
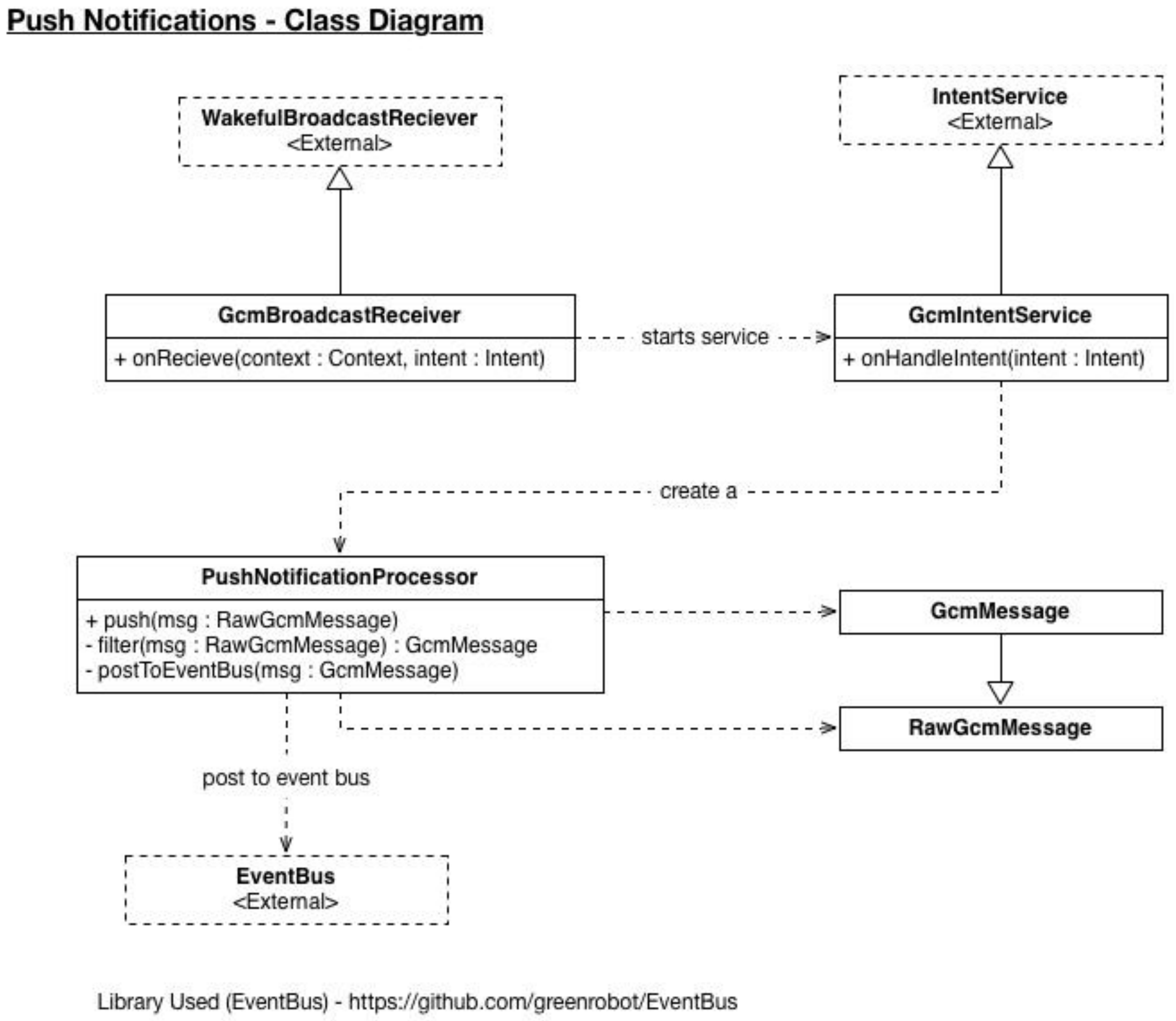


Figure 5.22 Push notification component diagram

5.23 Appendix W



Push Notifications - Class Diagram
WakefulBroadcastReciever
<External>
IntentService
<External>
GcmBroadcastReceiver
+ onRecieve(context : Context, intent : Intent)
starts service
GcmIntentService
+ onHandleIntent(intent : Intent)
create a
PushNotificationProcessor
+ push(msg : RawGcmMessage)
- filter(msg : RawGcmMessage) : GcmMessage
- postToEventBus(msg : GcmMessage)
GcmMessage
RawGcmMessage
post to event bus
EventBus
<External>
Library Used (EventBus) - https://github.com/greenrobot/EventBus

