VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Zehong Chen

Five-in-a-row Game on Android System

Technology and Communication

2017

ACKNOLEDGEMENTS

Many people have helped me in my thesis writing. I would like to express my gratitude to all those people.

Firstly, I would thank my supervisor, Dr. Ghodrat Moghadampour. He spent much time in my thesis working and provide me many suggestions.

Also thanks for the working of my English teacher Päivi Auranen. She gave me a lot of advice in writing academic text.

Finally, I would like to thank my friends for giving me inspiration of the topic. Thanks for your time in testing, I cannot improve the algorithm without your help.

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Bachelor of Engineering

ABSTRACT

Author Zehong Chen
Title Five-in-a-row Game on Android System
Year 2017
Language English
Pages 67
Name of Supervisor: Moghadampour Ghodrat

The topic of this thesis was programming a five-in-a-row game on an Android system. As a traditional game, five-in-a-row has many faithful fans, the most popular five-in-a-row game on Google Play has about 5 million downloads.

Five-in-a-row has simple rules, it is easy to play but hard to master. Designing a satisfactory algorithm for the five-in-a-row computer player was the main aim of the study.

This game has a single player mode as the main function. In this mode the players can play against a computer player, which is supported by an algorithm. In addition, the game also allows players to save their record in their local account, and players can set configurations for other functions.

The application was developed for an Android system, which is using Java as the programming language. The UI and algorithm both used Java. Some XML flies were also used to define the application layout and properties. For storing the data, SQLite was used save the record of players. This thesis describes the process and the theory which supports the algorithm of a computer player, and the structure of the application. The details such as the calculation of weight and the analysis of the situation were also covered in the thesis.

Keywords Game, Android, Java Programming, Five-in-row game

Contents

1. Introduction 1
1.1 Background 1
1.2 Motivations 2
1.3 Objectives and Topic description 3
2. Technologies 5
2.1 Android 5
2.1.1 Activities 5
2.1.2 Services 7
2.1.3 SurfaceView 8
2.1.4 Lists 8
2.1.5 SQLite 9
2.2 Java 10
2.3 XML 11
2.4 System relative 11
3. Application description 13
3.1 Functions Requirement 13
3.2 Functional Specification 14
3.3 Main Functions 16
3.4 Classes of the Application 16
3.5 Sequence Diagram 25
4. Database and GUI design 30
4.1 Design of the database 30
4.2 Design of different parts of GUI 31
5. Analysis 33
5.1 Analysis of game rules 33
5.2 Analysis of program strategy 41
6. Implementation 45
6.1 Implementation of program strategy 45
6.2 Implementation of UI and other function 46
6.2.1 Layout of menus 47
6.2.2 Functions of Buttons 48
6.2.3 Drawing the board 50
6.2.4 Background Music 51
6.2.5 User list 53
6.2.6 Database 55
7. Testing 57
7.1 Algorithm test 57

7.2 Functions and test....58
8. Summary 60
9. Conclusion 61
10. References 62

1. Introduction

The target of this project is to program a five-in-a-row game on an android system. The introduction will describe the basic background and objectives of this project.

1.1 Background

Five-in-a-row needs two players, and the players choose their color (black or white) before the game starts. Normally the player who takes the black stone will play first, and the players will alternate in placing their stone on an empty space. If a player gets a continuous row of five stones in his color horizontally, vertically, or diagonally, he wins the game /1/.

Five-in-a-row has two styles, the “Standard” style and the “Free” style. The standard style asks the players to get exactly five continuous stones in the same color. Free style requires five or more continuous stones, but they do not need to be exactly five stones.

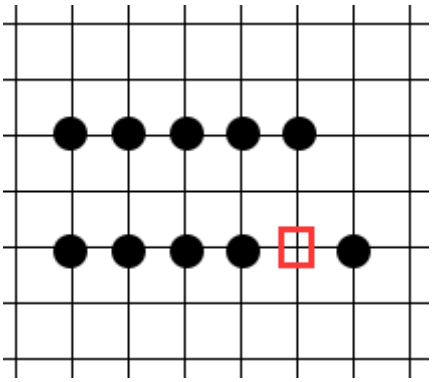

Figure 1. The marked place is not allowed in the standard style

In addition, the rules of five-in-a-row have many branches. For example, “Swap2” rule, it asks the first player to place three stones (two in black and one in white) before the game starts. The second player also has the right to choose which color he will take, or he can change the shape of stones and give the right to choose the color to player 1.

There are other optional rules, some rules ban “three and three” or “four and four” for the first player. This is because the first player will have the advantage (without Swap2 rule), by banning the “three and three” or “four and four”, which will make the game fairer.

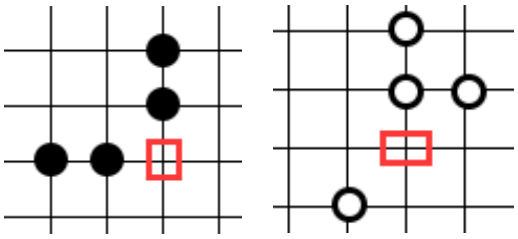

Figure 2. The marked places are not allowed when banning the “three and three”

1.2 Motivations

The personal motivation of this project is that I want to work in the game design field in the future. At the end of the bachelor’s degree, an individual android game as a topic of final thesis offers a good challenge to convert previous studies into a complete game. In addition, in future interviews at game companies, individual game design experience can also be an advantage.

The reason that Five-in-a-row game is chosen is the algorithm programming of five-in-a-row is a challenge. To make players have fun, the algorithm should not be so simple. The computer should be difficult to defeat, so it can give players a sense of accomplishment after they win the game. Which means the algorithm should be smart enough to against human player. This is a challenge to the algorithm designer.

1.3 Objectives and Topic description

As mentioned above. The game should have an algorithm to support computer against the players. But it is difficult to define what will be the appropriate level for this algorithm.

To find appropriate difficulty, in the test part, this algorithm will against different five-in-a-row game algorithms which are downloaded from the Google Play. The more the algorithm wins, the better it is. In addition, the game should also have some other functions to make the players enjoy it. The details of these functions will be introduced in **Application description** chapter.

In brief, the application should have these functions:

- **Single player** mode
- **Player vs Player** mode
- **Account system**
- **Setting** for configurations

Under **Single player** mode, the player should be able to play versus the

computer player and the player must be able to choose the level of the computer player. There are three levels to choose from: basic, algorithm and advanced.

The rule of the game is free style without any other optional rules. This is the most popular version of five-in-a-row, without any additional rule the game can be friendlier to beginners.

After every game, the result will be saved and updated to the SQLite. The application will give a rank to the player which is based on the record of the player. The record includes the total games that the player has played and how many times the player was won the game.

Player vs Player mode allows two players to play free style five-in-a-row on the same device. Two players take turns to move their stone. The black stone will move first as in the most five-in-a-row games. The game played under this mode will not counted into the record.

The **account system** saves player information on their accounts. When the game is started on a device for the first time, meaning no account or no database is detected, a default account will be generated, and the application will ask the player to input the name of the account.

Later, if the user wants to generate a new account, they can find this option in the **Setting** menu. In the **Setting** menu, the user can find options about switching an account and create a new account. Apart from the account system, the user can also turn on or turn off the background music in the setting menu.

2. Technologies

The technologies chapter describes most of the technologies which are used in the applications.

2.1 Android

Android is one of the most popular platforms in the world. It is an open-source mobile operating system based on Linux.

Android uses Java as the programming language. The most basic four components of an Android application are: Activities, Services, Broadcast Receiver, and Content provider.

In the application, only activities and services are used, so the rest of components will not be introduced in the thesis.

2.1.1 Activities

As mentioned above, Activity is one the basic components. Activity is the root of all the procedures. All the procedures are running in the Activities. Activity can be regarded as the most frequently encountered by developers /2/.

Activity can be considered to be a page of the mobile phone screen. Like any website page, pictures and buttons can be added to the activity and the function of those buttons can be defined by developers.

The layout of an activity is always defined by an XML file. In this XML file, the developer will define the path of resources and give them an ID. In this way, in the activity classes, the developer can call the ID of a button or a picture to find the resources.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.czh.eglake.thesis.activities.MainActivity"
    android:background="@drawable/bg_small">
```

Code Snippet 1. The XML is used to define the layout in Android project

An activity has its life cycle. For example, when an activity is generated or destroyed, some specific method will be called.

For example, *onCreate()* method is one of the most common methods in activity classes. When an activity is generated, this method will be called. This method is always used to set the layout to activity or load some data which is displayed by this activity.

In activities, listener of every button will be set. When the button is pressed, the activity will run the code which is written in the listener.

