U08928 Mobile Software Assignment Part 1
Design

Objective:

The objective of this coursework is to design and implement an android version of the game “Reversi” or “Othello”. The application must conform to the specification on the module website.

Design considerations:

Low level considerations:

When designing this app I had to consider a range of factors. Putting aside the user interface and navigation side of the design and focusing on the low level workings of the app I considered the following factors:

1. Performance considerations on a mobile device
2. Ease of writing the code
3. Ease of maintenance of the code (also legibility of the code)

The first consideration when designing this app is the performance. The devices that could be running this applications may have severe restrictions on memory, screen resolution, screen size and processing power. All mobile devices have limited battery life to consider also. In my app I will aim to do the following things to minimize load on system resources:

1. Where possible variables will be accessed directly, ie, wont use get() functions.
2. Avoid using 2D arrays, for the main board I will use a 1D array and use the modulo operator.
3. Keep objects to a minimum where possible
4. Use threads for any computation, ie, if I need to look through the status of the board I will do it in a separate thread to the main thread.

With all the above taken into account, this app is relatively simple and will probably not be to strenuous on the system resources unless I am very careless. In this instance I should be able to apply some conventional software engineering principles to keep the code easy to understand and maintain.

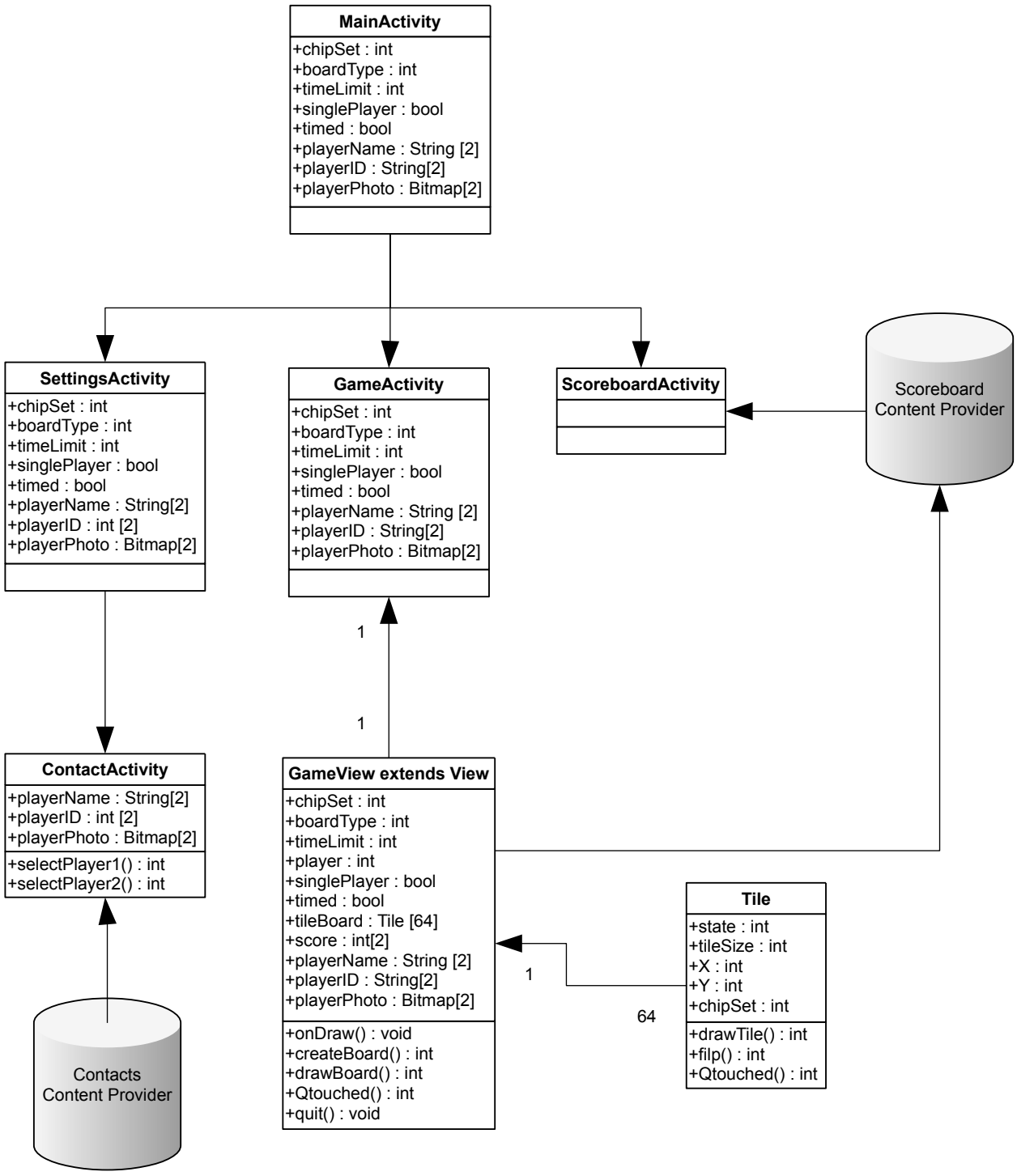
User interface considerations:

Aside from the low level functional considerations I must also consider how the user is going to interact with the application. No matter how good the underlying code is, if the user finds the app hard to use or unresponsive the app will fail. With this in mind I have chosen to implement the main board on a canvas. This will give me greater flexibility than the standard UI toolkit, that being said, I still need to make sure the UI I create is intuitive.