Figure 5.23 Push notifications class diagram

5.24 Appendix X

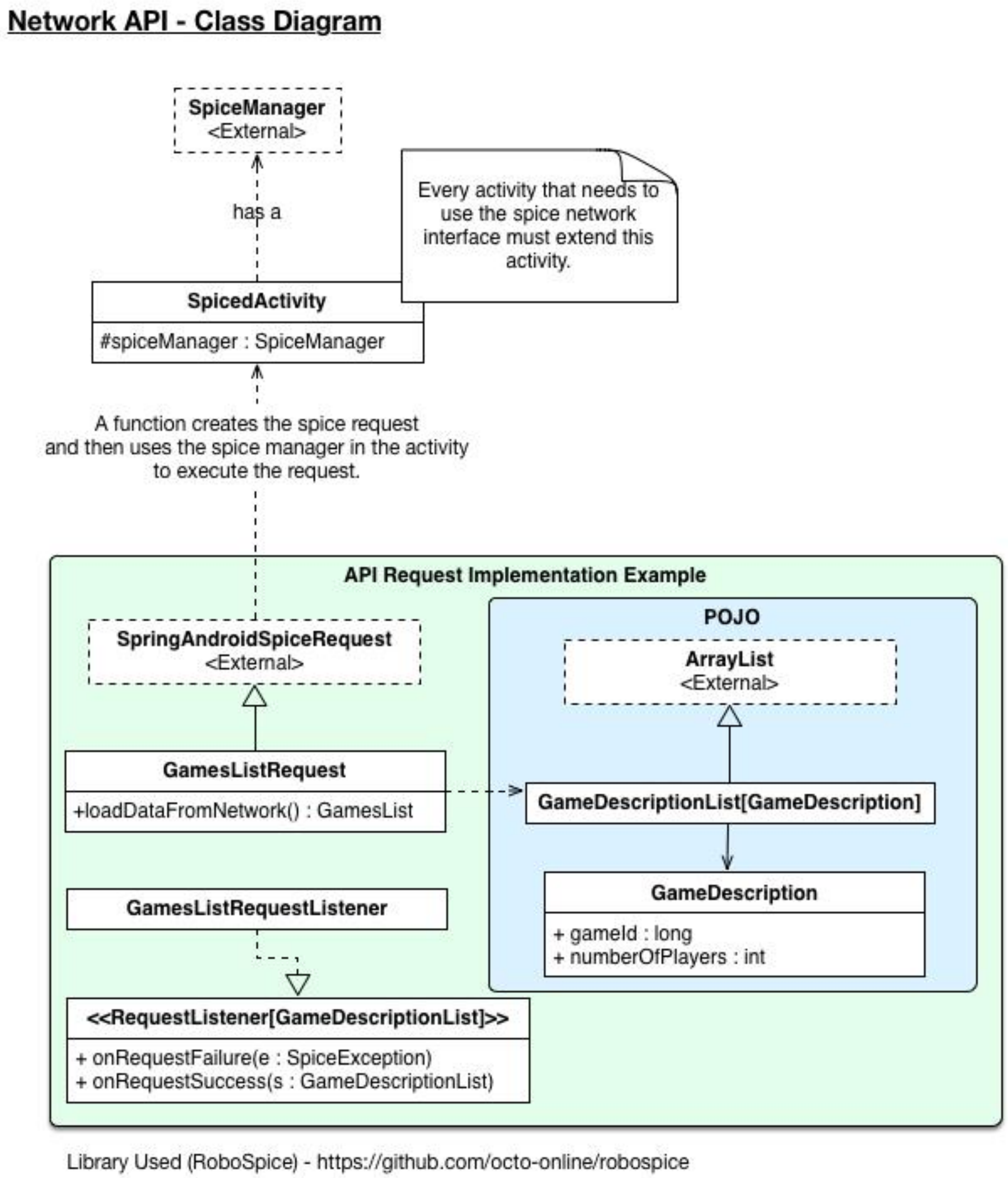

Network API - Class Diagram
SpiceManager
<External>
has a
Every activity that needs to use the spice network interface must extend this activity.
SpicedActivity
#spiceManager : SpiceManager
A function creates the spice request and then uses the spice manager in the activity to execute the request.
API Request Implementation Example
SpringAndroidSpiceRequest
<External>
GamesListRequest
+loadDataFromNetwork() : GamesList
GamesListRequestListener
<<RequestListener[GameDescriptionList]>>
+ onRequestFailure(e : SpiceException)
+ onRequestSuccess(s : GameDescriptionList)
POJO
ArrayList
<External>
GameDescriptionList[GameDescription]
GameDescription
+ gameId : long
+ numberOfPlayers : int
Library Used (RoboSpice) - https://github.com/octo-online/robospice

Figure 5.24 Network API class diagram

5.25 Appendix Y

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: _________________

5.26 Appendix Z

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: ____

____

5.27 Appendix AA

Test feedback form

Project: Chinese Checkers for Android

Project Phase: 3 Test Phase: Regression Testing Date: 3/18/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

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

Test Output (If Applicable): Attached and accounted for 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

5.28 Appendix AB

Test feedback form

Project: Chinese Checkers for Android

Project Phase: 3 Test Phase: Manual Testing – Medium AI Date: 3/19/14

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: 3 Test Phase: Regression Testing – Game Mechanics Date: 3/13/2014

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Still Failing

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:
    - Player pegs may be moved out of opposite triangle

Test Output (If Applicable): N/A

Action Taken: TBD

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: 3 Test Phase: Manual Testing – Difference in AI Date: 3/19/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure): Differences in Easy & Medium AI apparent. Good Job.

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: 3 Test Phase: Manual Testing - Dialogs Date: 3/18/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

Remarks (If fail what caused failure):

Resume Dialog – Pass
End of Game Dialog - 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: 3 Test Phase: Manual Testing - AI Date: 3/13/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Fail

Remarks (If fail what caused failure):

- App crash on starting game full of AI's – 2,3,4,6 Players
- App crash during game of a couple 2,3,4,6 Players

Test Output (If Applicable):

One of the game crashes for each of the modes attached:

- App crash for 5 Ai and 1 player game attached as 6player fail.mp4
- App crash for 3 Ai and 1 player game attached as 4player fail.mp4
- App crash for 2 Ai and 1 player game attached as 3player fail.mp4
- App crash for 1 Ai and 1 player game attached as 2player fail.mp4

Action Taken: TBD

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: 3   Test Phase: Manual Testing - Easy AI Retest Date: 3/16/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Fail

Remarks (If fail what caused failure):

App stalls at the end of game with all AI's.

While playing with AI if an AI wins the next opponent to go is considered the winner.

Easy AI Difficulty too difficult.