```
main_vscpu = (ImageButton) findViewById(R.id.imageButton);
main_vscpu.setOnClickListener(new View.OnClickListener() {
      public void onClick(View v) {

            /**
              If pressed, intent to next activity
              */

            Intent intent = new Intent(MainActivity.this, LevelActivity.class);
            startActivity(intent);
      }
});
```

Code Snippet 2. Find a button by ID, and set a listener for the button.

Activities can intent to other activities by creating an intent object. When an intent object wants to switch to the new activity, the old activity will be stopped automatically. If the developer wants to save more resources of the system, the code can finish this activity when it is stopped or paused. However, if this activity needs to be used later, it can be kept in memory.

When the “exit” button on the device is pressed, the newest activity will be stopped, and the second last activity will start running.

2.1.2 Services

Services are different from Activities, they do not have a page or any specific object to display. Most of the services are those functions which are running in the backend.

For example, most of the music players are services. Though they have a UI which are activities, but when user leaves the interface or jumps to another APP, the music should keep playing. Even if the user turns off the screen, the music will keep playing. To do this, services are needed.

In this application, the services are used to play background music. At the beginning of the application, when the first activity of this application is generated, the music will be played. This means at the onCreate() method of the activity, the services is started /3/.

2.1.3 SurfaceView

SurfaceView is a kind of class in Android. It can be used to deal with the change of graphics. In this application, the SurfaceView is used to draw the game board and stones.

Updating the screen in the main thread of the UI which may cause problems. Therefore, the developer needs to use SurfaceView to renew the UI. The SurfaceView class is a new thread that can be redrawn in a separate thread, and it will not influence the main thread, which means it can greatly increase the system stability.

2.1.4 Lists

The list is a kind of Activity (full name of the class is ListActivity). It is always used to display information in a list. If the developer need to display the name list or any kind of list, the ListActivity will help.

ListActivity class also has an editable layout which is defined by an XML file. In the layout XML, the XML can change the font size, the inserting of pictures, and other components.

The list also can be clickable. The ListActivity has a method which is called onListItemClick(), By using this method the developer can make the items in the list clickable, and no more button need to be added /4/.

Figure 3. A list activity.

2.1.5 SQLite

The SQLite is a database in Android. The developer does not need to add extra libraries to use SQLite in the code. SQLite is the default database of Android.

As a database, SQLite can save data in tables. In this application, SQLite is also used to record the games that were played on the device.

Because of the application may be downloaded and installed on different devices, the application will search for existing the local database to know if this is the first time itself is ran. If it is, a new database and new table will be generated automatically.

2.2 Java

Java is one of the most famous object-oriented programming languages. It is powerful and easy to use.

The Java environment itself is portable to new hardware platforms and operating systems. This is why applications which are written in Java can support different types of devices.

Android system have inherited the advantages of Java. Android is supported by tablets, smartphones, TVs, and a lot of other devices.

Different from C or C++. Java does not have functions, instead Java has methods. Methods describe the behavior of classes (objects). Every class can have many methods to support its actions /5/.

2.3 XML

The full name of XML is *Extensible Markup Language /6/,* The Extensible Markup Language allows users to structure documents and data so that they can be exchanged between departments, customers, and suppliers for dynamic content generation, enterprise integration, and application development. XML also allows users to search more accurately and more conveniently. In this application, XML is used to define the layout of activities, the path of resources, and other configurations of the application.

2.4 System relative

All the applications have requirements for an operation system. The Android application also has this kind of requirements. The newest Android system is API 25, Nougat. The lowest system for this application is API 15, IceCreamSandwich.

The Android system can be installed on many kinds of devices, such as smartphones, tablets, and TVs. However this application is designed for smartphones and it may not run on other kinds of devices

Table 1. The Lowest supported version *Android "Ice Cream Sandwich"* is Unveiled in October 2011. /7/

Version	**Name**
Android 1.1	Petit Four
Android 1.5	Cupcake
Android 1.6	Donut
Android 2.0/2.1	Éclair
Android 2.2	Froyo
Android 2.3	Gingerbread
Android 3.0/3.1/3.2	Honeycomb
Android 4.0	Ice Cream Sandwich
Android 4.1/4.2/4.3	Jelly Bean
Android 4.4	KitKat
Android 5.0/5.1	Lollipop
Android 6.0	Marshmallow
Android 7.0	Nougat

3. Application description

The application description will describe the functions and objects of the application in detail. Also, some diagrams of the application will be shown in this chapter.

3.1 Functions Requirement

