## 一． 实验目的

1、掌握中小型项目的需求分析、系统设计、编码、测试，以及项目的部署运行，具备复杂工程问题的解能力；

2、掌握综合应用 Android 知识点进行管理和软件开发的能力；

3、掌握文献检索、系统建模、文档写作等方面的能力；

4、综合运用所学知识，结合实习需求完成项目开发。

## 二． 实验内容

每2~5同学为一组，建议每组人数为2～3人，每个小组指定组长1名，完成运行于Android的原生App的设计开发以及综合实验报告。会根据运行效果、界面设计、控件使用、布局使用、服务、广播、图像、视频、Shared Preferences、本地保存、SQLite、Content Provider、网络使用和多线程等方面的实现给分。

确定综合设计题目后进行分工合作，需要让每位同学都能开发至少一个具体的功能模块。题目确定后，由组长在课堂派中新建小组，小组名称用**组长学号姓名－题目名称**命名，所有小组成员加入该小组。具体要求如下。

**1、一个Android原生App的设计与开发**

**2、撰写作品说明书**

内容包含：选题（系统简介、为什么要做该系统）、需求分析，数据库设计(分为逻辑设计、概念设计和物理设计)、详细设计、系统实现，系统测试，存在的问题及思路、总结。

## 三． 选题

 PixAdventure是一款由三个物联网工程专业在读本科生自主研发的一款全新益智解谜冒险游戏。游戏设定中，故事发生在一个遗弃在路边的老旧游戏机内的黑白像素世界，在这里，一个个充满未知与挑战的关卡正等待着玩家去探索。玩家将用智慧和勇气，去解救同班同学璃花——同时，逐步发掘PixAdventure世界的真相。

游戏用户端基于Android Studio开发，后端采用Linux系统。我们通过本次安卓应用项目的开发对安卓应用的基本开发思路、前后端架构和协调等方面进行充分的实践学习。

## 四． 需求分析

 PixAdventure益智解谜游戏的玩法参考了扫雷和康思(Conceptis Puzzles)的填色游戏(Fill-a-Pix)。我们通过Android Studio实现用户端的安卓应用开发，用户可以在安卓手机app中进行游玩。在后端，我们通过部署在阿里云服务器上的Linux系统的数据库等实现数据保存与传输，在前端与后端之间，我们采用Springboot等进行协调连接。

