COMPS413F Application Design & Development for Mobile Devices

Group Project in Android Application Development

2048 Game Application

Chiang Pui Shan 13360332

Lai Joe Wing 13261888

Lam Yin Ying Elayne 12239282

Mak Fu Hang 13427125

Martinez Kendrick Maglaque 13164554

Table of Content

Application Overview

We created a unique 2048 game by using Android Studio. The application can run on Android Studio and android phones. In the game, players are presented with a 4x4 grid, where they must combine matching numbers to reach the elusive tile with the number 2048. Players can move the tiles in four directions – up, down, left, and right – with every move combining adjacent tiles of the same number into one tile with the sum of the two numbers. The game continues until the grid is full and no more moves can be made, or the player successfully reaches 2048 square.

The target audience for 2048 is broad, encompassing players of all ages and skill levels. Its intuitive design makes it accessible to casual gamers, while the strategic depth appeals to those who enjoy puzzle challenges. Students can benefit from the game's cognitive skills enhancement, practicing critical thinking and problem-solving in an enjoyable format. Families can also engage together, making it a fun activity for all.

Additionally, 2048 serves as a fun way to relieve stress, offering satisfying gameplay and a sense of achievement as players see their scores rise. With its easy-to-use interface and engaging challenges, 2048 has become a favorite among mobile gamers, making it an ideal choice for a quick, entertaining experience on Android devices.

Project GitHub Link: https://github.com/s1336033/2048-Android

User Interface Design

Splash Screen

Main Screen

About Screen

Game Screen

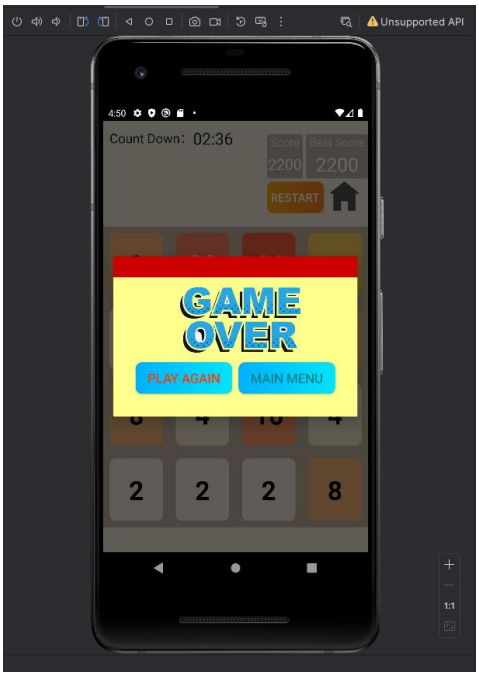
(Randomly start from “2” square)

Game Screen

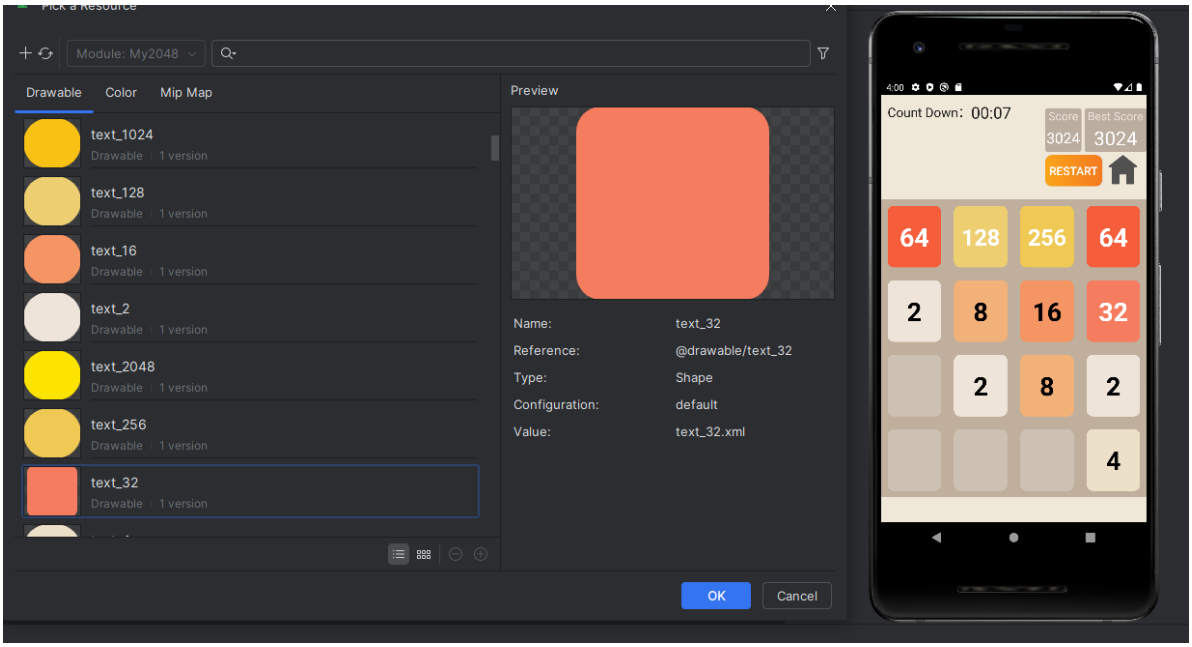
(Randomly start from “4” square)

