TDT4240

Software Architecture

Spring 2012

Group A17
Android SDK

Steinar Haram
Emil Andreas Mork
Stian Sørebø
Ole Jørgen Rishoff

Implementation Document

Primary focus:
Maintainability

Secondary focus:
Usability

Contents

List of Tables

List of Figures

1 Introduction

This document contains implementation details for our developed version of the classical *Nine Men's Morris* game. The game is developed as a native Android application. The applications primary attribute is modifiability, while its secondary attribute is usability. The second chapter will highlight the design and implementation details. The following chapter contains a user manual, while chapter four contains a brief description of the testing of functional and quality requirements. The relation between the implementation and the planned architecture will be reviewed in chapter five. Chapter six highlights encountered problems and gained experience.

2 Design and implementation details

2.1 Skiller multiplayer framework

Due to the desire of developing a fully functional multiplayer game, the *Skiller multiplayer framework* [?] has been used. This is a third party COTS software, and its usage has sped up the development process. Registration was needed in order to gain access to the Skiller SDK. When the registration was done, a new game could be created, and an application ID, an application key, and an application secret was supplied. These are used in the code to identify the specific application.

This framework supplies a server solution for turn based games, and it has been implemented in the network class. When playing a network game, the GameController class tells the Game model to network class sends event messages to the server, and the server delivers it to the opponent.

2.2 Activities

Figure 1 shows an overview of the application's different activities, and how the user interactions can change them. The user can navigates back to the menuActivity by pressing the back button.

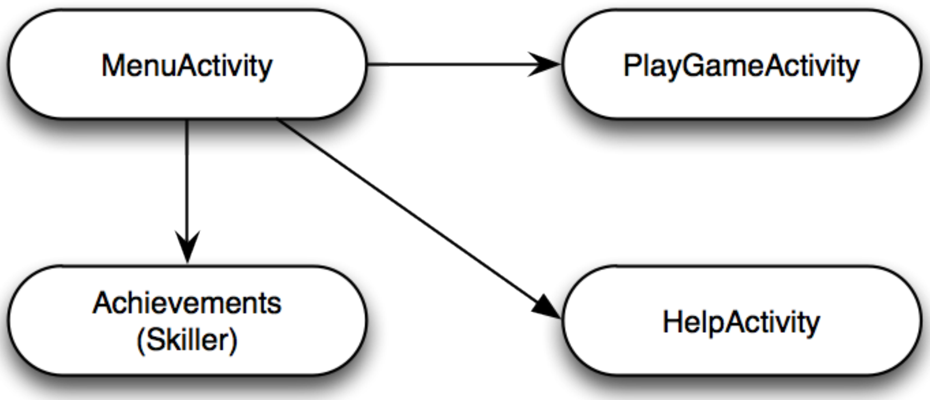


Figure 1: Application activities

The MenuActivity shows a menu consisting of five items, allowing the user to create or join a multiplayer game, start a local game, check achievements, or check the game rules. The PlayGameActivity is responsible for creating or joining multiplayer games, and starting local games. The HelpActivity is responsible for showing the game rules. The achievement screen is supplied by *Skiller*.

Screencaps of the different activities, and the achievement screen, are shown in section 3.5.

2.3 MVC

Our implementation of the MVC structure is based interface communication. In regular Java it is more common to create a MVC structure where the view has model and the controller contains view and model. Android has its own MVC structure where an activity represent controller and view, which makes it difficult to build ut the MVC in a regular way. The most favorable way of createing MVC in android is to use interfaces to update the view when there are changes in the model. Figure 2 shows an overview of the MVC structure, with BoardView (View), GameController (Controller) and Game (Model). When the user interact with the view the controller handles the user action and tells the model what to do. When the model is updated, the view receive a message via the interface, and updates the screen.