Functions requirements list all the necessary objects of the project. It helps to organize development processes and make a development plan. QFD describes the requirements of functions and it distinguishes the importance level of different functions.

Table 2. Functions requirements

The application must have:
● Users can play a five-in-a-row game with AI ● Users can play a five-in-a-row game with friends on the same device. ● Users can save and update their game records in database ● Users can save and update their configuration in database ● Users can create accounts and switch account ● The application can play the background music
The application should have:
● The algorithm of application should hard enough to bring player fun in the game. ● The application should run smoothly without crashes. ● The application should accurately locate the click event from user

The application nice to have:
● The algorithm of application should hard enough to bring players a challenge. ● The application should run smoothly without lag.

3.2 Functional Specification

The specification table lists all the functions of the application. It mentions what kind of functions does the application have and what kinds of exceptions may happen.

Table 3. Functional Specification

No.	Function	User Action	Expected result	Exception
1	Start application	Click the APP on an Android device	The main menu is displayed and the database is initialed.	The application crashes, or the database not working.
2	Single mode	Click the “Single mode” button	The level selection menu is displayed	The application crashes. Selection menu is not displayed correctly
3	Single mode Game	Select the level	The game board is displayed	The game board is not displayed correctly

No.	Function	User Action	Expected result	Exception
4	Game running	Click on game board	The AI plays by turns with you and the move of AI is logical.	The move of AI is not logical.
5	PVP mode	Click the ”Player vs Plyer” button in the main menu	The game board is displayed and players can play game by turns	The game board is not displayed correctly
6	Setting	Click the ”Setting ” button in the main menu	The setting menu is displayed.	Setting menu is not displayed correctly
7	BGM	Click the ”ON/OF F” button in the setting menu	The background music is changed.	The button is not working correctly.
8	Account system	Click the ”Change User” button in the setting menu	The user list is displayed.	The list is not displayed correctly.
9	Setting save	Exit game and run it again.	The setting which is changed last time should be kept.	The change is not saved.
10	Exit	Click the ”Exit” button in the main menu	The application exit without crashes and background music is turned off automatically	The applicatio n crashes or backgroun d music is keeping playing

3.3 Main Functions

The main functions of this application are quite simple and they are be sorted into the Figure 4：

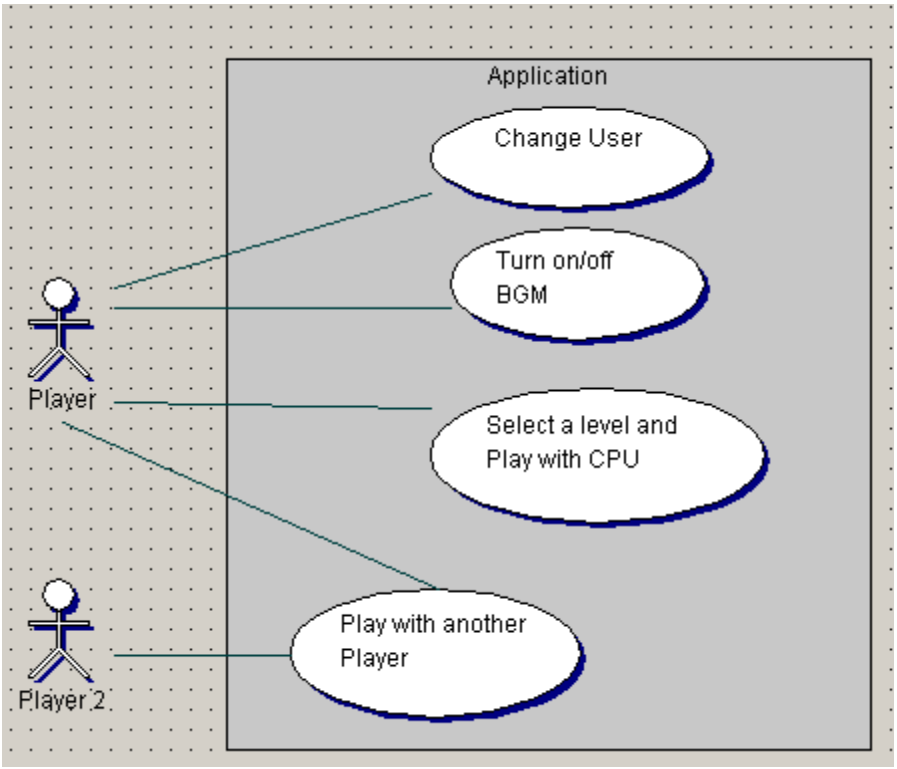

Figure 4. Main functions

3.4 Classes of the Application

The classes of the application are sorted into packages. All the other packages are related to Activity package.

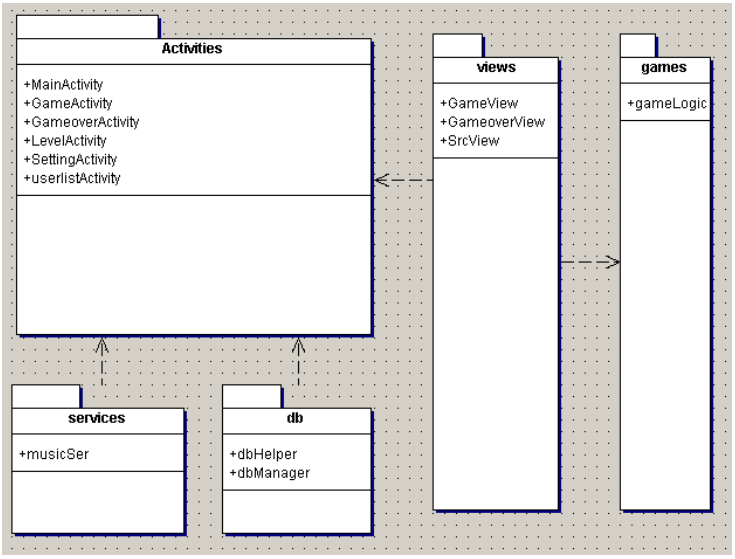

Figure 5. Packages and Classes inside

The activities are individual and every activity does not have a relation to other activities. Views are extended from SrcView and then they need to be set to activity to display themselves. GameLogic class only has a relation to Views, which is used to display the game boards. More detail about these classes are presented in Figure 6:

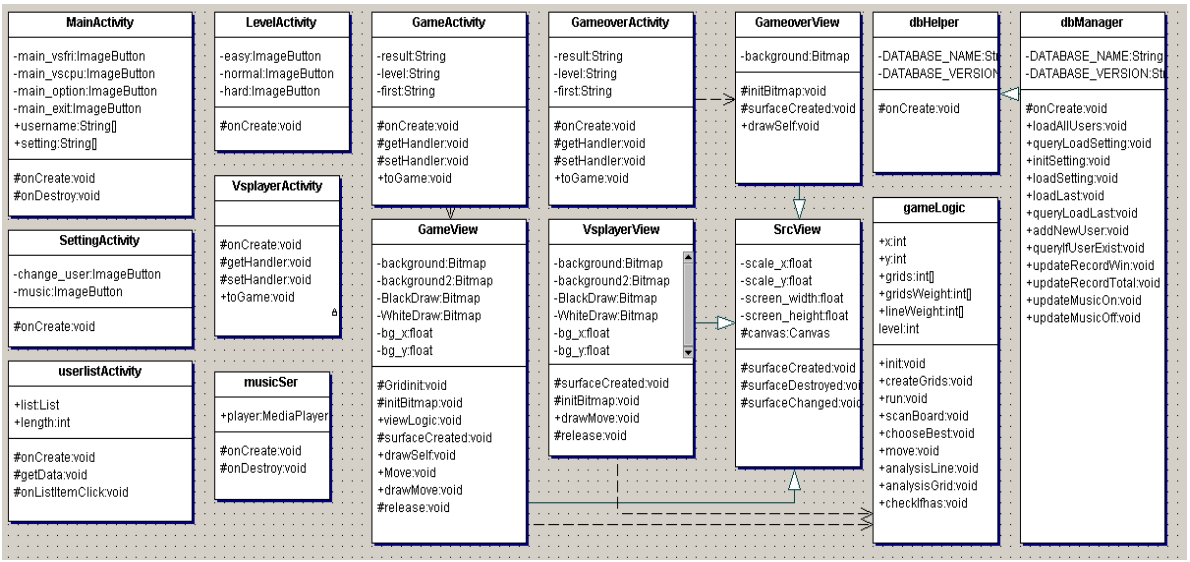

Figure 7. Classes of the application

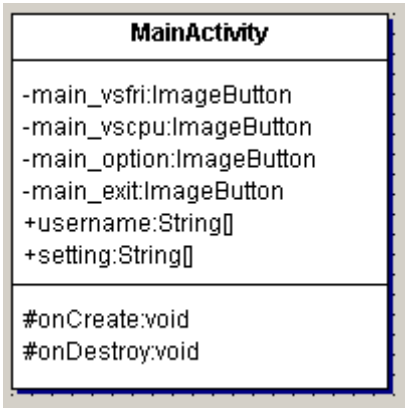

Figure 8. MainActivity Class

The Figures 7 is about the MainActivity class. It is used to display the main menu. When the user runs the application, the main menu will be displayed. It will also check if there are any user exists, if no user or no database is found, a new account will be generated.

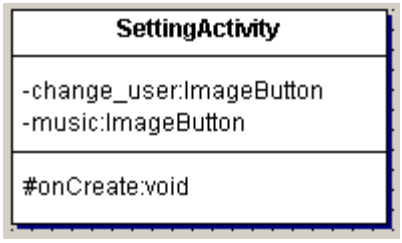

Figure 9. SettingActivity Class

The SettingActivity class is an activity which is used to change the settings of the game. There are two buttons on the menu: BGM ON/OFF and Change User. The BGM button is used to control the background music and the Change User button will create an intent object which can jump to the user list.

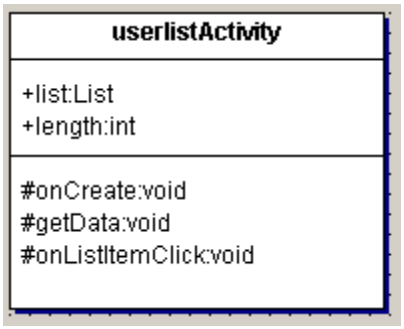

Figure 10. userlistActivity Class

Figure 9 is about the userlistActivity class. This class will display all the users and their records. The item in the list is clickable, a user can choose the account by clicking the name.

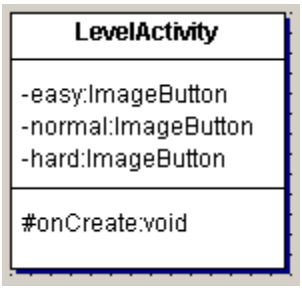

Figure 11. LevelActivity Class

The LevelActivity can be called from the MainActivity. It is used to provide a UI for level choosing. There are three level buttons inside: basic, algorithm and advanced. The level which is chosen will be sent to next activity.

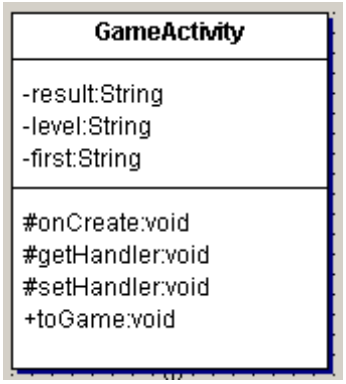

Figure 12. GameActivity Class

After the player has chosen the level, the GameActivity class will be called. GameActivity will set a view as its content. At the same time, a game object will be created. After a game is finished, it will generate an intent to GameoverActivity.

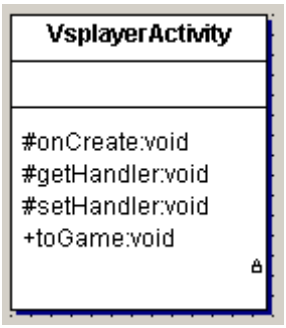

Figure 13. VsplayerActivity Class

This activity is almost the same with GameActivity, but it does not have the handler for the levels.

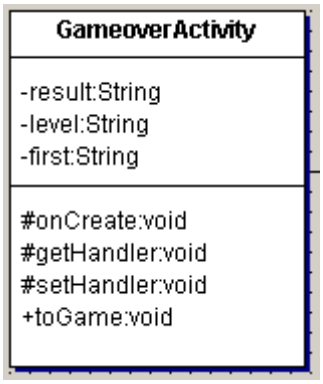

Figure 14. GameoverActivity Class

The GameoverActivity class will be called when a game is finished. It will display the GameoverView, which is used to display the winner of the game. The result of the game will be saved in database.

Figure 15.musicSer Class