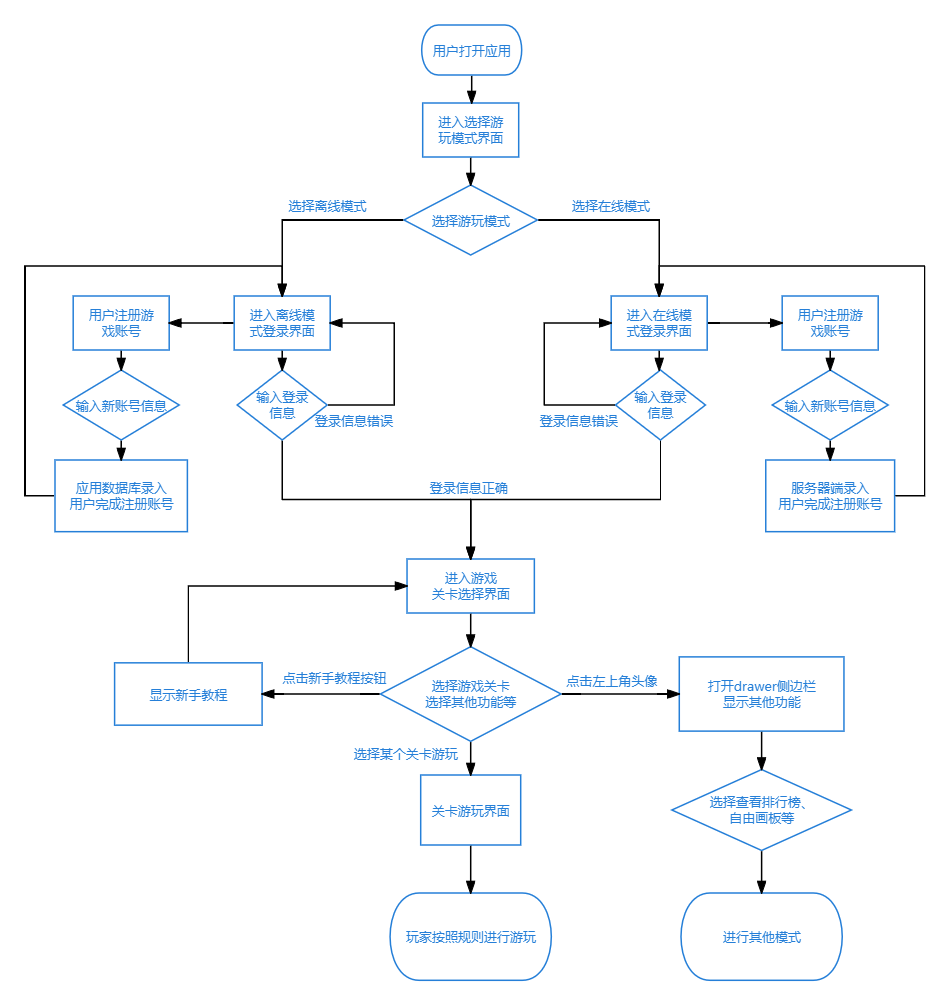
Linux服务器数据库端

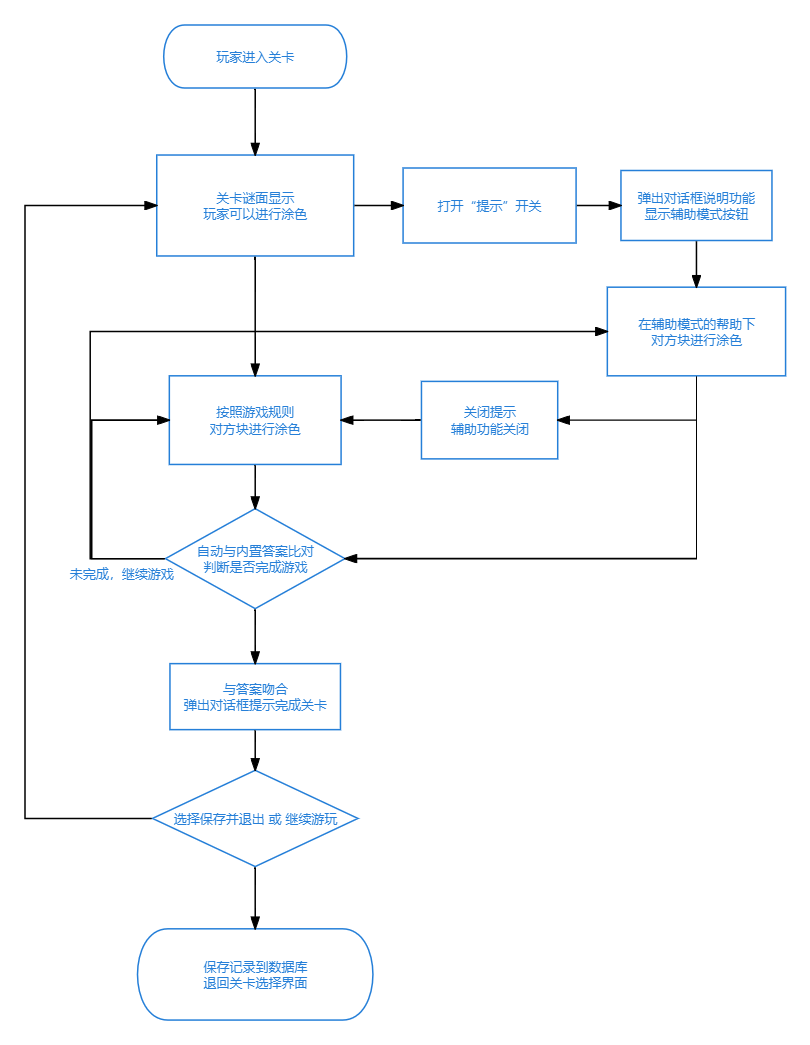
Springboot连接前后端

Android用户端

## 五． 概要设计

整个应用的基本流程和具体游玩的过程设计如下图：





## 六． 详细设计

1. 对于基本功能：

在各自模式（在线或离线模式）的登录界面，用户可以输入已保存的账号登录，或者是点击注册按钮进行新账号的录入。在注册界面下，用户可以选择头像，输入账号和密码，在点按保存后，应用会根据各自模式将新账号保存到本地数据库或线上服务器，随后退回登录界面进行登录。

在关卡选择界面，玩家有八个关卡可以选择，关卡难度依次增大，玩家可以根据自身水平自由选择将要游玩的关卡。在该页面的显眼处还设计了一个“新手教程”按钮，点按该按钮后玩家可以阅览游戏的基本玩法，帮助新玩家更快上手。此外，类似QQ等应用，玩家可以点击左上角头像打开侧边栏，侧边栏中有各种其他功能，如查看游戏剧情、排行榜和自由画板等，这些功能下文会提到。使用侧边栏的设计提升了用户交互友好性，并且预留了足够的空间为后续新功能开发提供便利。

在关卡的游玩界面，我们实现了康思的填图游戏，玩法规则与康思原版填色一致。可以通过用户单击、长按滑动等不同操作实现方块颜色改变（白色色块点击一次涂为黑色，再次点击涂为灰色，再次点击回到白色。玩家还可以拖动使得路径上的色块一同变色），通过应用后台自动判断用户的输入是否合法，以及填色结果是否通关等。

此外，在关卡的游玩界面我们还创新性地设计了一个原版没有的辅助开关，设计该开关的目的是照顾游戏过程中有困难的玩家。打开该开关后，游戏的交互玩法将会被改变，在辅助模式下玩家可以自动长按方格，系统会自动将点击方格周围九格范围内的色块涂为正确颜色；打开辅助模式中的错误开关，系统会自动显示玩家涂色的错误，将玩家涂色与实际涂色不相符的地方自动标红；点击辅助模式下的答案按钮，系统会弹出对话框显示该关卡的正确答案。