History Score List

Game Over Window

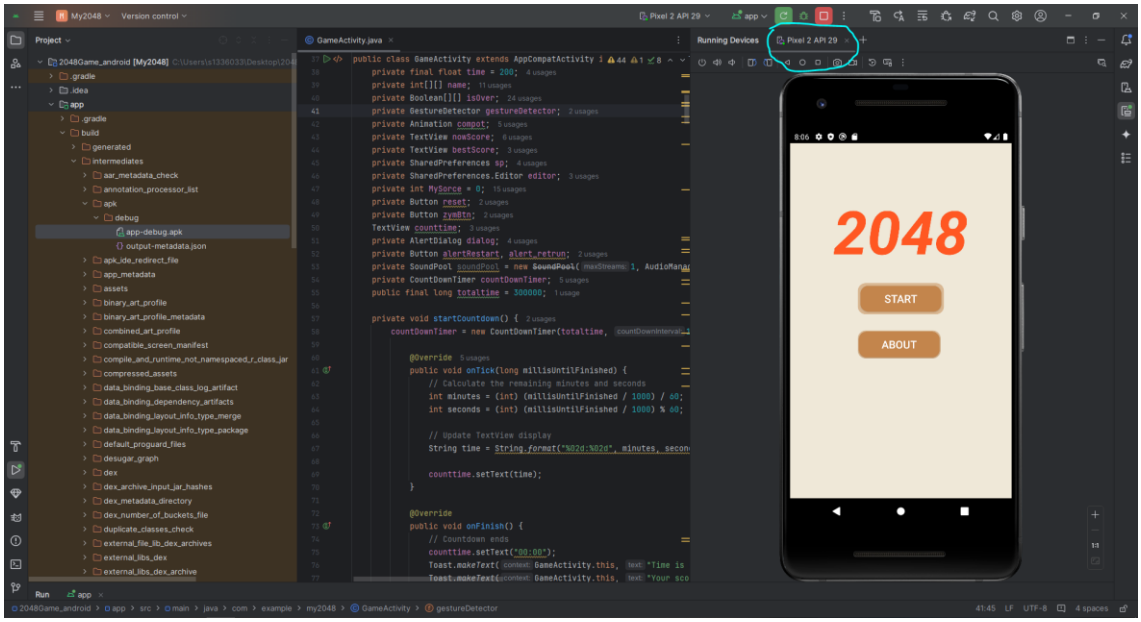


The grids are designed with different colors according to different values.