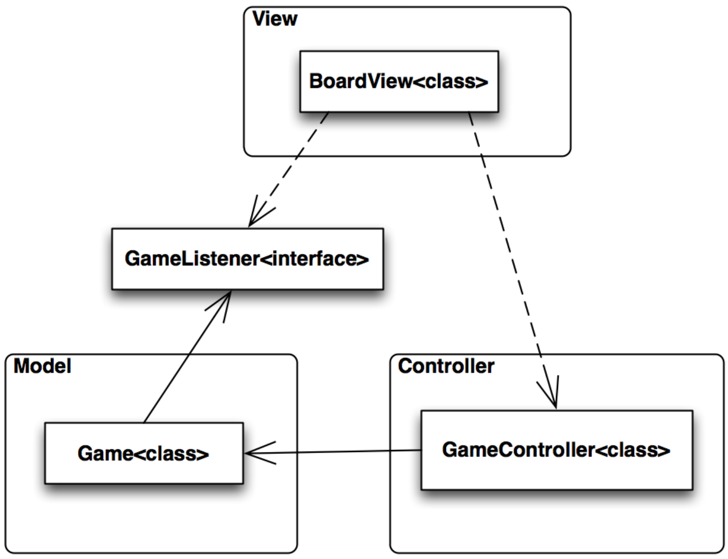


Figure 2: MVC structure

2.4 States

The Game class is implemented with associated states. The states change on runtime depending on player interaction. All states have their own way of controlling user possibilities, and telling the user what to do while playing. Figure 3 shows the relation between the Game class, the State interface, and the context classes implementing the State interface.

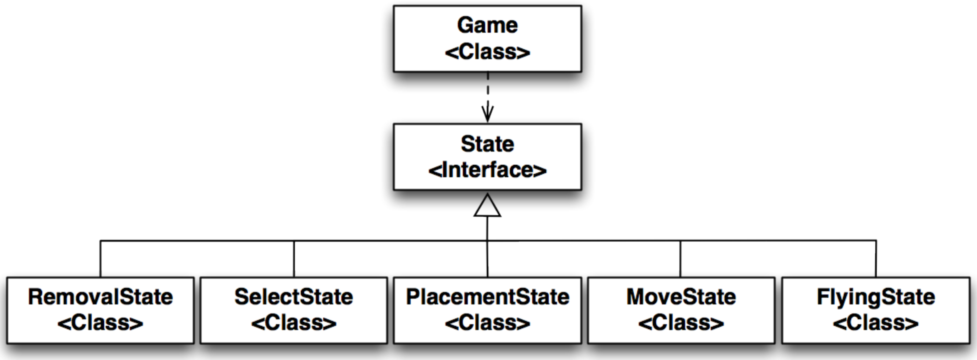


Figure 3: State structure

3 User manual

3.1 Functional requirements

- Requires the user to have a android device with Android OS v2.2 or newer.
- Requires Internet to enable online play.

3.2 Running the application

The application is avalible at URL.
The Eclipse-project is avalible at ANOTHER URL.

3.2.1 Emulator

To run the application in the emulator, the user needs to open the project in Eclipse. File -> Open project -> Existing source code -> path to downloaded project.

3.2.2 On Android device

Method 1 To run the application on the Android device, the user needs to open the project in Eclipse. File -> Open project -> Existing source code -> path to downloaded project. Then connect the Android device with an USB cable to the PC, and run the project.

Method 2 Download the apk file form URL, transfer to the android device with an USB cable and explore and install the application on the device.

3.3 Game rules

The game is implemented with the same set of rules as the classic board game *Nine Men's Morris* [**?**]. The goal of the game is to either block any opponent moves, or to reduce your opponent's piece number to less than three. If you get three pieces in a row, you enter a morris state, and are allowed to remove one of your opponent's pieces. Pieces that are in a morris state, i.e. forms three in a row either horizontally or vertically, are not removable.

3.4 Creating Skiller account

The first thing that meets the user after starting the application for the first time, is a Skiller dialogue asking the user to create a Skiller account. This account will later be used for playing the game online.

3.5 How to play

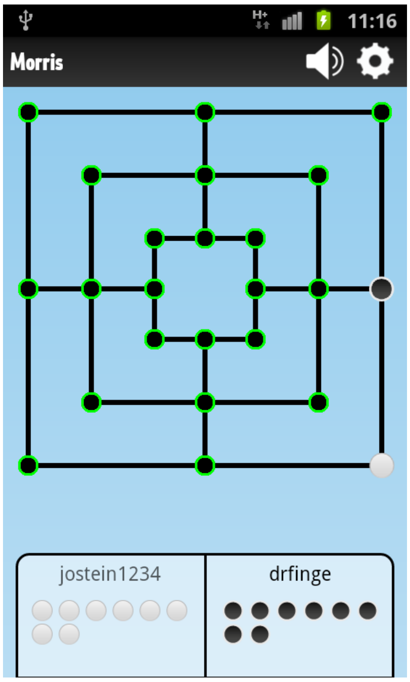
3.5.1 Choosing game mode

A user can choose between online mode or hotseat mode. Clicking "Crate Game" or "Join Game" will start a game in online mode. Clicking "Hotseat" will start a game in hotseat mode.