The musicSer class controls the background music service. It will start playing music when it is created. The music will keep looping until the user stops it in the setting menu or the application close.

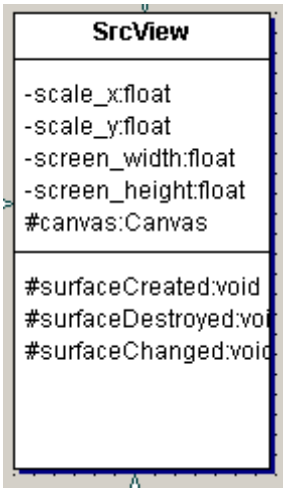

Figure 16. SrcView Class

Srcview class provides the basic methods and values for GameView and VsplayerView. It include values that will be used to calculate the size of the device's screen.

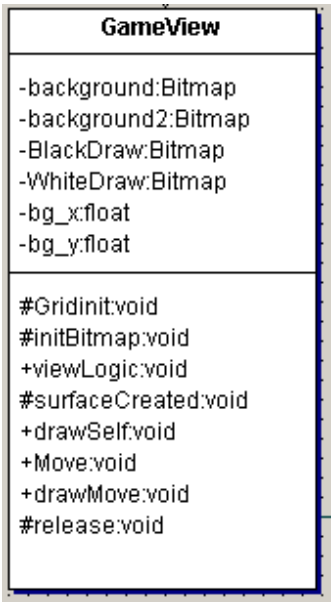

Figure 17. GameView Class

GameView class extends SrcView class. The game board is drawn on this view. It reads the picture resources and converts them into bitmaps and display

them on the screen. It also has the listener of the clicking event, the location of clicking event will be converted into a number of the grid and then sent to the gameLogic object.

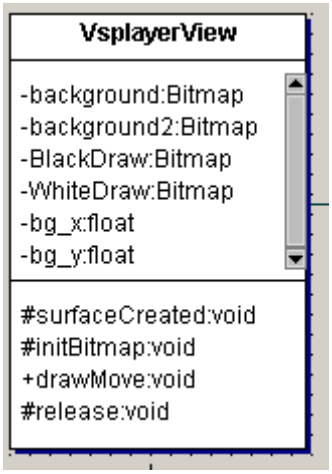

Figure 18. VsplayerView Class

This view is almost the same with GameView class, but it does not ask the gameLogic object for solutions. It will only ask the gameLogic object if there is a winner. At the end of the game, the VsplayerView class will generate an intent to GameoverActivity class.

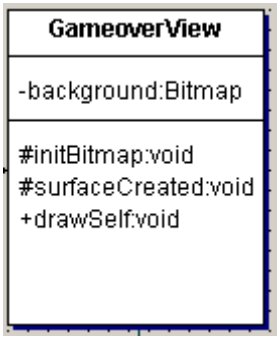

Figure 19. GameoverView Class

GameoverView class is displayed on GameoverActivity class, it will display the

winner of the game. It ALSO has a click listener. If a clicking event happen, it will generate an intent to a new game activity and start a new game.

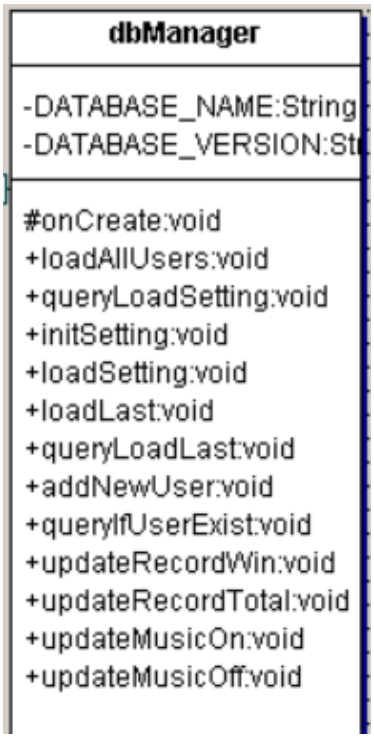

Figure 20.**DBManager** Class

The DBmanager class has many methods for sending a request to the database. The other class can create a manager object and call the methods inside.

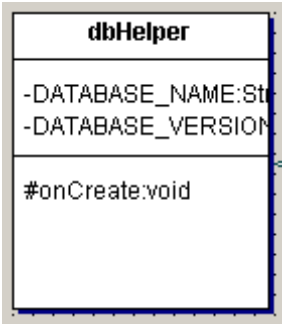

Figure 21.**DBHelper** Class

The DBHepler class checks the database at the beginning of a game. If the database or the table does not exist, it will create an empty one.

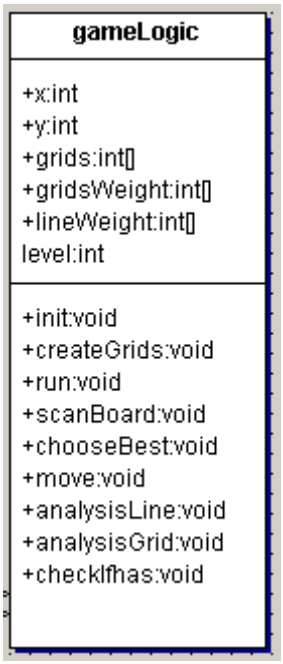

Figure 22.**GameLogic** Class

The GameLogic class controls the actions of the CPU player, it will read the girds from Views and send solutions to Views. It will take in an integer as the AI level when it is created.

3.5 Sequence Diagram

Sequence diagrams display the process of functions. The functions mentioned in “Main functions” chapter are presented here.

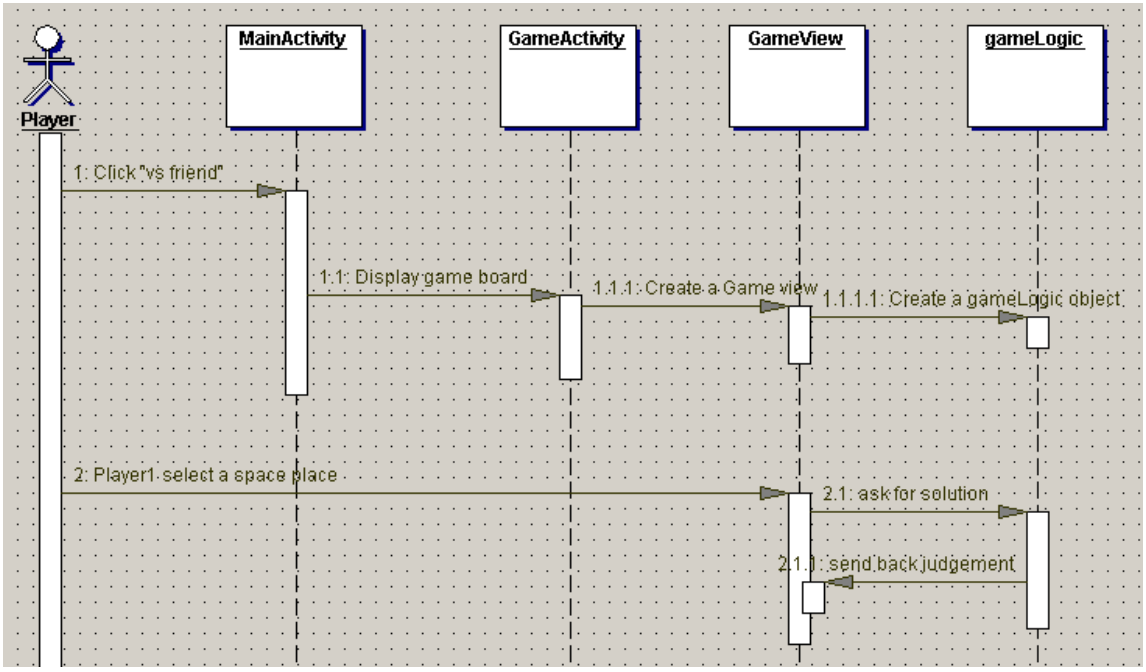

Figure 23. The process when a player against AI

All the events will be listened in GameView class and the location of the event will be translated into the grid number of the game board. This number will be sent to gameLogic object and the object will give out a solution and send it back to the front view. After the view receives the solution, it will update the canvas to match the latest change. This process will keep looping until the game is finished.

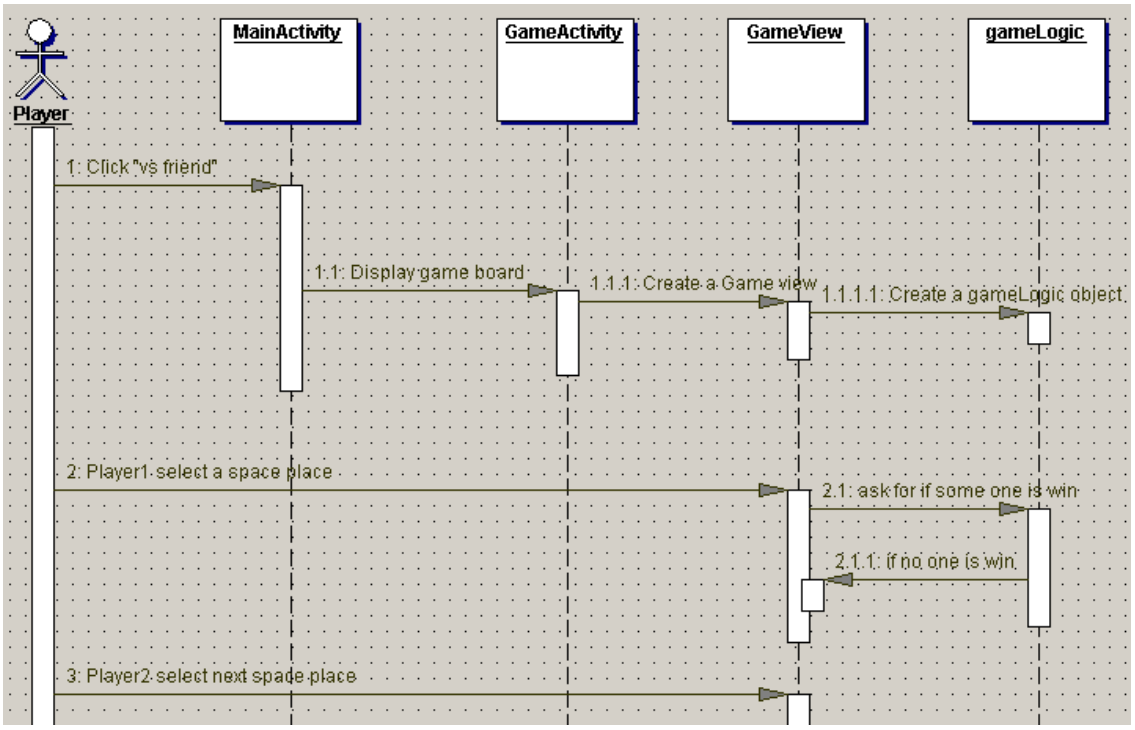

Figure 24. The process when a player against a player

Vs friend function does not have many differences with vs AI. The only part different is the gameLogic object, which will not give a solution for the game. It is only used to judge if there is a player who wins the game. Every time the stone moves, the logic class will scan the board again to make sure if the game has ended.

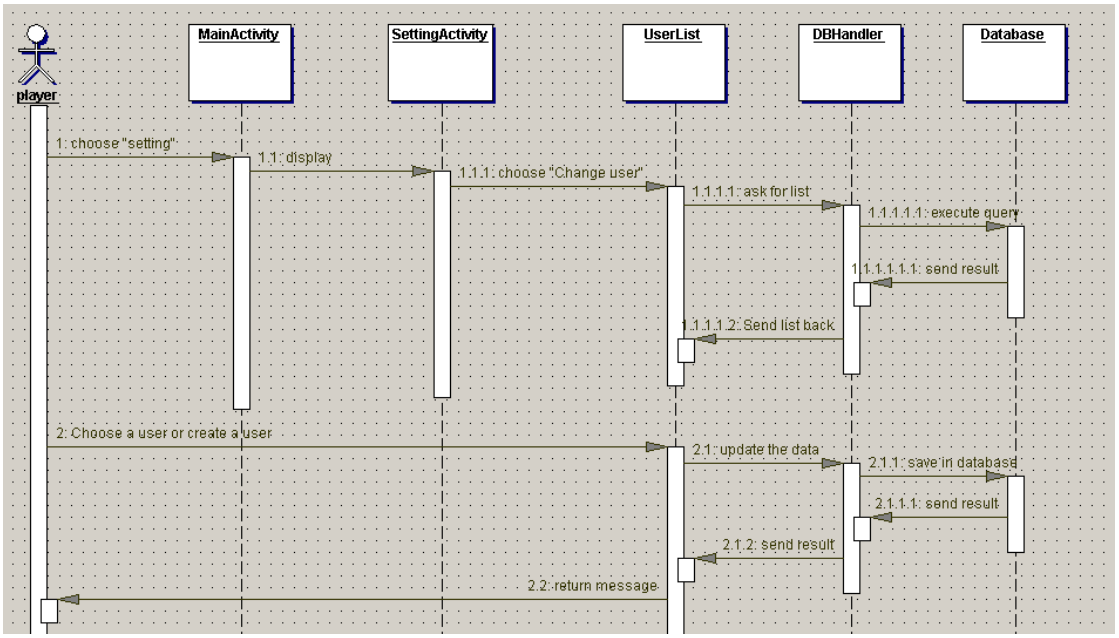

Figure 25. The process of changing user

Players can change the options of the game. Players only need to click the "Setting" button located in the main menu to enter the setting menu.

The change user function asks for some user data from the database and then display it. The user list is clickable. When a user name is clicked, it is chosen.

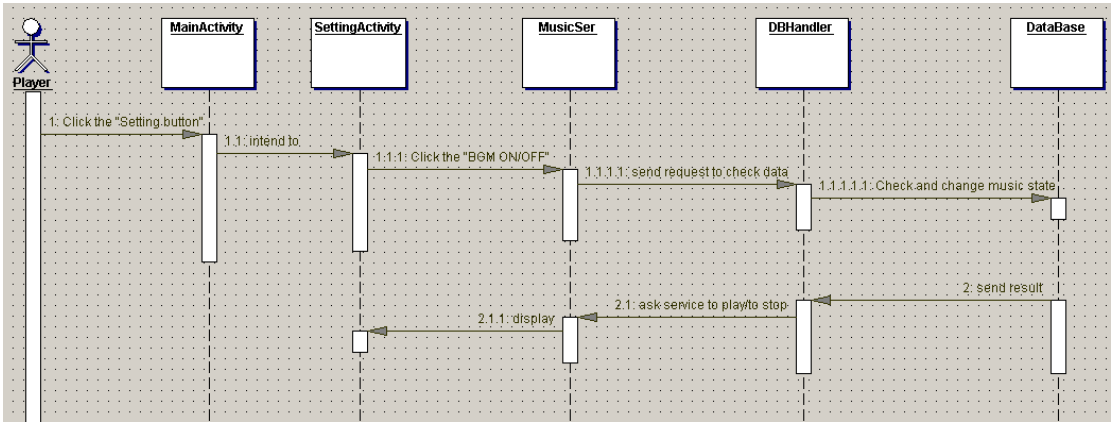

Figure 26. The process of turning on/off BGM

To turn on or turn off the background music, a user needs to enter the setting menu first. A users can click the “Setting” button on the main menu to enter the setting menu. Then a user can press the “On/Off” button. After the button is clicked, a request will be sent to a DBManager and it will change the data in the database, then send a message to the screen to tell the user the updating is done.

4. Database and GUI design

A database is a warehouse that organizes, stores and manages data according to the data structure. GUI provides a more user-friendly interface for operations. A database and GUI are necessary for most applications. This chapter shows how the database and GUI of the application generated.

4.1 Design of the database

The database of this application is used to store user configuration and the record of games which are played on this device.

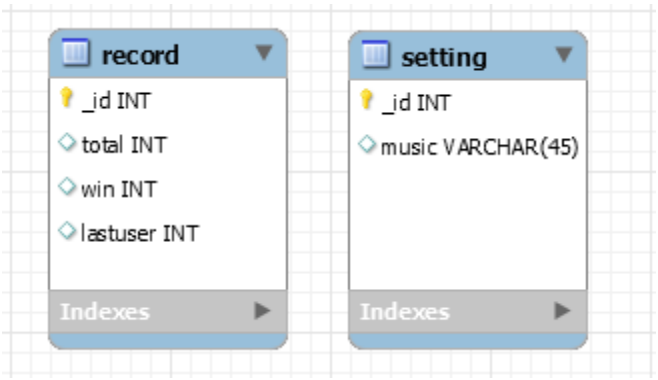

Figure 27. The structure of the database

The two tables in Figure 26 are individual, so there is no connection between these tables. The first table is named “setting”. As the name, this table included the configurations of the game. If a device is running the application for the first time the application will generate this table and give default values for it.

The table only has one row and it cannot be increased. In this row, different columns mean different options, and the choice of the user is stored by an INT value. For example, the “music” column means the “off” or “on” of the background music. The value “0” means “off” and value “1” means “on”.

Another table, which is named “record”, is used to save account information. The same applies to “record” table, a default account will be created in the first time running. Later if user creates more accounts, a new account will occupy a new row in a table.

In the table the columns “total” and “win” record how many games are played and how many games are won under this account. The last column, “lastuser” is used to locate the current user. If the user is the current user, the “lastuser” will be set to 1. Otherwise, it will be 0.