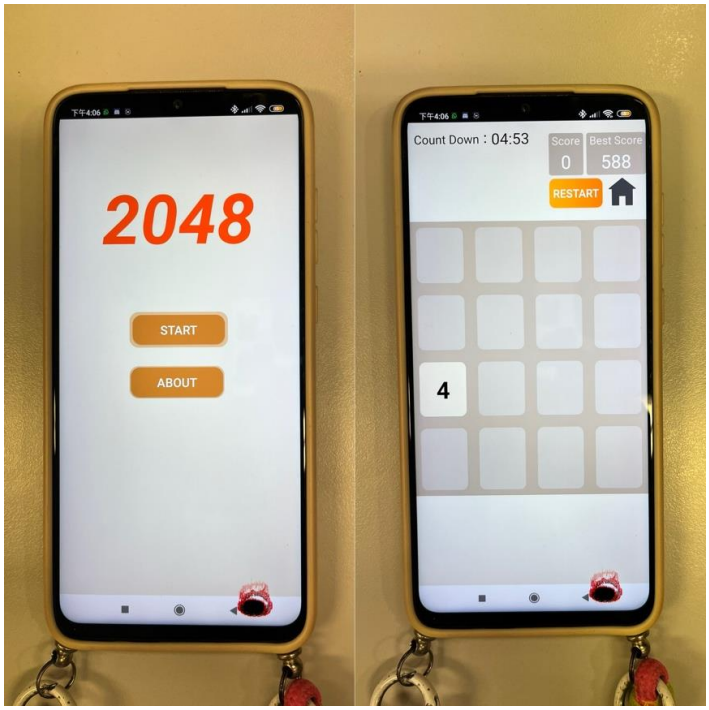


Technical Requirements

1. **Android API is applied**

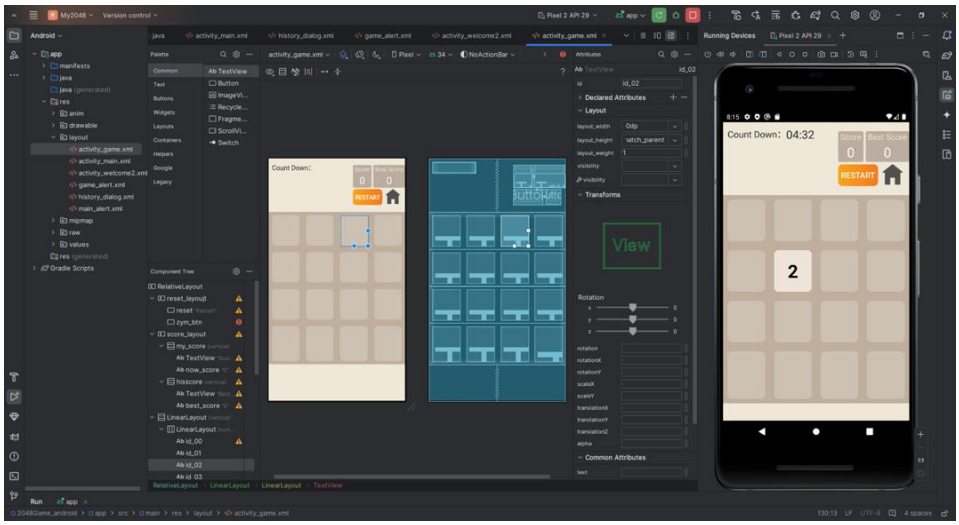


The application can run successfully on Android Studio Virtual Devices (Such as on Pixel 2 API 29).



It also can run on android phones, such as Samsung, Xiaomi, Vivo etc.

2. Using GUI



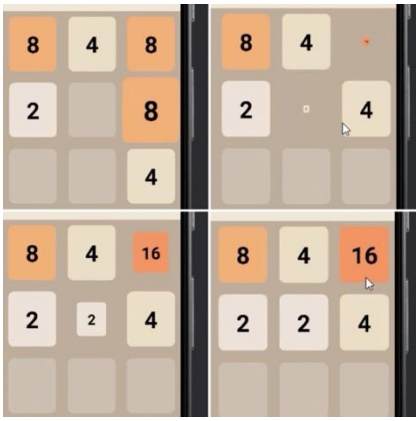
2.1 Layout

The Game board layout is designed in *Activity.game_xml*, the layout XML files.