Test Output (If Applicable): N/A

Action Taken: TBD

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: 3 Test Phase: Manual Testing - Easy AI Retest 2 Date: 3/19/14

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

5.29 Appendix AC

Test feedback form

Project: Chinese Checkers for Android

Project Phase: 3 Test Phase: Automated UI Unit Date: 3/18/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

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

Test Output (If Applicable): Attached and accounted for 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

Test feedback form

Project: Chinese Checkers for Android

Project Phase: 3 Test Phase: Automated UI Integrated Date: 3/18/14

Tester: Saajid Mohammed

Pass\Fail\Extra Consideration Required: Pass

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

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

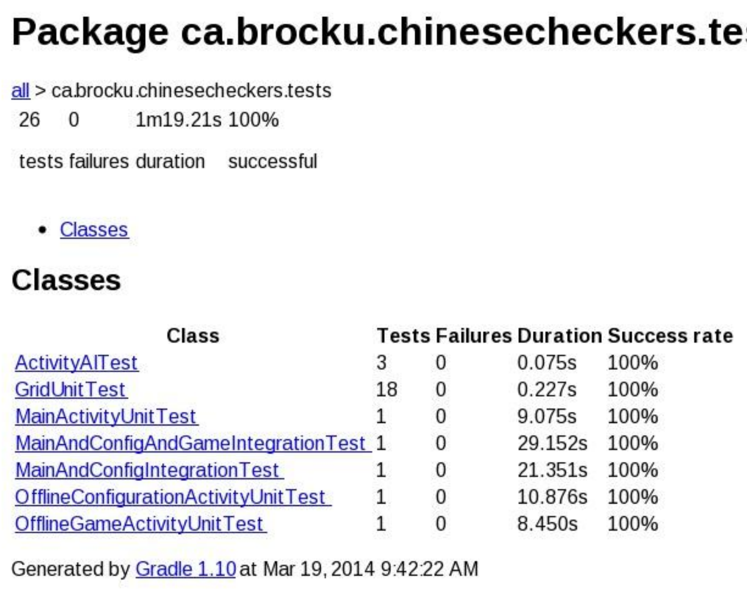
Action Taken: N/A

Tester Signature: ______________ Project Lead Signature: ________________

Test Lead Signature: _______________ Project Lead Name: Curtis Smith

Test Lead Name: Saajid Mohammed

5.30 Appendix AD



Package ca.brocku.chinesecheckers.tests

all > ca.brocku.chinesecheckers.tests

26	0	1m19.21s	100%
tests	failures	duration	successful

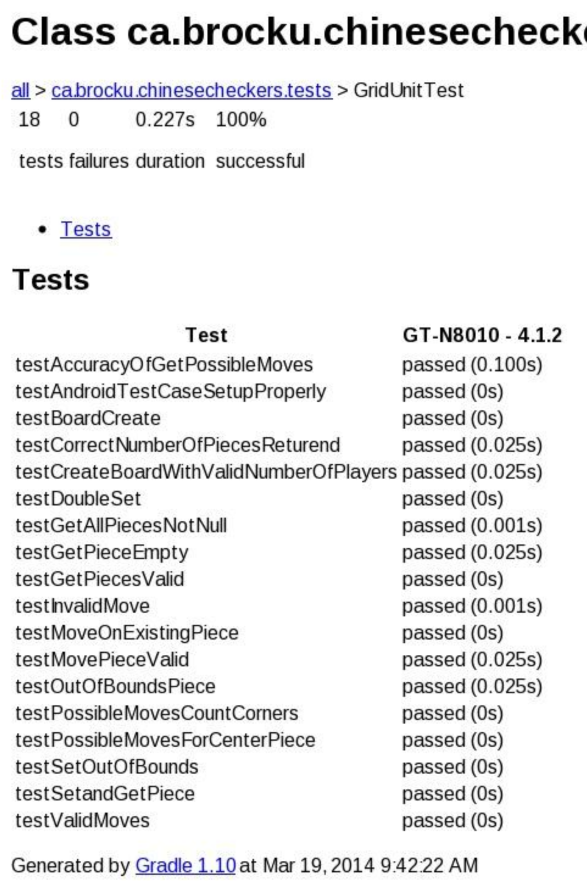
- Classes

Classes

Class	Tests	Failures	Duration	Success rate
ActivityAITest	3	0	0.075s	100%
GridUnitTest	18	0	0.227s	100%
MainActivityUnitTest	1	0	9.075s	100%
MainAndConfigAndGameIntegrationTest	1	0	29.152s	100%
MainAndConfigIntegrationTest	1	0	21.351s	100%
OfflineConfigurationActivityUnitTest	1	0	10.876s	100%
OfflineGameActivityUnitTest	1	0	8.450s	100%