在关卡模式中，玩家可以随时点击保存按钮，玩家游玩的地图数据会根据各自模式被存入本地数据库或是传入后端服务器。保存的数据还会在关卡选择界面中显示实际的完成度（玩家正确填涂与正确答案的百分比），通过图形化的显示使得玩家对每个关卡的完成情况一目了然。

在排行榜界面，应用会检索各自模式中的账号，按照每个玩家的分数由高到低动态生成排行榜。每个玩家的数据占一行，显示排名、账号名称、头像和分数等。分数会根据每个账号的所有关卡完成情况有所不同。

之所以分别设计在线和离线模式，是为了满足玩家们的各自不同情况。对于在线玩家，每个注册的玩家都会被记录入服务器，可以在排行榜上互相看到对方的分数排名，由于我们采用的是网络服务器，玩家范围更广；对于离线模式，注册的账号会被保存在app所安装的设备上，玩家在离线模式下看到的排行榜是所有本地账号的排行榜，由于不依赖网络服务器，离线模式适合给个人、家庭或社区等玩家及群体游玩，进行家庭排名比拼或集体团建等。

在像素画板界面，玩家可以自由涂色，没有通关条件，自由创作作品。涂色后的结果可以点击保存到本地，所有人都在应用所安装的设备上共享一块画板，从而实现多人创作。

点击侧边栏中的“剧情”或“关于我们”，进入不同的信息显示页面以供玩家阅览。剧情使得游戏更饱满，提升玩家的游玩体验和游戏完整性。

2. 对于用户友好性：

我们还在以下方面从普通用户操作的角度考虑，致力于提高我们应用的用户友好性：

上文1中提到的所有界面都设计了一个返回按钮，玩家可以点击该按钮返回上一级页面。与此同时，用户也可以直接点击系统的返回键（在全面屏手势下是从边框向内拉），其实现的功能与应用中的返回键一致。当触发返回时，不会直接造成应用关闭，提升了玩家的应用使用体验。