2.2 GUI Widgets

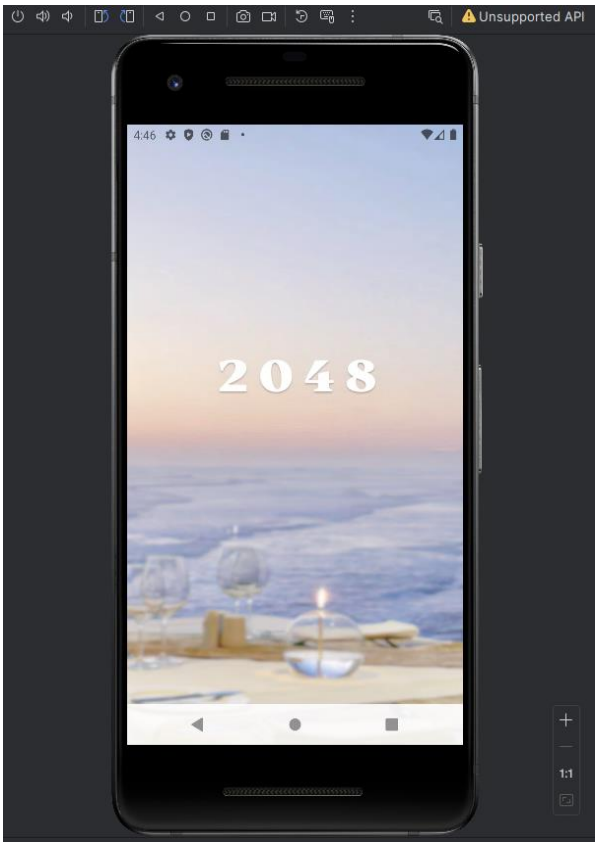
The Main Screen and Game Screen are using lots of GUI widgets to design, included buttons such as (e.g. START button, ABOUT button, RESTART button etc.), textviews (e.g. History Score title, About Screen content and Score), grids (e.g. “2” or “4” Grids randomly generate on the 16 grid pits).

2.3 Transformations: Changing Scale



When two grids with the same number collide, totally new grids will appear in the grid pit from smaller size to larger size. The newly generated 2 or 4 grids each round also appear with same expansion effect. This enhances the game's dynamism, elevating the overall experience for players.

3. Splash Screen



The game logo 2048 in white will be displayed in the middle of the splash screen, and the background is a using a relaxing and bright colored wallpaper.

```
42      private void startCountdown(){  1 usage
43          countDownTimer =new CountDownTimer(timeLeftInMills,   countDownInterval: 1000){
44
45              @Override  5 usages
46              public void onTick(long millisUntilFinished) {
47                  timeLeftInMills=millisUntilFinished;
48                  int secondsRemaining = (int)(millisUntilFinished/1000);
49                  tv_countdown.setText(secondsRemaining + "s");
50              }
```

The splash screen will remain for 3 seconds when 2048 APP is clicked to open.

4. **An “About” screen callable from menu command**

Press the “ABOUT” button on the main screen, then the About the Game screen page will pop up.

The **About Screen** displays the subject names, Student IDs of all our group members, and the date the Application was successfully developed.

Rules of 2048 are also shown below to facilitate players to know how to operate the game.

5. **Data storage**

Database Used: **LiteORM**

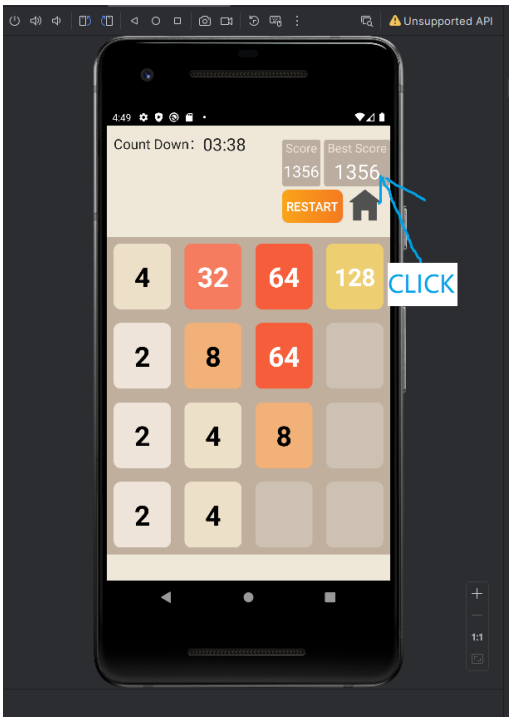
Score Storage Mechanism: The **savescore()** method in GameActivity.java utilizes **LiteOrmDBUtil** to insert scores into the database.

```
28  import com.example.my2048.util.LiteOrmDBUtil;
29  import com.example.my2048.util.ScoreBean;
30  import com.example.my2048.util.SoundPlayUtils;
```