Using the canvas as the basis for the app I can give the app its own look and feel, I have control over how nice the app looks.

I need to make sure the application is responsive. On the low level side this will be taken care of using threads, on the UI level this can be taken care with spinners and animations. I will aim to use animations on the main board game to keep the user informed of the progress, ie, the chips will be animated as will the CPU player moves.

Design:
With the above considerations I have come up with the following initial design:



Activity / class descriptions:

Main Activity:
This is the main menu for the application and is the top activity. From this activity you will be able to access the settings activity, the main game activity and the scoreboard activity. The variables in this activity will be a default value until they are overwritten by values passed from the settings activity in an explicit intent.

Settings Activity:
This will be the settings menu where you will be able to do the following:

1. Choose a chip set (aesthetic).
2. Choose a board type to play on (aesthetic).
3. Choose if the game is timed or not.
4. Choose the time limit per move if the game is timed.
5. Choose if the game is single player or two player.
6. Select player details from the device contacts content provider

This activity will have the following variables which will be passed back to the main activity via an explicit intent when the settings page is exited:

int chipSet -This will determine what chip set is used.
0 = Standard
1 = Alternate
2 = Any other alternate

int boardType -This will determine the board type, much like the chipSet variable.
0 = Standard
1 = Alternate
2 = Any other alternate

int timeLimit -If the game is a timed game this will be the time limit per turn in seconds.

boolean singlePlayer -This determines if the game is single player or multi player.
True = Single Player
False = Two Player

boolean timed -This determines if each turn has a set time limit.
True = Game is timed
False = Game is not timed

String[2] playerName-This contains player 1 and player 2 names.

int[2] playerID -This contains player 1 and player 2 ID numbers from the content provider.

Bitmap[2] playerPhoto-This contains both players photos in a bitmap.

Game Activity:
This is the main functional activity that actually starts the game. It has one game view which it sets focus to. The variables in this activity are taken from the main activity via an explicit intent. These are either at the default values or the modified values if the settings activity has been used.

Game View:
This will either be a Surface View or a standard View. I haven't tried implementing the Surface View yet so I'm not sure at this stage. This view is where the game is played. It will use a canvas to draw the board and implements the onTouch listener to take action when the canvas is touched. It will contain an array of Tiles in a 64 element array, these tiles are the tiles on the board and store their own coordinates and have their own functionality, see the Tile class description to follow. The Game View has the following variables:

Tile[64] tileBoard -This is the array of tiles that make up the board.

Int[2] score -This array keeps the current score for all players. This will be written to a bespoke content provider.

This view also has access to all variables from the parent activity. This view has the following functionality:

void onDraw() -This draws the initial board and is called when the invalidate() method is called. It it therefore responsible for updating the status of the board after a move.

int createBoard() -This populates the tileBoard array and is responsible for giving each tile its coordinates.

int drawBoard() -This runs through the tileBoard array and runs each tiles drawTile() method on each tile.

int Qtouched() -This is called from the onTouch listener and will check which tiles have been touched and decide what other tiles need to be flipped as a result.

void quit() -This method is called when the game ends and the user decides to quit or when the user exits to the main menu.

Tile:
This is a simple object that represent a tile of the board.It stores it own coordinates and tells if its been touched when queried. It also draws itself, the chip that's on it and stores what state it is in, ie, free, potential move, player 1 chip or player 2 chip. It has the following variables:

int state -This store what state the chip is in:
0 = empty
1 = Player 1 chip
2 = Player 2 chip
3 = Potential next move

int tileSize	-This is the width and length of the tile, the tile is a square. This will be scaled using the canvas.getWidth() method so should scale for all devices.
int x	-This is the X coordinate for the top left of the tile
int y	-This is the Y coordinate for the top left of the tile
int chipSet	-This will determine what chip will be used, this will most likely point to a drawable.

The tile has some basic functionality, this is to aid code maintenance and should really have much of an effect on performance as its pretty basic:

int drawTile()	-This simply draws the tile using canvas.drawRect(). It will also draw the chip depending on the state variable.
int flip()	-This method flips the tile by toggling the state between 1 and 2. This is used for automated flipping.
int Qtouched()	-This method decides if the tile has been touched and changes its state accordingly.

Scoreboard Activity:
This will just present a view of the content provider that store the high scores.

Contact Activity:
This will allow the user to select a contact for each player.

Learning reflections:
Designing the above system has been challenging. The nature of a university project is that it is a learning exercise, as such this design my not be a complete due to my current lack of experience with Android. For example, I've never used a content provider before, my above implementation is to the best of my current knowledge.

So far in the project I've learnt a great deal about mobile software. I am now familiar with the separation principles android employs in the form of intents, this was new to me. I've also had to design an application without knowing the hardware limitations, for example the screen size. Another concept I've had to come to grips with is selective programming, ie, thinking about the variable types I'm using to save on resources.

As stated before, I think I will learn a lot more during the implementation of the above design, whilst I've done some prototype coding for this project, I'm yet to implement some of the more complex features of android.