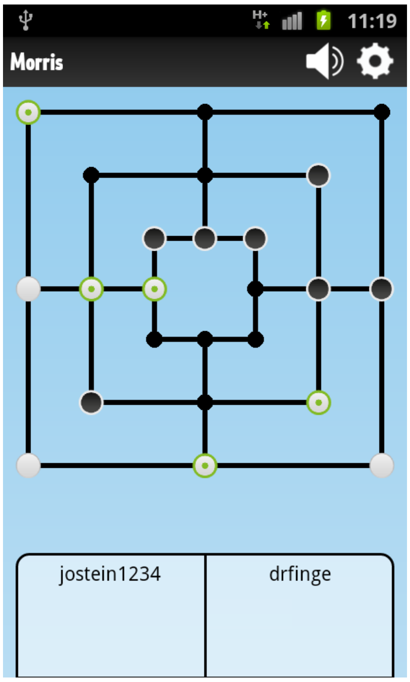
(a) Available game modes (b) Achievements (c) Help explaining the games rules

3.5.2 Placing, selecting, moving, and removing pieces

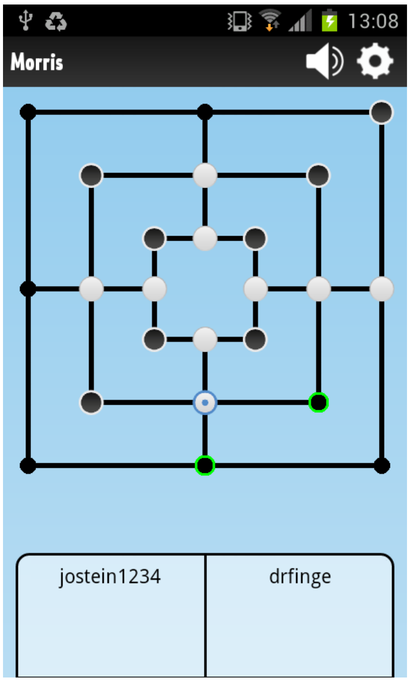
When it is your turn to move, either the board or your pieces will be highlighted. In addition, the name of the current player will be blinking as the game progresses.



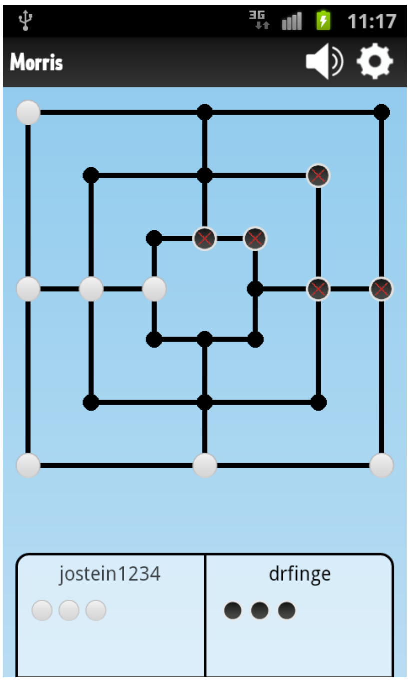
(d) Green indicator shows where you can place a piece



(e) Highlights selectable pieces



(f) Highlight selected piece, green indicator on possible moves



(g) Highlights removable pieces with a red cross

3.5.3 Hotseat mode

If you start a local game as described in section 3.5.1, you can control both players from the same device.

3.5.4 Online mode

If you start an online game as described in section 3.5.1, you are taken to the board screen, and need to wait for another player to join your game. The guest, i.e. the one who joins the game, will get the initial move. Your own pieces will always be white.

4 Test report

The report should contain test reports for both functional requirements and quality requirements (quality scenarios).

4.1 Functional requirements testing

FR1 - Placement of pieces	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	5 minutes
Evaluation	The players successfully placed all nine pieces.

Table 1: Testing of FR1

FR2 - Moving pieces	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	3 minutes
Evaluation	The players successfully moved their pieces one length at the time.

Table 2: Testing of FR2

FR3 - Morris state	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	3 minutes
Evaluation	When placing three pieces in a row, the game successfully changed state, and a piece was removed from the opponent.

Table 3: Testing of FR3

FR4 - Flying pieces	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	3 minutes
Evaluation	When the player had three pieces left, the game successfully changed state to Flying state, and the player was allowed to move to any vacant field.

Table 4: Testing of FR4

FR5 - Multiplayer	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	10 minutes
Evaluation	Ole and Emil connected to each other via the Skiller framework, and successfully played a whole game.

Table 5: Testing of FR5

FR6 - Game board	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	1 minute
Evaluation	The game has a board conforming with the layout of *Nine Men's Morris*.