4.2 Design of different parts of GUI

Graphical User Interface, referred to as GUI, refers to the use of graphical display of a computer operating user interface. A suitable image and design will bring a more user-friendly interface.

Similarly to most Android applications, this application can be operated by clicking the device screen. In the case, the button has two sets of style to display when the buttons are released or pressed.

Figure 28. The "single player" in the first picture is released and in the second pictures it is pressed

Most of the menus look like Figure 7, but in the game, a board to play game is needed.

5. Analysis

In this chapter, an analysis about game logic will be described. The analysis is based on the game rules and also on other judgement in extreme cases. Different players have different way to play, but in those extreme cases, they having almost the same choices. By analyzing those choices, the AI can play the game more skillfully.

5.1 Analysis of game rules

Before the analysis, in order to facilitate the presentation of the combination on the board. Table 4 shows some combinations in abbreviations. Those abbreviations will be used later to introduce and analyze the judgement in games. By the table, the corresponding combinations can be found.

Table 4. Combinations

Combination	**Description**	**In short**
X	"X" means black stone	--
O	"O" means white stone	--
XX or OO	Two consecutive stones without block	L2 (Means: "Live 2", "Live" means no block on two sides.)
OXX or XOO or OOX or XXO	Two consecutive stones with block	D2 (Means: "Dead 2", "Dead" means have one or two blocks on two sides.)
XXX or OOO	Three consecutive stones	L3

Combination	Description	In short
OXXX or XOOO or OOOX or XXXO	Three consecutive stones with block	D3
XXXX or OOOO	Four consecutive stones	L4
OXXXX or XOOOO or OOOOX or XXXXO	Four consecutive stones with block	D4
XXXXX or OOOOO	Five-in-a-row	--

When players look into the game rules, they will find one thing is never changing: if you already have four continuous stones in a row without a block at the ends and it is your turn, you always can win the game.

All five continuous stones come from four continuous stones, in the same way, all four continuous stones come from three continuous stones. In sum, if you want to win, in most situations, you should increase the length of your continuous stones.

So, the rule one is found: **Increasing your continuous stones to as many as possible.**

Separately, because this application does not ban the “three and three” for the player who starts first, the player should also find the place which can bring a “three and three” as it is better than the place which can bring you one three-in-a-row. Figure 28 demonstrates that the blue space is better than the red one.

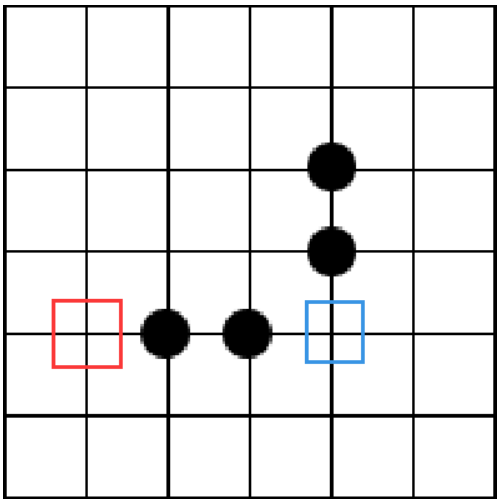

Figure 29. The blue mark is a better place than red mark. Because there are more attack chances around blue mark. Blue brings 2 L3, red only brings one L3.

In another situation, the player also needs to consider is there is any stone from another player blocking the continuous. The Figure 29 shows the difference.

Both blue and red place and bring a continuous 3 stone query to black stones, but the blue place is better. This is because the blue place brings an L3 to black, but the red place only brings a D3.

L (live) query is always better than a D (Dead) one, because another stone needs two steps to fully block ("Fully" means both two sides of a continuous query are blocked) the L query.

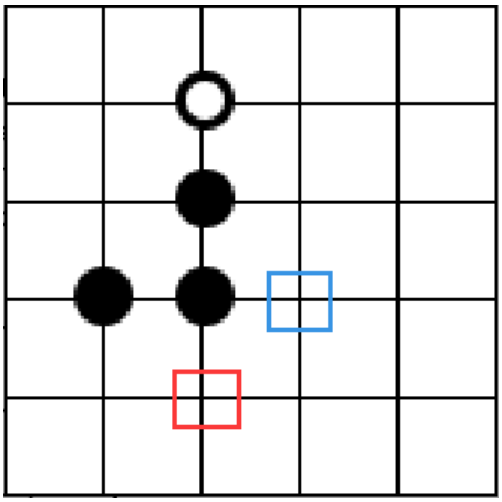

Figure 30. The blue mark is a better place than red mark, because the continuous stone in vertical direction are blocked. But it horizontal direction the continuous stones is free (without block).

In each grid, there are two values to estimate whether this place is worth moving to. By analysing these two values, the CPU/Player will decide the best location of the move.

For example, in Figure 1, the area in red bound does not have any attack value to the white stone in this row. The reason is that the number of places between two black stones is 4. This means there is no way to have five white stones in a row between these two black stones.

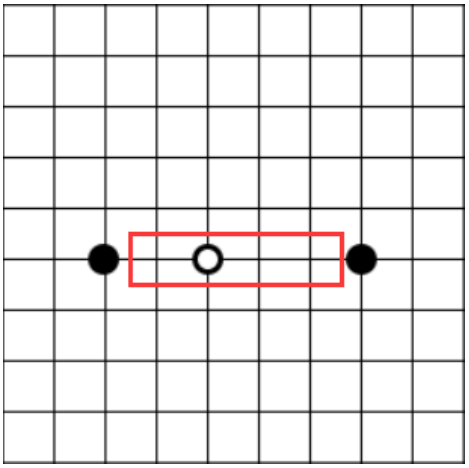

Figure 31. In the red bound region, the maximum possible length of white continuous stones is 4. This means the white player has not chance to win in the red bound region. Therefore, it makes no sense to place a white stone there.

On the other hand, the area in red bound does not have any defense value to the black stones. The reason is the same: There is no way to have five white stone in a row between these two black stones, so the black stones should not waste turns to blocking white stones in the red bound region.

It can be found that the attack value of one player equals the defense value to another player. Therefore, the critical place is the same to both sides.

Figures 31 will demonstrate two different cases: If the next turn is the player with the white stones, the player will find the red bound region is quite a critical place. It can block the black and let the white make a D4. However, the red bound region is not the best choice. A black stone can move in the blue bound region to make the previous attack useless.

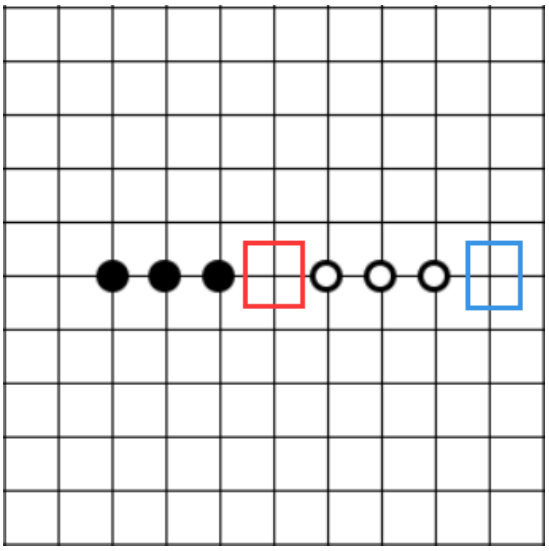

Figure 32. The red place brings a D4 to white and blocks an L3 of black, The blue place brings an L4 to white. In most cases, an L4 means win.