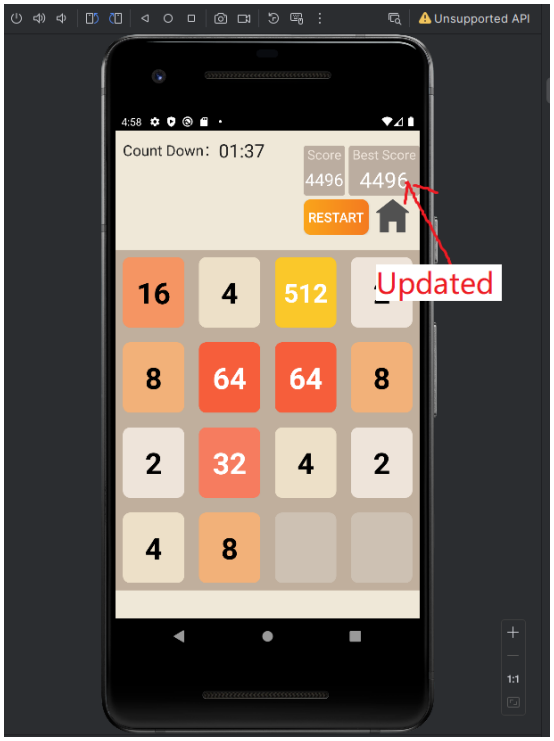
```
682     public void savescore() {  4 usages
683         ScoreBean s = new ScoreBean();
684         s.score = nowScore.getText().toString();
685         LiteOrmDBUtil.getLiteOrm( context: GameActivity.this).insert(s);
686
687     }
```

History Score List records all game result sequentially.

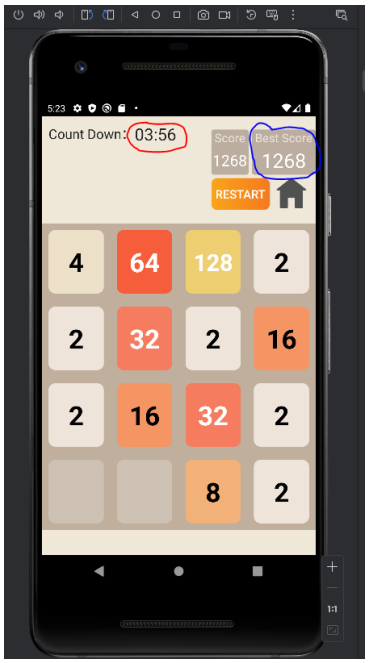
Click “Best Score” column in the Game screen, History Score List will pop out.



After the scores are recorded, the highest score will be displayed in the “Best Score” column in the Game Screen.
Best Score will be updated simultaneously until a higher score appears.



