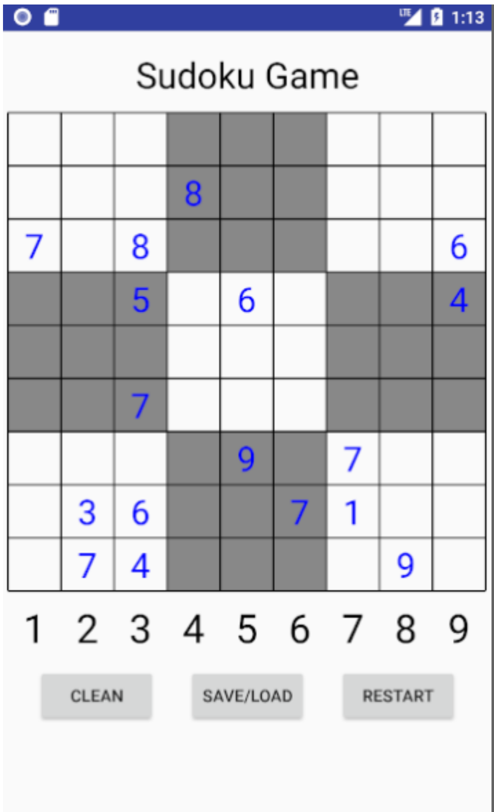
## Project Statement

My project is Sudoku game. With high speed development of technology, mobile devices could provide more functions to user rather than calling and texting. Playing games is a good example of it. Sudoku game is a logic-based, combinatorial number-placement puzzle. Users are required to fill a 9 x 9 grid with digits so that each column, each row, and each of the nine 3 x 3 sub-grids that compose the grid contains all of the digits from 1 to 9. Moving Sudoku game on mobile devices will let users playing this game everywhere without papers and pencils. And this application also can help users to consume times on bus or subway.

## Application Design

This application targets both smartphone and tablet in portrait mode. It supports regular Sudoku game functions, and also Save & Load functions which allows users to mark one or more step as "checkpoints", and later go back to one of the saved checkpoints.

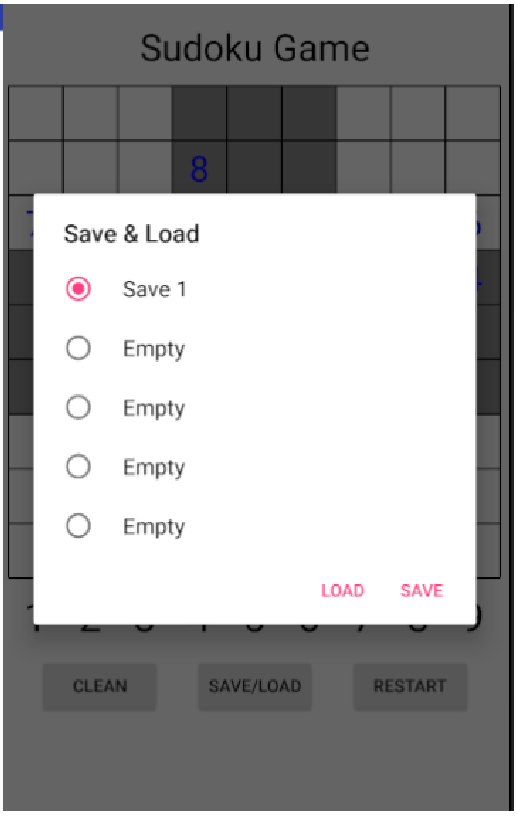
The application only has one activity which will contains all components. The game UI is showing in Figure 1.1. The game board is a custom View component; it will interact with user and update the choice bar when user clicks the cell on board. There are three buttons. Clean button will clear the number on the current cell which user chooses; Save/Load button will open the save & load dialog to let user to save and load game checkpoint, the UI is showing in Figure 1.2; Restart button will clear all user choose number on the board.



Sudoku Game

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| | | | 8 | | | | | |
| 7 | | 8 | | | | | | 6 |
| | | 5 | | 6 | | | | 4 |
| | | | | | | | | |
| | | 7 | | | | | | |
| | | | | 9 | | 7 | | |
| | 3 | 6 | | | 7 | 1 | | |
| | 7 | 4 | | | | | 9 | |