The best choice for the player with the white stone is the blue bound region. By moving here, the player with white can make an L4, in this way, the black player needs two steps to block this L4, but the white one only needs one more step to win.

This may tell the player that the values of attack and the value of defense will decrease each other, but in other cases, the values of attack and the value of defense may increase each other.

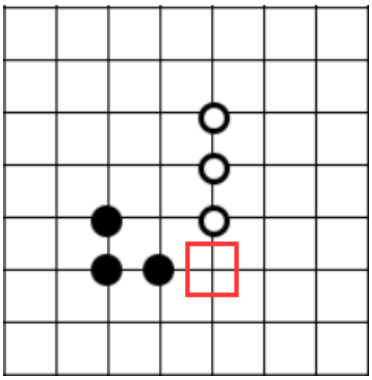

Figure 33. The red place can attack and defend in same time.

In other words, if stones in different colors are placed in one line (one direction), they will decrease the value of each other. But if they are placed in different directions, the crossing point (if the place is empty) will be more meaningful.

In this case, every grid can be separating analyzed in four different directions. The stones in each direction will not influence the value of another direction. This means that the values of these four directions are individual. After the values of these four directions are counted, the value of the central point can be calculated by the sum of these directions.

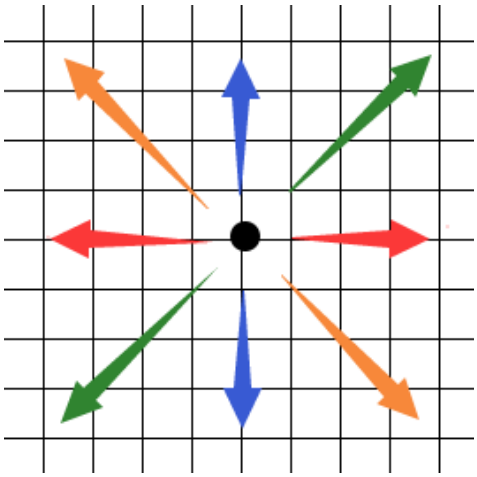

Figure 34. The player only needs to care about the four directions. Other grids have no influences to the central place.

There is an example in Figure 34:

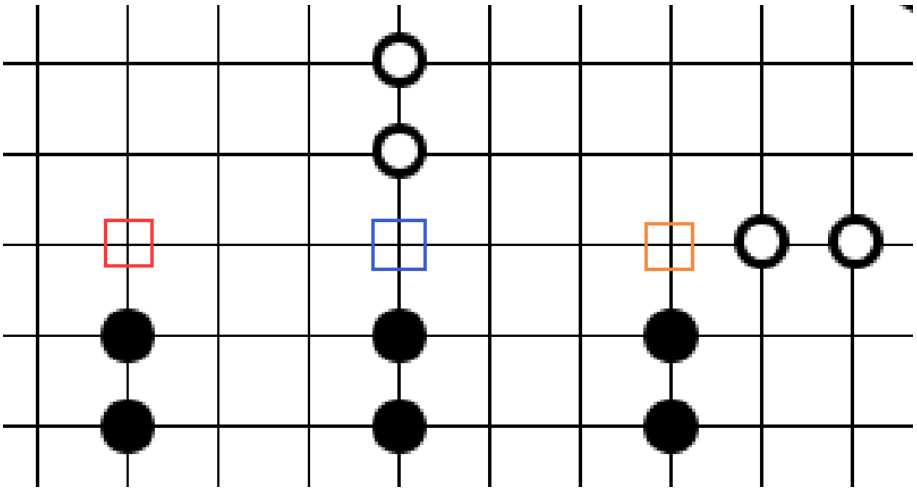

Figure 35. Chooses

You are the black player. If there are there chooses on the board: **RED, BULE, ORANGE.** Which one will be the best? These three case can be considered

separately.

Red: The red place will make an L3 for the black.
Blue: The blue place will make a D3 for the black, and block the white L2.
Orange: The orange place will make an L3 for the black, and block the white L2

If the value of making an L2 is A, the value of making an L3 is B, the value of making a D3 is C, the value of blocking an L2 of the white is D. We can find:

Value of Red: A
Value of Blue: C+D
Value of Orange: B+D

If the player is a beginner, he/she cannot know if a D3 is better than an L2, but one thing is sure, an L3 is better than an L2 and an L3 is better than a D3.

This means: B > A and B > C. Therefore, Orange > Blue and Orange > Red. Orange is the best choose.

5.2 Analysis of program strategy

To apply the game rules into strategy, the first thing to do is make AI able to "see" the situation of the board. In detail, the action "choose a place for the next step" can be understood in this way: "Find the best choice of all available places". In other words, the first step is marking the empty places on board, because only empty places are available. Then, choose the best place.

The values of places are already discussed previously. The problem is how this can be applied to the program strategy. In the previous example, the example only gives an approximate value, but in order to make AI understand the meaning of judgement, the specific weight (or value) must be given to each situation.

In fact, the judgement of the situation can distinguish the player's game style, for those offensive players, they will think highly of the value of attacks. However, they will choose to defend in some case, no matter how they like the attacks.

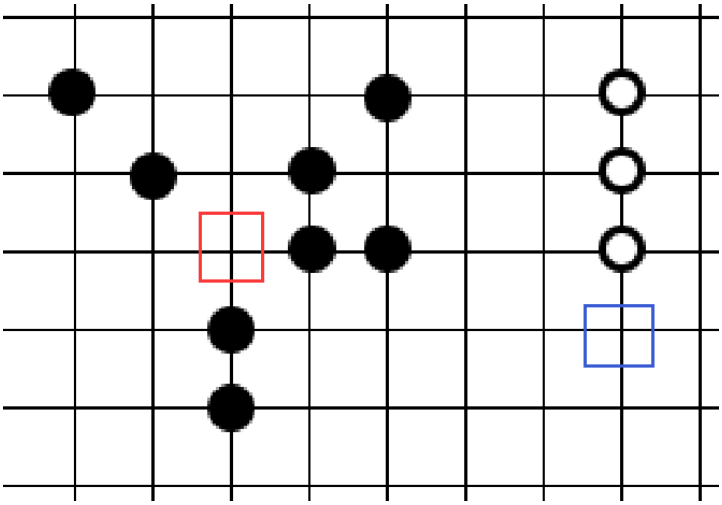

Figure 36. Even the red place can bring 4 L3 to black, but the blue place is still the better choice.

In this example, red is a good place to attack, but black has to move the next stone to the blue place to prevent failure. Even if the black has more chances to attack, the white is faster.

By giving these examples, we can find: **Blocking 1 L3 > Making 4 L3.** If the value of making an L2 to an L3 is **A**, the value of blocking an enemy L3 is **4A+1** at least.

Another case which is shown below demonstrates a similar judgement:

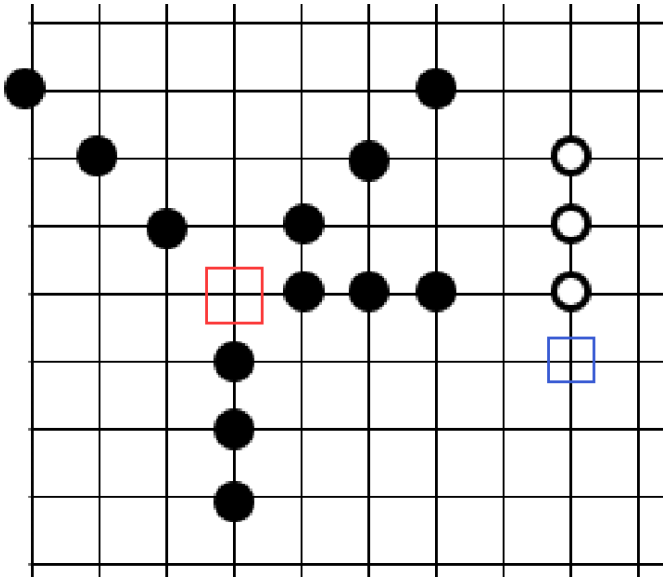

Figure 37. Even if the red place can block 4 L3, the blue place is still the better choice.

In Figure 38, the black has 4 L3, the white has only one L3 and the white will move the next step. It seems that the white has a bad condition, but the fact is, the player with white stones already wins. The white player just needs to move the next stone to the blue place. The black stones are one step late so it cannot make a five before the white. On the other side it is also not possible to block the two sides of white, so the white is going to win the game.

By this example, it is found: **Making 1 L3 to L4 > Blocking 4 L3**

Let the value of **Blocking 1 L3** be B, the value of **Making 1 L3 to L4** is at least 4B + 1. And when looking back to the last example, it can be found that **B** = **4A+1** and 4B = 16A + 4.

Along with this rule, we can give a specific weight for all cases in Table 5:

Table 5. The weight of different cases

Case	Role	Value	Symbol
L1	SELF	4B + 1	A
L1	ENEMY	B	B
D1	SELF	A - 1	a
D1	ENEMY	B – 1	b
L2	SELF	4D + 1	C
L2	ENEMY	4A + 1	D
D2	SELF	C - 1	c
D2	ENEMY	D - 1	d
L3	SELF	4F + 1	E
L3	ENEMY	4C + 1	F
D3	SELF	E - 1	e
D3	ENEMY	F - 1	f
L4	SELF	MAX (one more step means win) >> E	G
L4	ENEMY	0(already lose)	H
D4	SELF	MAX (one more step means win) >> E	g
D4	ENEMY	MAX - 1(one more step means lose)	h

6. Implementation

In the implementation chapter, some details of how the application is programmed by following the theory are represented. In addition, some parts of the code are presented.

6.1 Implementation of program strategy

The implementation of the strategy makes the program understand how to find the best grid.

To make a program know the weight of a grid, it needs to scan the four directions (**Figure 39**). The scanning is done by the analysisLine(int[], int) method. It takes in a String array, this array is the statement of the nearest 8 grids in one direction and the grid itself.