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.31 Appendix AE



Class ca.brocku.chinesecheckers.tests.GridUnitTest

all > ca.brocku.chinesecheckers.tests > GridUnitTest

18	0	0.227s	100%
tests	failures	duration	successful

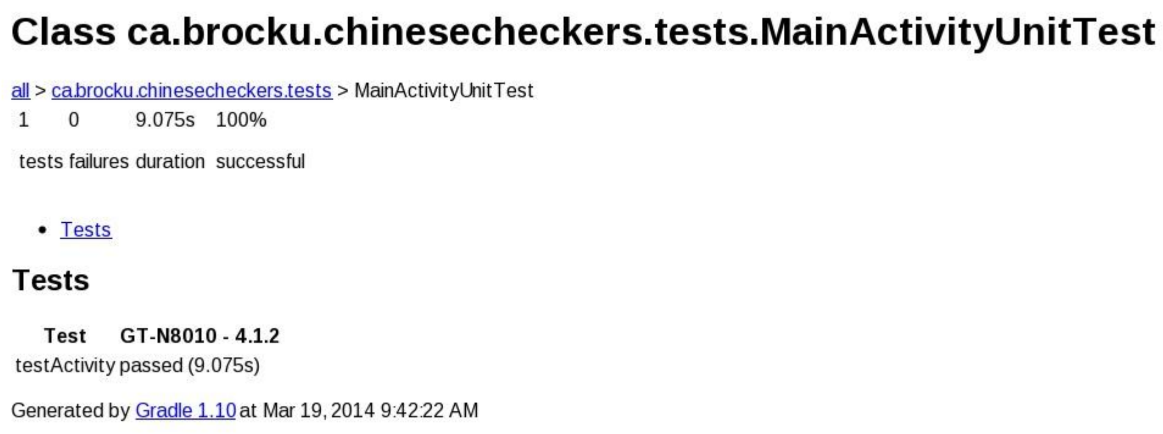
- Tests

Tests

Test	GT-N8010 - 4.1.2
testAccuracyOfGetPossibleMoves	passed (0.100s)
testAndroidTestCaseSetupProperly	passed (0s)
testBoardCreate	passed (0s)
testCorrectNumberOfPiecesReturend	passed (0.025s)
testCreateBoardWithValidNumberOfPlayers	passed (0.025s)
testDoubleSet	passed (0s)
testGetAllPiecesNotNull	passed (0.001s)
testGetPieceEmpty	passed (0.025s)
testGetPiecesValid	passed (0s)
testInvalidMove	passed (0.001s)
testMoveOnExistingPiece	passed (0s)
testMovePieceValid	passed (0.025s)
testOutOfBoundsPiece	passed (0.025s)
testPossibleMovesCountCorners	passed (0s)
testPossibleMovesForCenterPiece	passed (0s)
testSetOutOfBounds	passed (0s)
testSetandGetPiece	passed (0s)
testValidMoves	passed (0s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.32 Appendix AF



Class ca.brocku.chinesecheckers.tests.MainActivityUnitTest

all > ca.brocku.chinesecheckers.tests > MainActivityUnitTest

1	0	9.075s	100%
tests	failures	duration	successful

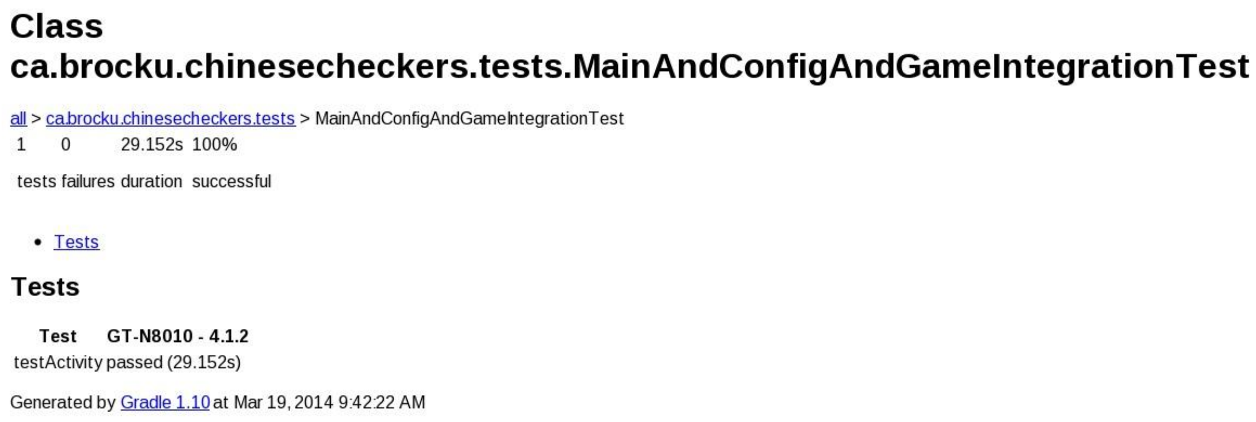
- Tests

Tests

Test	GT-N8010 - 4.1.2
testActivity	passed (9.075s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.33 Appendix AG



Class ca.brocku.chinesecheckers.tests.MainAndConfigAndGameIntegrationTest

all > ca.brocku.chinesecheckers.tests > MainAndConfigAndGameIntegrationTest

tests	failures	duration	successful
1	0	29.152s	100%

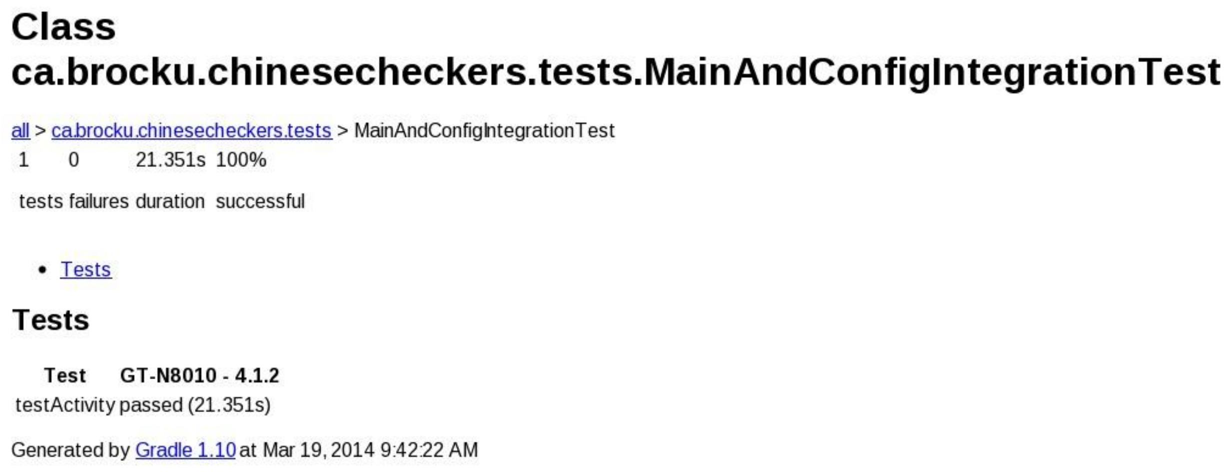
- Tests

Tests

Test	GT-N8010 - 4.1.2
testActivity	passed (29.152s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.34 Appendix AH