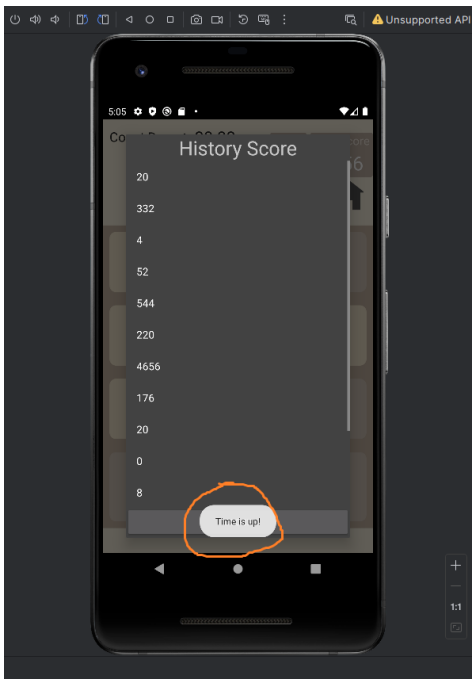
6. Multithreading

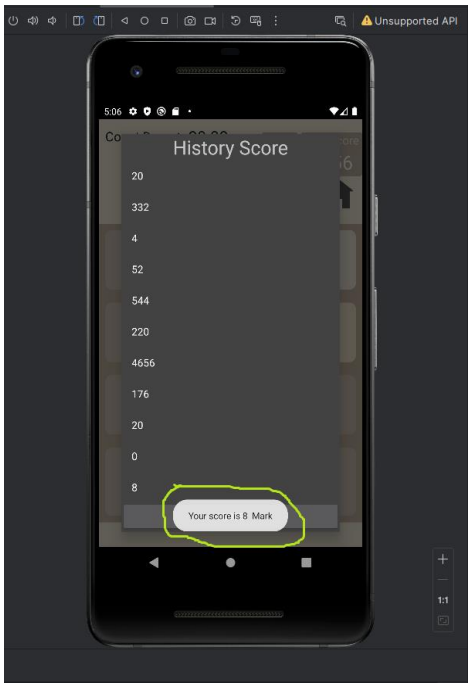


The multithreading functions are **Scoring** and **Timing** function.

Each round is limited to 5 minutes.

Calculating the player's score and the timer counting down **at the same time**.



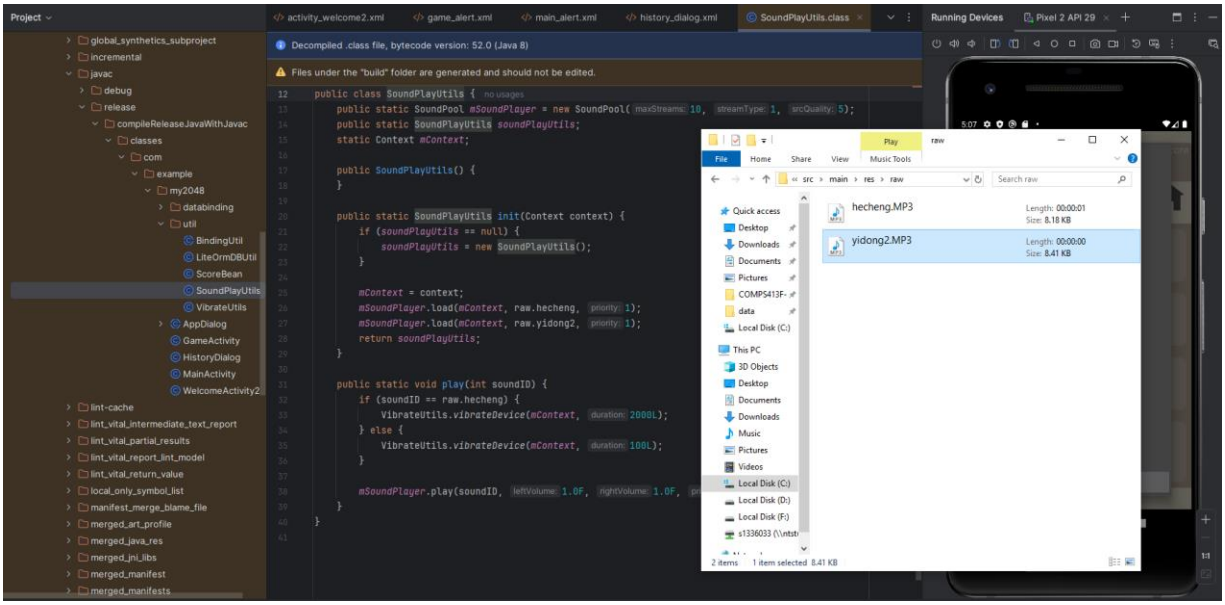


When the countdown time is up, the history list and “Time is up!” message will pop out.

Then, “Your score is __ mark” message will also pop out.