Figure 1.1



Save & Load

● Save 1

○ Empty

○ Empty

○ Empty

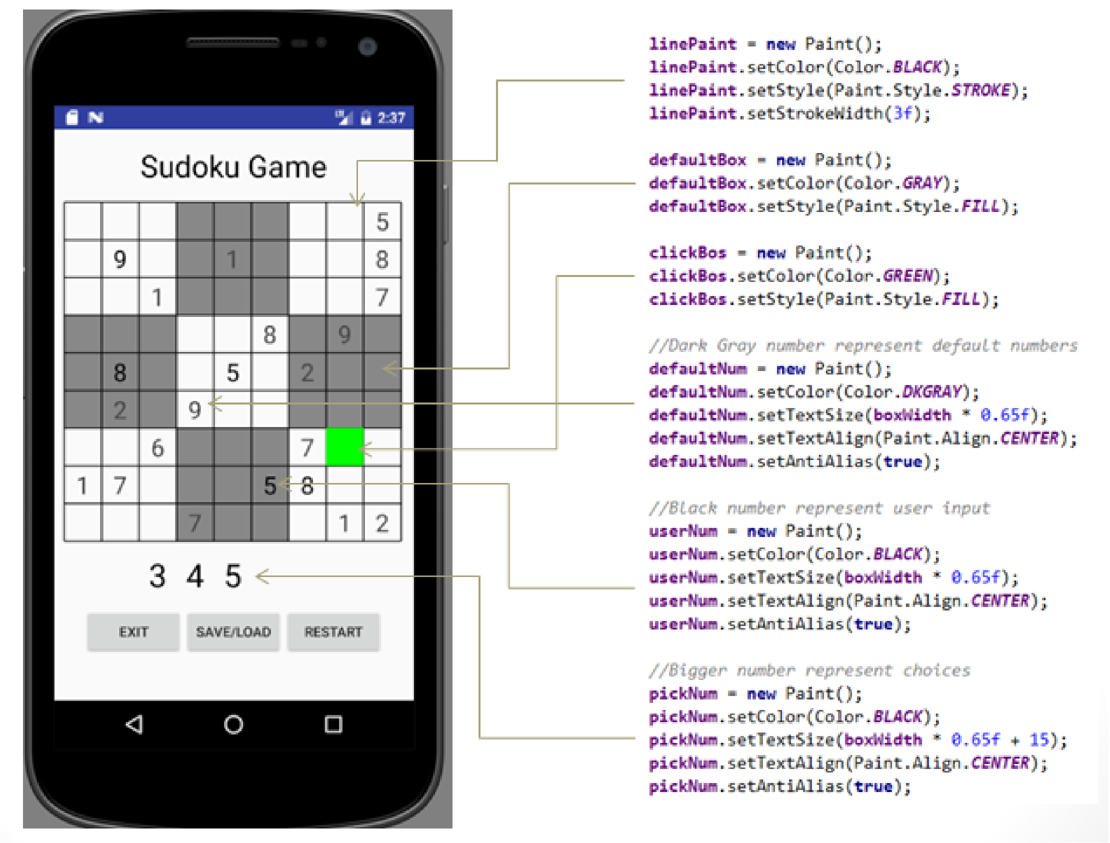
○ Empty

Figure 1.2

## Application implementation and Evaluation

There are two classes in the application, one for UI layer called SudokuView, and other for logic layer called SudokuBoard.

In SudokuView class, the onDraw() method is overridden to draw the game board. This class first gets the Screen width to calculate the width for each cell on the board. By doing this, the game board will fit the whole screen. In this class, there are many Paint object. As show in Figure 2, each Paint object will print one component for the game board, such as the board edge, default number, and user choice number.



|  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  | 5 |
|  | 9 |  |  | 1 |  |  |  | 8 |
|  |  | 1 |  |  |  |  |  | 7 |
|  |  |  |  |  | 8 |  | 9 |  |
|  | 8 |  |  | 5 |  | 2 |  |  |
|  | 2 |  | 9 |  |  |  |  |  |
|  |  | 6 |  |  |  | 7 | 🟩 |  |
| 1 | 7 |  |  |  | 5 | 8 |  |  |
|  |  |  | 7 |  |  |  | 1 | 2 |

```java
linePaint = new Paint();
linePaint.setColor(Color.BLACK);
linePaint.setStyle(Paint.Style.STROKE);
linePaint.setStrokeWidth(3f);

defaultBox = new Paint();
defaultBox.setColor(Color.GRAY);
defaultBox.setStyle(Paint.Style.FILL);

clickBos = new Paint();
clickBos.setColor(Color.GREEN);
clickBos.setStyle(Paint.Style.FILL);

//Dark Gray number represent default numbers
defaultNum = new Paint();
defaultNum.setColor(Color.DKGRAY);
defaultNum.setTextSize(boxWidth * 0.65f);
defaultNum.setTextAlign(Paint.Align.CENTER);
defaultNum.setAntiAlias(true);

//Black number represent user input
userNum = new Paint();
userNum.setColor(Color.BLACK);
userNum.setTextSize(boxWidth * 0.65f);
userNum.setTextAlign(Paint.Align.CENTER);
userNum.setAntiAlias(true);

//Bigger number represent choices
pickNum = new Paint();
pickNum.setColor(Color.BLACK);
pickNum.setTextSize(boxWidth * 0.65f + 15);
pickNum.setTextAlign(Paint.Align.CENTER);
pickNum.setAntiAlias(true);
```

Figure 2

Also, the onTouchEvent method is overridden too. This method is overridden to interact with user. Once user click the box on the board, the View will update the available choices under the board. And when user clicks the choice, the View will update the board data. After each operation, the whole board will be redraw. This method will calculate the current box number and user choice by using the event's x and y position. The class has fixed box size and offset, so it's not hard to calculate.

In SudokuBoard class, it initializes the game board, gives the available choices for current box, and the checks for good map or not. This class is a singleton class. There will only be one game board in the game. It is not necessary to make the board data as the parameter to some method if the board is a singleton class. Every class can access the board data everywhere; no multithread in the program, so it won’t have any issues.

Finally, for the save and load function, AlterDialog is used for it. When users click Save/Load button, a dialog will show out. The application will translate two dimensions integer array board data to one string object, and then save it into file.

## Experiences and Thoughts

This project gives me a clean idea about Android programming. I have learned knowledge about Activity, View, and Dialog. Because of time limitation, I don’t finish the Sudoku generation. Right now, the application only generates a valid game board, but this board may have more than one solutions. Maybe in future, the application will generates a good Sudoku game board.