Class ca.brocku.chinesecheckers.tests.MainAndConfigIntegrationTest

all > ca.brocku.chinesecheckers.tests > MainAndConfigIntegrationTest

1	0	21.351s	100%
tests	failures	duration	successful

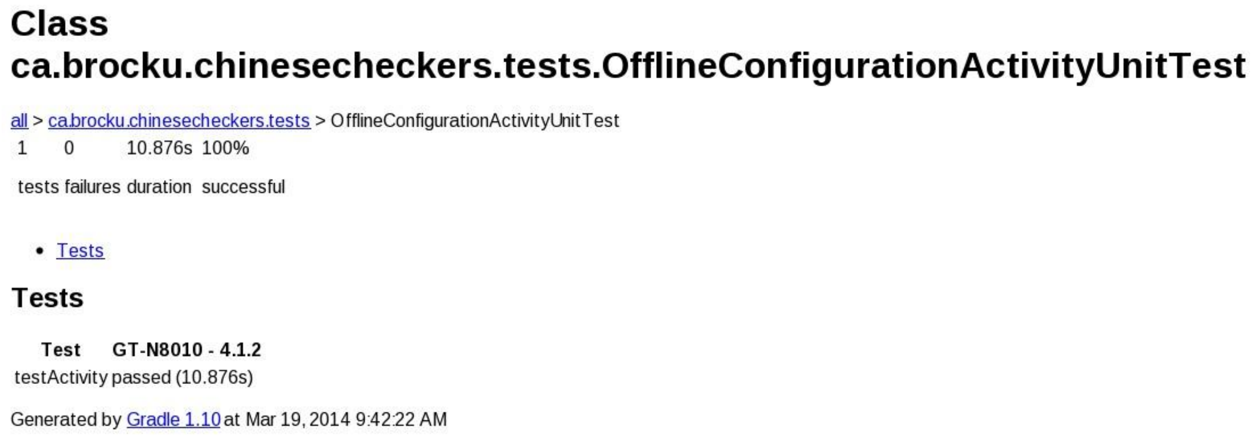
- Tests

Tests

Test	GT-N8010 - 4.1.2
testActivity	passed (21.351s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.35 Appendix AI



Class ca.brocku.chinesecheckers.tests.OfflineConfigurationActivityUnitTest

all > ca.brocku.chinesecheckers.tests > OfflineConfigurationActivityUnitTest

1	0	10.876s	100%
tests	failures	duration	successful

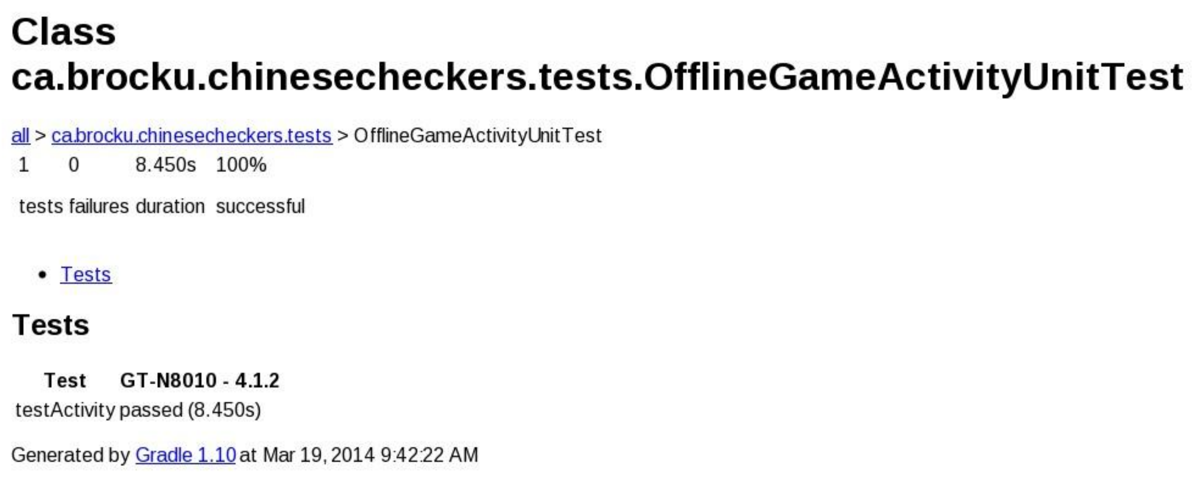
- Tests

Tests

Test	GT-N8010 - 4.1.2
testActivity	passed (10.876s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.36 Appendix AJ



Class ca.brocku.chinesecheckers.tests.OfflineGameActivityUnitTest

all > ca.brocku.chinesecheckers.tests > OfflineGameActivityUnitTest

1	0	8.450s	100%
tests	failures	duration	successful

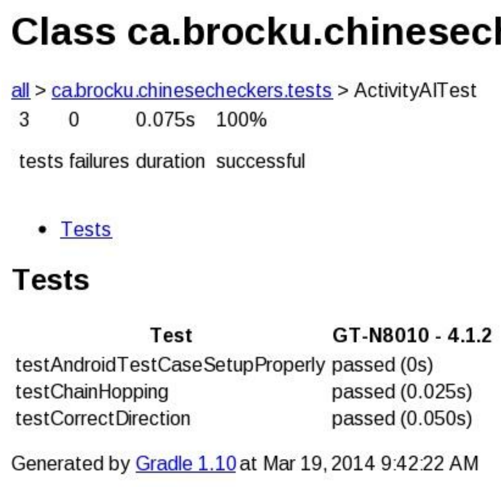
- Tests

Tests

Test	GT-N8010 - 4.1.2
testActivity	passed (8.450s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.37 Appendix AK