```
public int analysisLine(int[] n, int role)
```

Code Snippet 3. Analysis Line

Then, it will judge if there any continuous stones are nearby. If there are any, the value “Continues” will add one. At the end of scanning, the method will find how long the continuous query is and give the value to this query.

```
if(continus == 1){
    out = out*4+1;
    if(flag < 0) out--;
    if(live == 0) out = out/2 -1;
    else if(live == -1) out = 0;
```

```
}else if(continus == 2){
    out = (out*4+1)*4+1;
    if(flag < 0) out=out/4-1;
    if(live == 0) out = out/2 - 1;
    else if(live == -1) out = 0;
}else if(continus == 3){
    out = ((out*4+1)*4+1)*4+1;
    if(flag < 0) out = out/4-1;
    if(live == 0) out = out/4 - 1;
    else if(live == -1) out = 0;
}
```

Code Snippet 4. Give the value to this query based on the length.

The grid's value is the sum of the weight in four directions as mentioned above. The method "analysisGrid(int)" is used to sum up the weight in four directions.

```
public int analysisGrid(int n) {
    lineWeight[0] = analysisLine(scanGridsRow(n), -1);
    lineWeight[1] = analysisLine(scanGridsColumn(n), -1);
    lineWeight[2] = analysisLine(scanGridsDiagonalLTR(n), -1);
    lineWeight[3] = analysisLine(scanGridsDiagonalRTL(n), -1);
    int out = Math.abs(lineWeight[0]) + Math.abs(lineWeight[1]) +
Math.abs(lineWeight[2]) + Math.abs(lineWeight[3]);
```

Code Snippet 5. Analysis Grid

This process will be applied to every gird and the grid which has the highest value will be chosen

6.2 Implementation of UI and other function

This chapter explains how the layouts and functions are implemented. The

layouts are defined by XMLs, the functions are defined in Activities, the graph is drawn in views. The details can be found below.

6.2.1 Layout of menus

The layouts of the menus are decided by layout XMLs. Those XML files are located in resource folder. There is a “layout” folder for these XML files.

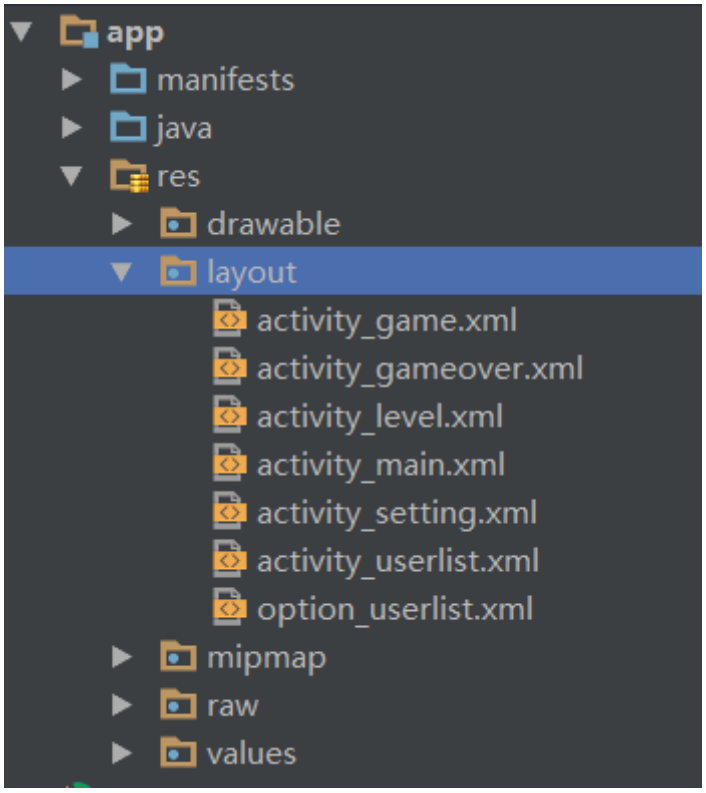

Figure 40. The layout files

Every XML file corresponds to an Activity class. In an XML file, the programmer needs to locate all the images and buttons. The Android Studio provides a component window which can easily drag a component to the

place where you want to locate. However, if the programmer wants to change the details, they need go to the XML file and to find the specific options.

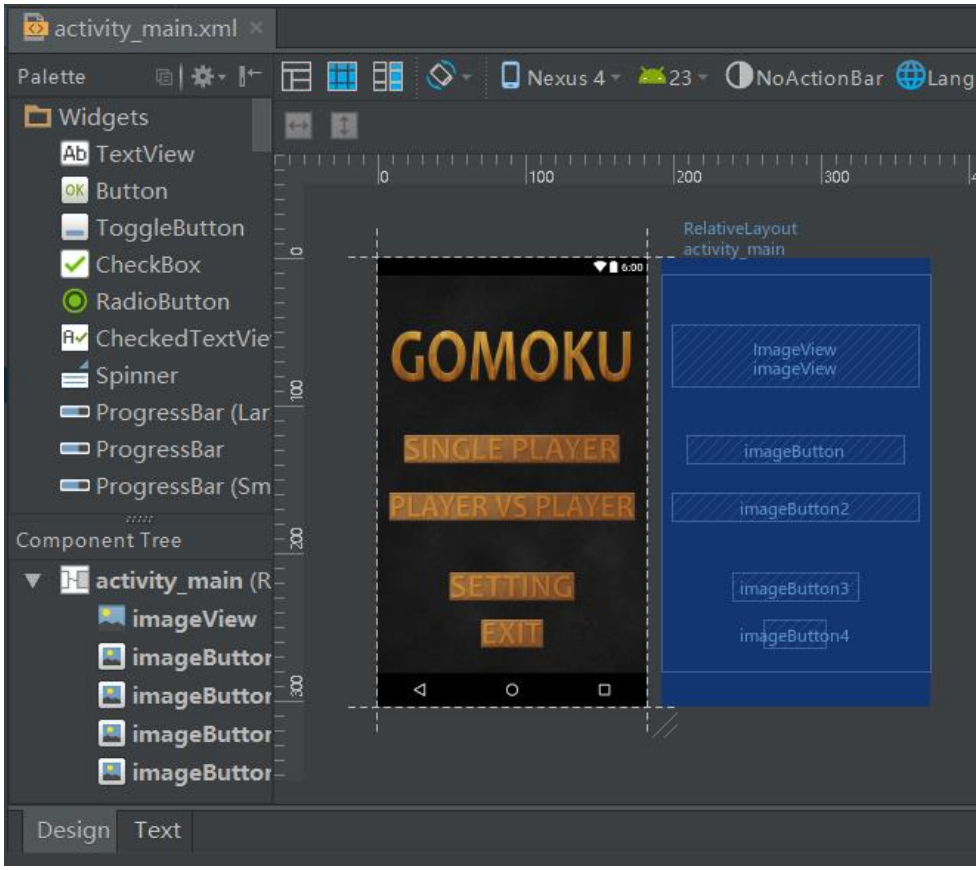

Figure 41. The component window

When specifying the details in XML files, every component will be given an ID. This ID will be used to connect the function of button and the location of button. In the next chapter, this will be described in more detail.

6.2.2 Functions of Buttons

Before the function is defined, the programmer needs to know which button

needs to be edited. This is the meaning of ID. This ID is defined in the XML file, if the programmer does not specify the ID of button, an auto-increased ID will be automatically generated when the button is created.

```
<ImageButton
    android:layout_width="90dp"
    android:layout_height="40dp"
    android:background="@drawable/main_exit_selector"
    android:layout_below="@+id/imageButton3"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="27dp"
    android:id="@+id/imageButton4" />
```

Code Snippet 6. An example of a button in XML file.

After a button is created in the layout, the programmer needs to give it a function. The functions are written in Activity classes. Firstly, the programmer needs to create an empty button and then set the click listener and the layout of the button by ID.

```
main_vscpu = (ImageButton) findViewById(R.id.imageButton);
main_vscpu.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {

        /**
         If pressed, intent to next activity
         */

        Intent intent = new Intent(MainActivity.this, LevelActivity.class);
        startActivity(intent);
    }
});
```

Code Snippet 7. The example of adding listener.

After adding the listener, the programmer can start to write what will happen after the button is pressed. The action in method “onClick” (Figure 20) will be run if the button is clicked.

6.2.3 Drawing the board

To draw the game board an interface which can be refreshed is needed. This interface can be implemented in Views. In GameView class, the pictures are imported and displayed on screen at the beginning. The initBitmap() method resizes all the pictures and makes them suit the different screens.