7. Multimedia

7.1 Sound effects

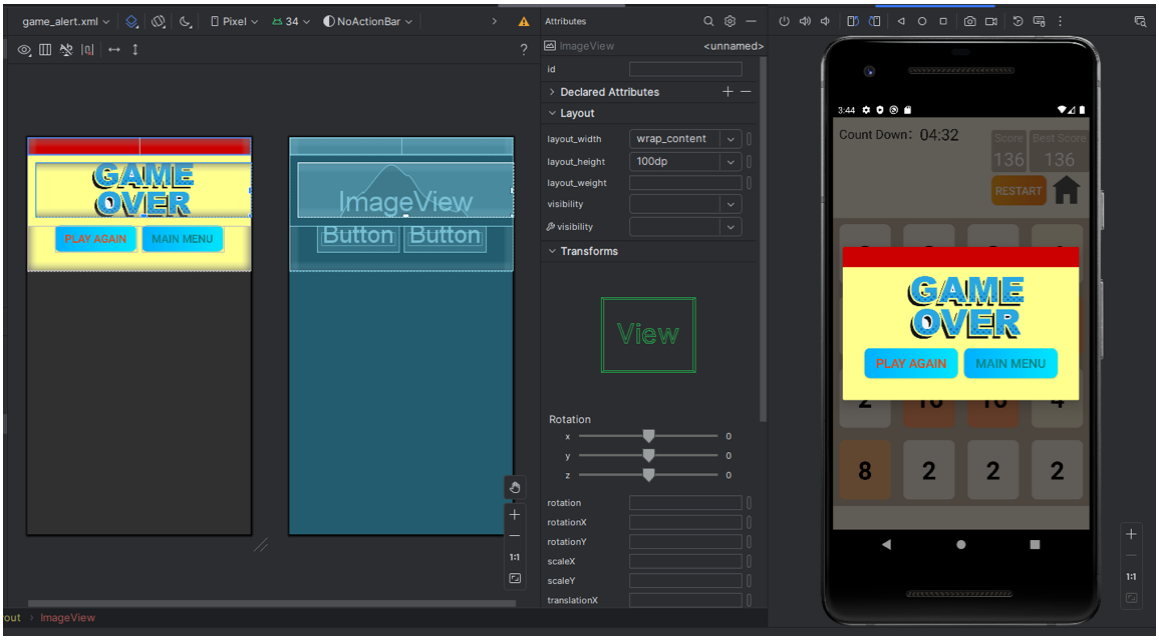


Sound effects are inserted in *SoundPlayUtils.class*.

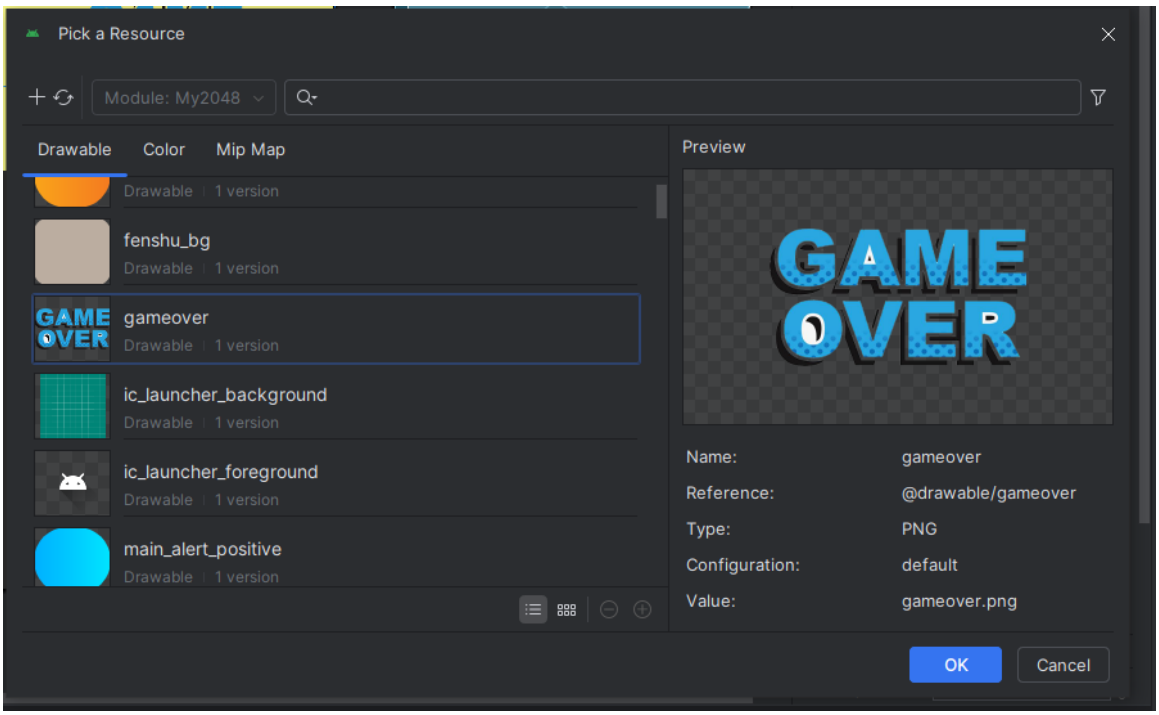
```
216     private void setLeft(int i, int j) {
217         for(int h = 0; h < 4; ++h) {
218             if (i == h && j != 0) {
219                 for(int w = j; w > 0; --w) {
220                     TextView ahead = (TextView)this.findViewById(this.name[h][w - 1]);
221                     TextView local = (TextView)this.findViewById(this.name[h][w]);
222                     if (ahead.getText().toString() != "" && ahead.getText().toString().
223                         SoundPlayUtils.play( soundID: 1);
224                         int num = this.parse(ahead.getText().toString());
```