Class ca.brocku.chinesecheckers.tests.ActivityAITest

all > ca.brocku.chinesecheckers.tests > ActivityAITest

3	0	0.075s	100%
tests	failures	duration	successful

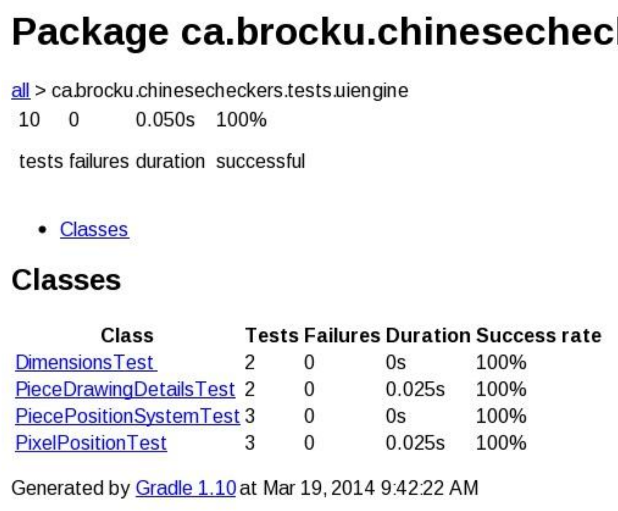
- Tests

Tests

Test	GT-N8010 - 4.1.2
testAndroidTestCaseSetupProperly	passed (0s)
testChainHopping	passed (0.025s)
testCorrectDirection	passed (0.050s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.38 Appendix AL



Package ca.brocku.chinesecheckers.tests.uiengine

all > ca.brocku.chinesecheckers.tests.uiengine

10	0	0.050s	100%
tests	failures	duration	successful

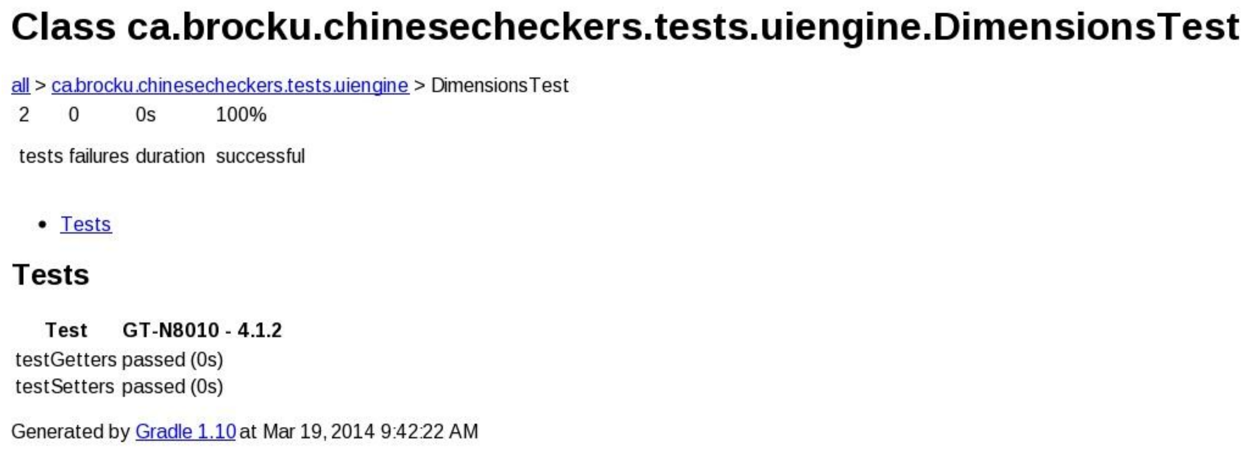
- Classes

Classes

Class	Tests	Failures	Duration	Success rate
DimensionsTest	2	0	0s	100%
PieceDrawingDetailsTest	2	0	0.025s	100%
PiecePositionSystemTest	3	0	0s	100%
PixelPositionTest	3	0	0.025s	100%

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.39 Appendix AM



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

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

2	0	0s	100%
tests	failures	duration	successful

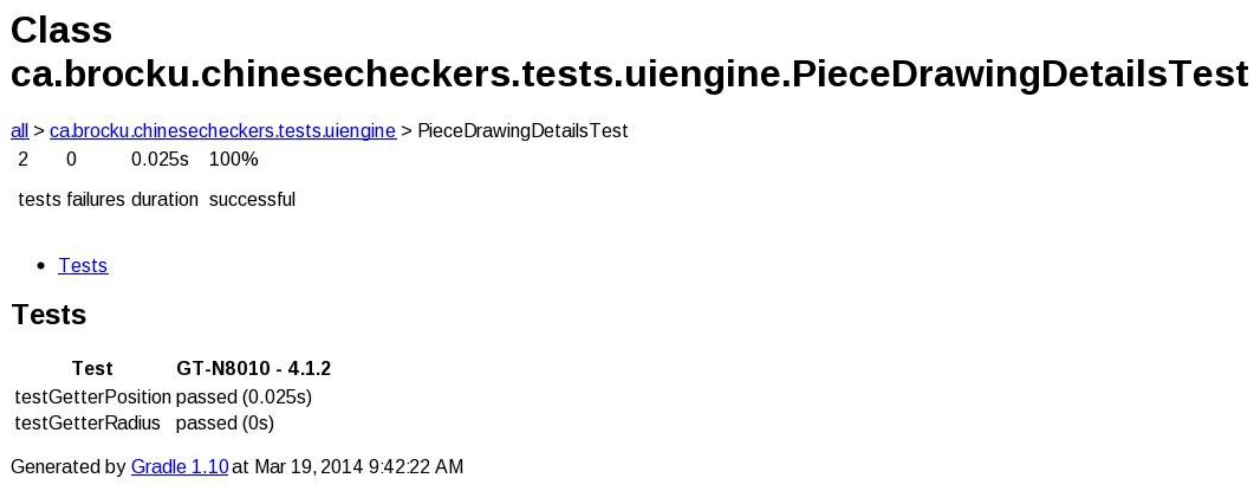
- Tests

Tests

Test	GT-N8010 - 4.1.2
testGetters	passed (0s)
testSetters	passed (0s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.40 Appendix AN