```
@Override
public void initBitmap() {
    // TODO Auto-generated method stub
    BitmapFactory.Options options = new BitmapFactory.Options();
    options.inSampleSize = 2;

    background=BitmapFactory.decodeResource(getResources(),R.drawable.bg
    _small,options);

    background2=BitmapFactory.decodeResource(getResources(),R.drawable.b
    oard);

    BlackDraw=BitmapFactory.decodeResource(getResources(),R.drawable.bla
    ck);

    WhiteDraw=BitmapFactory.decodeResource(getResources(),R.drawable.wh
    ite);

    scalex = screen_width / background.getWidth();
    scaley = screen_height / background.getHeight();
    scalex2 = screen_width / background2.getWidth();
    scaley2 = screen_height / background2.getHeight();
    scalexXY = screen_width / BlackDraw.getHeight() /17;
```

```
        scaleyXY = screen_width / WhiteDraw.getHeight() /17;
        bg_y = 0;
        bg_y2 = bg_y - screen_height;

        Log.d("Init : "," Init finished !");
    }
```

Code Snippet 8. The initBitmap() method.

In Figure 21, scale is calculated by the rate between the screen size and the image size. This scale will be used when the images are drawn. In this way, the image can suit the size of the screen automatically.

Every time the players click the screen (when players move a stone) the game board will be refreshed and updated to the newest move. The statement of the stones on the board will be stored in views class also. By doing this, the graph can renew faster.

After defining the objects in the view, it needs to be set to an activity. A view will be created in activity and the content of the view will be set to the activity and the view can be displayed.

```
GameView gameview = new GameView(this, game);
setContentView(gameview);
```

Code Snippet 9. Content of the view will be set to the activity

6.2.4 Background Music

The background music is played by a service. The service will start playing the background music and looping it until the user stops it. When the service is created, the music player will be started.

```
@Override
public void onCreate(){
    super.onCreate();
    player = MediaPlayer.create(this, R.raw.bgm);
    player.setLooping(true);
    player.start();
    System.out.println("Service is started");
}
```

Code Snippet 10. The onCreate method will be run when the service is created.

After defining the music player in the service, the service needs to be activated in Activity. The background music need to start playing after the application is run. Therefore, when the application is started, an intent object will activate the service.

```
if(setting[1].equals("1")){
    Intent startIntent = new Intent(this, musicSer.class);
    this.startService(startIntent);
}
```

Code Snippet 11. Intent the service

In the setting menu, the background music can be turned on/off and also the data in the database can be changed. So the music playing setting will be kept when the next time the game is run.

```
dbManager dbM = new dbManager(SettingActivity.this);
dbM.initSetting();
setting = dbM.LoadSetting();
/**
 if the "MUSIC" is turned off, turn is on and start playing music
 */

if(setting[1].equals("0")){
	Intent startIntent = new Intent(SettingActivity.this, musicSer.class);
	SettingActivity.this.startService(startIntent);
	dbM.updateMusicOn();

	/**
	 if the "MUSIC" is turned on, turn is off and stop playing music
	 */

}else if(setting[1].equals("1")){
	Intent stopIntent = new Intent(SettingActivity.this, musicSer.class);
	SettingActivity.this.stopService(stopIntent);
	dbM.updateMusicOff();
}
dbM.closeDB();
```

Code Snippet 12. Change the data in the database

6.2.5 User list

User list is a special Activity for displaying information in a list. The list properties can be defined in an XML file. For example, the font size, the format, the color of the text. The Activity will read the format of the list when it is created.

```
<LinearLayout android:orientation="vertical"
	android:layout_width="wrap_content"
	android:layout_height="wrap_content"
	android:layout_margin="8px">
```

```
    <TextView android:id="@+id/name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#FFFFFFFF"
        android:textSize="35px" />
    <TextView android:id="@+id/total"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#FFFFFFFF"
        android:textSize="18px" />

</LinearLayout>
```

Code Snippet 13. The defining of font size and format.

In Activity the programmer can define the click functions for the item in the list. The click listener can be added to every item, and the user select function in this application is implemented by clicking the list item.

```
@Override
protected void onListItemClick(ListView l, View v, int position, long id) {
    Log.d("click", ""+position);
    text = new EditText(userlistActivity.this);
    if(position == (length-1)){
        builder.setTitle("New User");
        builder.setView(text);
        builder.setPositiveButton("Create",new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface dialog, int whichButton) {
                if(text.getText().toString() != null) {
                    Log.d("text : ",text.getText().toString());
                    dbManager dbM = new dbManager(userlistActivity.this);
                    dbM.addNewUser(text.getText().toString());
                    dbM.closeDB();
                    Toast.makeText(userlistActivity.this,"Welcome! "
                    + text.getText().toString(),Toast.LENGTH_LONG).show();
                    userlistActivity.this.finish();}
```

```
            }
        });
        builder.show();
    }else{
        String username = (String)list.get(position).get("name");
        dbManager dbM = new dbManager(this);
        dbM.updateRecordLastUser(username);
        dbM.closeDB();
        Toast.makeText(userlistActivity.this, "User has been changed !
                " ,Toast.LENGTH_LONG).show();
                userlistActivity.this.finish();
        }
    }
```

Code Snippet 14. The clickable item in list

6.2.6 Database

To do all operations with the database, a special class which is used to send requests and catches the result needs to be defined. This class will automatically generate the database and the tables if this is the first time the application is run or if the database is deleted wrongly.

```
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE IF NOT EXISTS setting (_id INTEGER PRIMARY KEY
    AUTOINCREMENT, color INTEGER, music INTEGER)");
    db.execSQL("CREATE TABLE IF NOT EXISTS record (_id INTEGER PRIMARY KEY
    AUTOINCREMENT, name VARCHAR, total INTEGER , win INTEGER, lastuser
    INTEGER)"); }
```

Code Snippet 15. The generation of tables

In addition, the method of operating the database will be defined in the class. If the programmer needs to operate the database in other classes, the object can be created and call the method directly.

```
if(dbM.queryIfUserExist()==0){
        dbM.addNewUser("Player");
}
```

Code Snippet 16. An example of the database operation

7. Testing

The testing is divided into two parts: an algorithm test and a functions test. The algorithm test is about testing of the algorithm of the game AI to see if it can reach a level that can make players feel there is a challenge or beat some other AI of similar applications on the internet.

7.1 Algorithm test

Some volunteers were inverted to play this game. The test result can be found in Table 6.

Table 6. Testing result with human

Player	Total	AI win	AI lose
A	10	7	3
B	10	8	2
C	10	7	3
D	10	6	4

In addition, some five-in-a-row applications were downloaded from Google play and the highest level of the applications are used to play against the algorithm.

Table 7 Testing result with other application

Application name	Total	AI win	AI lose
五目并べ	10	7	3
EveryOMok	10	5	5
GomokuFree	10	6	4

The algorithm won more than a half of the games of different applications. Some algorithms are still better than this one.

7.2 Functions and test

This chapter describe the testing result of functions which are required in the Chapter “Functions Requirement”

Table 8. The test result of functions

No.	Function	User Action	Expected result	Result
1	Start application	Click the APP on an Android device	The main menu is displayed, and the database is initialed.	The main menu is displayed, and the database load normally
2	Single mode	Click the “Single mode” button	The level selection menu is displayed	The level selection menu is displayed normally
3	Single mode Game	Select the level	The game board is displayed	The game board is displayed
4	Game running	Click on game board	The AI plays by turns with you, and the move of AI is logical.	The AI plays by turns with players. Logic works well.
5	PVP mode	Click the ”Player vs Plyer” button in the main menu	The game board is displayed, and players can play game by turns	The game board is displayed

No.	Function	User Action	Expected result	Result
6	Setting	Click the ”Setting” button in the main menu	The setting menu is displayed.	The setting menu is displayed.
7	BGM	Click the ”ON/OFF” button in the setting menu	The background music is changed.	The button is working correctly.
8	Account system	Click the ”Change User” button in the setting menu	The user list is displayed.	The list is displayed correctly.
9	Setting save	Exit game, and run it again.	The setting which is changed last time should be kept.	The change is saved.
10	Exit	Click the ”Exit” button in the main menu	The application exit without crashes, and background music is turned off automatically	The application works well

8. Summary

The project achieved its goals successfully and most functions work well.
The logic test is not as easy as the functions test. When testing the function, people can easily give an answer: “yes” or “no”. If a function is not working, it is not good. However, the logic of an algorithm does not have a clear level to judge. The only thing can be done is to asking people to play with it.

Because it is difficult to find a master of five-in-a-row, the people who have participated in the test may be not very good at strategy games like Five-in-a-row. It is still hard to decide the level of the algorithm. Still, there is no doubt that the game brings some challenge to players. The algorithm is logical most of the time. Helpfully this algorithm can be improved more in the future.

9. Conclusion

The main functions which are mentioned in “Functions requirement” are implemented. Simply, the application allow the users play against an AI with changeable level. Also, players are able to play against their friends. In addition, users can create new accounts and change accounts and set ON/OFF to the back ground music.

During the programming, many problems occurred. One of the biggest problems in the process was that at the beginning the application crashed for some reason. One of the reasons was, when dealing with a bitmap, there are always ridiculous memory space was asked by the system. It was about 1 GB. But the whole size of the application is about 50MB, this makes the requirement of the memory strange.

In the end the problem was found. The reason of this was that high-resolution pictures were used in the application. Debugging means more than finding bugs in the code. The format of resources, the size of the pictures, as well as many details need to be attended to. This is the greatest lesson I have learned in this project.

Anyhow, the project succeeded and this project will serve as a great experience in the future work and study.

10. References

1. Gomoku, Wikipedia ,
https://en.wikipedia.org/wiki/Gomoku
Last Access 24.4.2017

2. Activities | Android Developers,
https://developer.android.com/guide/components/activities/index.html
Last Access 21.4.2017

3. Service | Android Developers
https://developer.android.com/reference/android/app/Service.html
Last Access 21.4.2017

4. ListActivity | Android Developers
https://developer.android.com/guide/components/activities/index.html
Last Access 19.4.2017

5. What are C++ features missing in Java?, GeeksQuiz,
http://quiz.geeksforgeeks.org/c-features-missing-java/
Last Access 17.4.2017

6. XML Introduction - W3Schools
https://www.w3schools.com/xml/xml_whatis.asp
Last Access 22.4.2017

7. Android – History,
https://www.android.com/history/
Last Access 22.4.2017