When the same mesh is synthesized, a sound effect will be played to let the player know that they are getting some score, making the game more dynamic with the utilization of sound feedback.

7.2 Image



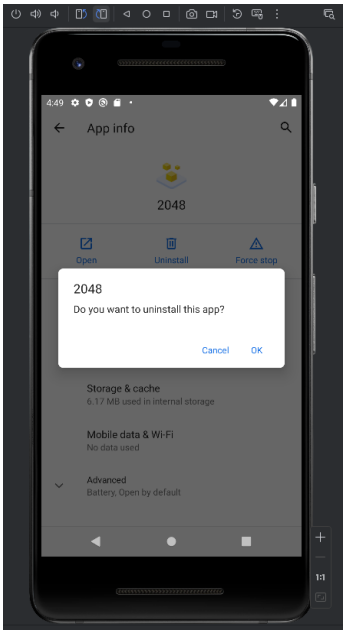
The layout of the Game Over Window uses additional image。



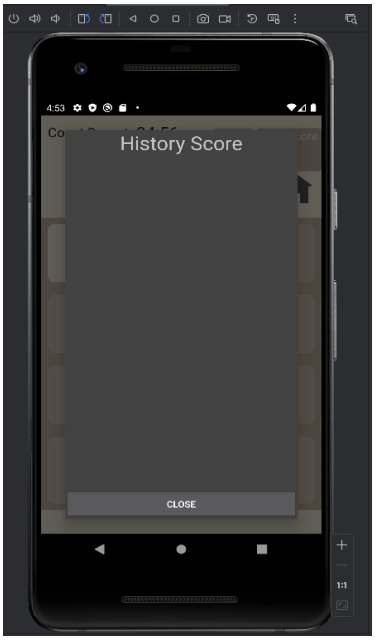
To enrich the game layout, we additionally designed the artistic text of “Game Over”.

Extra Significant Function

Clear Up Data Function



When users want to clear all results in the History Score list, they can uninstall the 2048 application and then re-download it again.



After redownloading the application, the Best Score will be reset to 0 and with the History Score List returning to being empty in the first time to opening the game.

Conclusion

Our project primary goal is to create a unique Android application, specifically a version of the popular 2048 game. The 2048 game is not only entertaining, but also an effective way for players to enhance their cognitive abilities through engaging gameplay. During the development period, our plans were based on continuous research and iterative attempts to refine and improve the gameplay. Ensuring that users will have a smooth and enjoyable experience. The development process establishes a clear progression framework that allows for regular updates and enhancements as the project evolves.

The key to our success lies in the combined efforts of our team members. Each member contributes unique skills and perspectives, using teamwork and dedication are crucial to creating an engaging game. Driven by a shared commitment to excellence, we have crafted a beautiful version of the 2048 game.