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

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

2	0	0.025s	100%
tests	failures	duration	successful

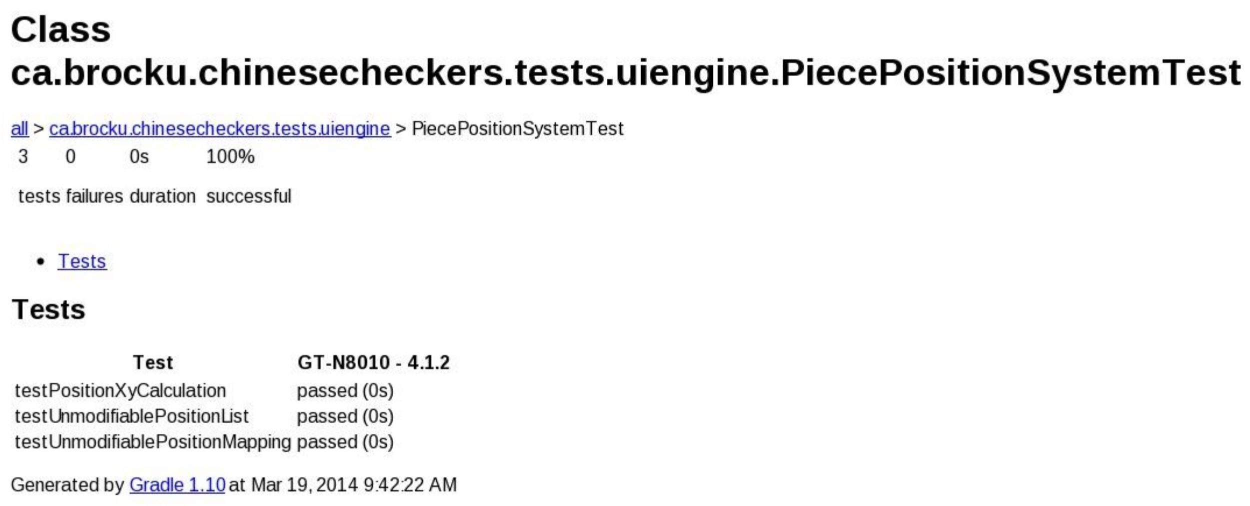
- Tests

Tests

Test	GT-N8010 - 4.1.2
testGetterPosition	passed (0.025s)
testGetterRadius	passed (0s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.41 Appendix AO



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

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

3	0	0s	100%
tests	failures	duration	successful

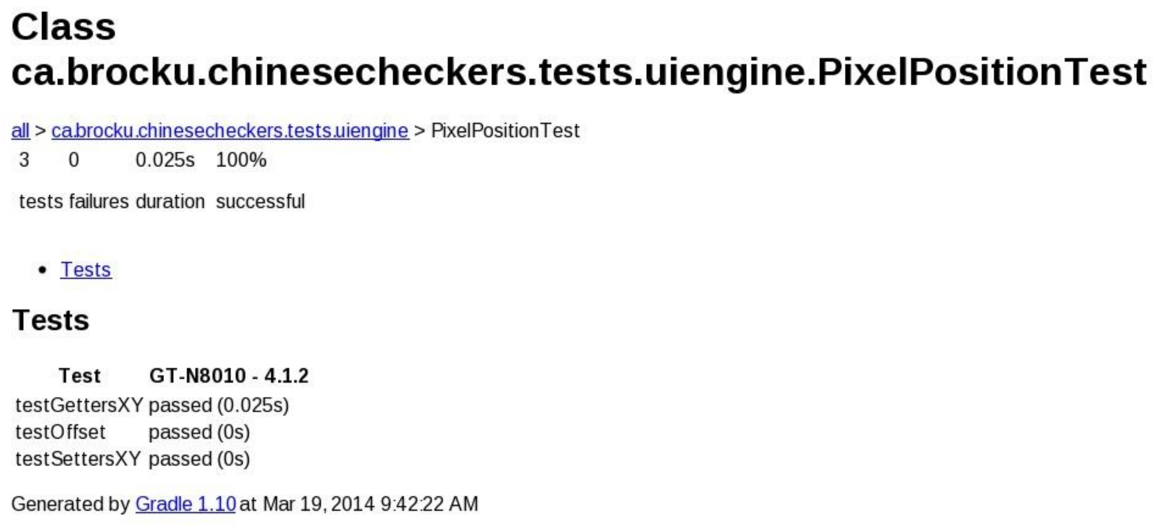
- Tests

Tests

Test	GT-N8010 - 4.1.2
testPositionXyCalculation	passed (0s)
testUnmodifiablePositionList	passed (0s)
testUnmodifiablePositionMapping	passed (0s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.42 Appendix AP