Table 6: Testing of FR6

FR7 - Setting player name	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	5 minutes
Evaluation	A player can set his own name when creating a Skiller account.

Table 7: Testing of FR7

FR8 - Denied Morris state	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	10 minutes
Evaluation	The game automatically ends a players turn when he enters a Morris state, and all the opponents pieces also is in a Morris state

Table 8: Testing of FR8

FR9 - Game over	
Executor	Ole Jørgen Rishoff
Date	12.04.2012
Time used	5 minutes
Evaluation	When a player has only two pieces left, or cannot move any of his or her pieces, the game successfully ends.

Table 9: Testing of FR9

4.2 Quality requirements testing

M1 - Add new game variant	
Executor	Stian Sørebø
Date	27.04.2012
Stimuli	Addition of a new game variant
Expected response	Implementing and testing within 10 hours
Observed response	No more time to extend the program.
Evaluation	Not implemented

Table 10: Testing of M1

M2 - Local hotseat	
Executor	Stian Sørebø
Date	27.04.2012
Stimuli	Addition of a game that runs locally on the phone
Expected response	Implementing and testing within 10 hours
Observed response	Implemented and tested within 8 hours.
Evaluation	Players are able to play the game locally on the phone, successful.

Table 11: Testing of M2

M3 - Chat	
Executor	Stian Sørebøf
Date	27.04.2012
Stimuli	Addition of a in-game chat
Expected response	Implementing and testing within 12 hours
Observed response	No more time to extend the program.
Evaluation	Not implemented

Table 12: Testing of M3

M4 - Time constraint	
Executor	Stian Sørebø
Date	12.04.2012
Stimuli	If no response is given from a player within 15 seconds, they should automatically loose the game.
Expected response	Implementing and testing within 4 hours
Observed response	Implemented the time constraint in 2 hours
Evaluation	Player not responding, will gain a loss after 15 seconds. Successful

Table 13: Testing of M4

U1 - Learning game rules	
Executor	Stian Sørebø
Date	23.04.2012
Stimuli	Addition of visible game rules, reachable from the menu.
Expected response	A player unfamiliar with the game rules should improve his placement time with 30%.
Observed response	The user was placing his pieces close to double speed as before.
Evaluation	Successful, results may vary from testsubjects

Table 14: Testing of U1

U2 -	
Executor	Stian Sørebø
Date	23.04.2012
Stimuli	Possible moves are highlighted for both players
Expected response	A player should improve his decision time by 20% when making moves.
Observed response	The user was reducing his decision making time by $^{1}/_{3}$.
Evaluation	Successful, results may vary from testsubjects

Table 15: Testing of U2

T1 - T1	
Executor	Stian Sørebø
Date	12.04.2012
Stimuli	Connection should be stable and fast.
Expected response	A player joins an open game, and should be ingame within 10 seconds
Observed response	10 out of 10 attemps joined in under 10 seconds.
Evaluation	Successful, 100% positive result.

Table 16: Testing of T1

5 Relations to the architecture

This section should list the inconsistencies between your architecture and the implementation. Give the reasons for these inconsistencies. Discuss whether they could have been discovered at an earlier point, for instance during the ATAM evaluation.

6 Issues

It turned out that the Skiller framework was poorly documented and it gave some unreadable exception messages. This slowed down the testing quite a bit. In addition, because we normally have only had two Android devices at our disposal, much of the testing have been done in the emulator. This is of course not very effective. Also, when running the application in the emulator and creating games, players would automatically join without user interaction. The Skiller team was unable to give us any good answers to why we experienced this problem.

We underestimated the importance of groupwide understanding of the framework, and the implementation should have been done collectively. As it was, we assigned one person to this task, and the rest of the group were unable to do anything with the framework in his absence. We also should have a done a more thourogh research regarding the use of the framework and its documentation.

6.1 Gained experieces

In future projects we will spend more time researching possibilities when choosing to work with a third party framework. We did not look into the documentation of the framework before implementing it, and if we had, we would probably have chosen a different framework. We will also perform a quick search for problems related to it before making a final decision.

Some members of the group went into this project with more experience than others, developing native Android applications. Due to good communication we have been able to use this experience to our advantage.

We have gotten a better understanding of the patterns we chose to implement. The experience of implementing a MVC pattern in a native Android application has been very useful. The state pattern and its usage was unknown for all of us, but we now have a good understanding of how it can be used. We could perhaps made this pattern a bit more clear, and assign more responsibilities to the different states.