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

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

3	0	0.025s	100%
tests	failures	duration	successful

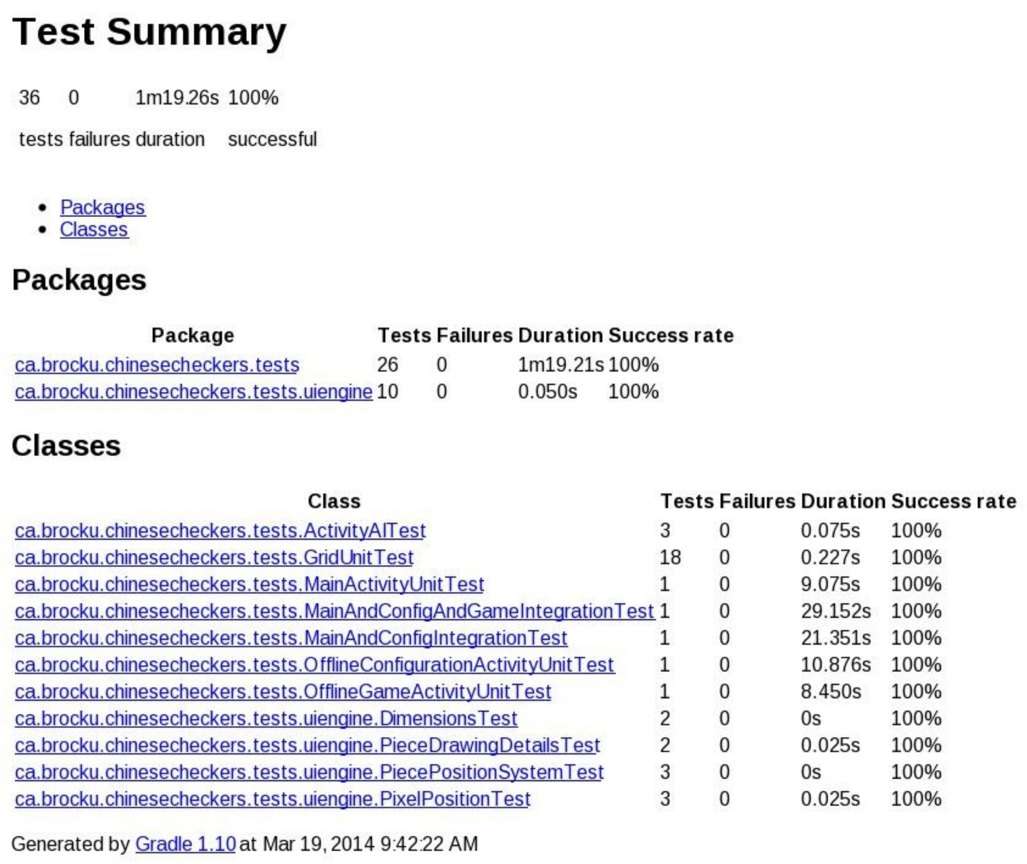
- Tests

Tests

Test	GT-N8010 - 4.1.2
testGettersXY	passed (0.025s)
testOffset	passed (0s)
testSettersXY	passed (0s)

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.43 Appendix AQ



Test Summary

36	0	1m19.26s	100%
tests	failures	duration	successful

- Packages
- Classes

Packages

Package	Tests	Failures	Duration	Success rate
ca.brocku.chinesecheckers.tests	26	0	1m19.21s	100%
ca.brocku.chinesecheckers.tests.uiengine	10	0	0.050s	100%

Classes

Class	Tests	Failures	Duration	Success rate
ca.brocku.chinesecheckers.tests.ActivityAITest	3	0	0.075s	100%
ca.brocku.chinesecheckers.tests.GridUnitTest	18	0	0.227s	100%
ca.brocku.chinesecheckers.tests.MainActivityUnitTest	1	0	9.075s	100%
ca.brocku.chinesecheckers.tests.MainAndConfigAndGameIntegrationTest	1	0	29.152s	100%
ca.brocku.chinesecheckers.tests.MainAndConfigIntegrationTest	1	0	21.351s	100%
ca.brocku.chinesecheckers.tests.OfflineConfigurationActivityUnitTest	1	0	10.876s	100%
ca.brocku.chinesecheckers.tests.OfflineGameActivityUnitTest	1	0	8.450s	100%
ca.brocku.chinesecheckers.tests.uiengine.DimensionsTest	2	0	0s	100%
ca.brocku.chinesecheckers.tests.uiengine.PieceDrawingDetailsTest	2	0	0.025s	100%
ca.brocku.chinesecheckers.tests.uiengine.PiecePositionSystemTest	3	0	0s	100%
ca.brocku.chinesecheckers.tests.uiengine.PixelPositionTest	3	0	0.025s	100%

Generated by Gradle 1.10 at Mar 19, 2014 9:42:22 AM

5.44 Appendix AR

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: 3

Task Remarks:

Task 2 Score: 6

Task Remarks: noticeable difference in difficulty

Task 3 Score: 3

Task Remarks: didn’t really know where to go

Task 4 Score: 3

Task Remarks: useful

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: 3

Task Remarks: I won the game, despite having less than fantastic skills. The Easy AI seemed fair.

Task 2 Score: 7.5

Task Remarks: This game started out more difficult than the last one. Unfortunately the AI started repeating a move over and over again instead of winning the game. I ended up beating the AI even though there was no possible way that I should have. I do not understand why this happened.

Task 3 Score: 2

Task Remarks: Worked

Task 4 Score: 4

Task Remarks: Help button was difficult to locate from within the game. The frequently asked questions were useful and well explained.

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: 3

Task Remarks: He doesn’t take the shortcuts. I can always win against him.

Task 2 Score: 5

Task Remarks: He could’ve won, but he kept moving back and forth with his last piece.

Task 3 Score: 0

Task Remarks:

Task 4 Score: 8

Task Remarks:

General Remarks: I enjoyed learning how to play!

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: _5____

Task Remarks: __I almost won ____

____

Task 2 Score: _7____

Task Remarks: __I got beat very quickly ____

____

Task 3 Score: __4___

Task Remarks: __it would have been easier if there was a "clear" button to remove all of the letters of my username so that I could change it ____

____

Task 4 Score: _10____

Task Remarks: _This would have been easier for me if there were less words and more pictures ____

____

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: _8____

Task Remarks: __It beat me and I only got 4 of my pegs all of the way to the end zone, ____

____

Task 2 Score: _8____

Task Remarks: __Same result as with the easy robot. I only got 4 of my pegs to the end zone. ____

____

Task 3 Score: _3____

Task Remarks: _Pretty easy to do. ____

____

Task 4 Score: __5___

Task Remarks: _Too much reading. ____

____

General Remarks: ____

____

6 References

The Rules of Chinese Checkers. (2012). Masters Traditional Games. Retrieved March 17, 2014, from http://www.mastersgames.com/rules/chinese-checkers-rules.htm

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