当返回键或清空键被按下时，应用会弹出提示框询问是否确认退出或清空画板，这一举措很大程度上防止了用户误触发导致的进度丢失，提高了用户满意度。

在关卡游玩界面和像素画板界面，我们充分考虑到了玩家的不同操作，极大地减小了玩家在多指点击等情况下填色bug的产生，提高游戏稳定性，保护了玩家在游玩过程中的体验。我们还设计了动画效果，手指接触过的色块会由紫色到目标颜色渐渐改变，同时还有点击音效，渐变颜色动画和点击音效亦不会因为玩家的快速操作而卡顿，提升了玩家的游玩乐趣和满意度。

在关卡选择界面、游戏游玩等界面我们都精心挑选了舒缓悦耳的背景音乐，玩家可以根据自身意愿点按页面中的音乐开关来暂停或打开背景音乐，应用在后台运行再恢复之后，音乐的实际开关状态亦不会改变，应用在关闭后或切换到下一个页面会被释放资源停止音乐。

我们还通过界面交互设计、图标设计和背景设计等精心美化了整个游戏，使得应用整体简洁而内容丰满，内容直观且准确，玩家游玩过程心情愉悦。

在登录游戏、进行一些点按操作时，我们的应用会弹出Toast提示或对话框等显示相应信息，使得用户对我们应用运行的状况一目了然，对游戏的操作方式更明了。

对于新玩家和游戏中有困难的玩家，我们分别设计了新手教程和游戏提示辅助，充分照顾了不同用户人群的不同需求，尽力使得每一位游玩PixAdventure的玩家都能乐在其中。

3.关于游戏在线功能

在游戏在线功能方面，我们选择了稳定的前后端数据库交互方案。前后端通过spring boot连接，通过Xshell连接部署在阿里云服务器的linux系统进行jar文件储存和自动脚本编写运行等操作。

在安卓应用客户端，玩家的账号信息和每个账号对应的地图数据会被保存到sqlite本地数据库中，玩家的游玩数据不会因为应用关闭而丢失，实现了离线保存功能；如果玩家在游玩在线模式，本地的数据库会和线上的服务器数据库进行数据交互，本地保存的数据会被储存到云端的数据库，自此可以实现在线保存，玩家在应用本地数据清除之后（例如删除应用）也可以通过登录账号，继续云端的游戏进度。

此外，由于我们的Linux后端服务器是部署在阿里云服务器上的，所以我们的游戏只要网络连接稳定，都可以进入在线模式，不需要限制在小范围的局域网中；此外，我们在Linux后端服务器中编写了自动检查springboot是否运行的脚本，每隔10秒检查一次，若未在运行则重启，保障了在线游戏服务器的稳定与可靠性，同时也省略掉了繁杂的开启和检查服务器的操作。

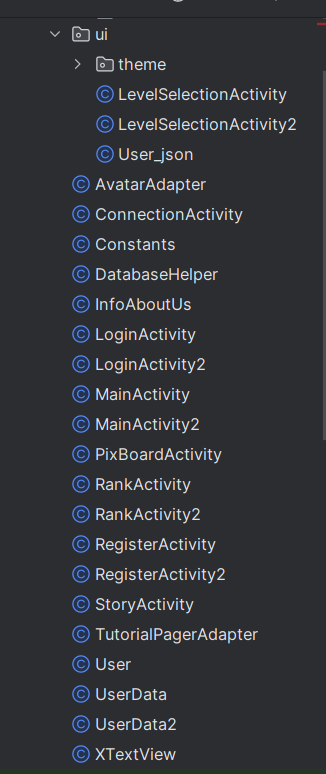
## 七. 系统实现

**（一）、主菜单**

考虑到可以让玩家在不联网的情况下也可以玩我们的游戏，我们分别设计了离线模式和联机模式。两种游玩方式可以在开始菜单选择，如下图：

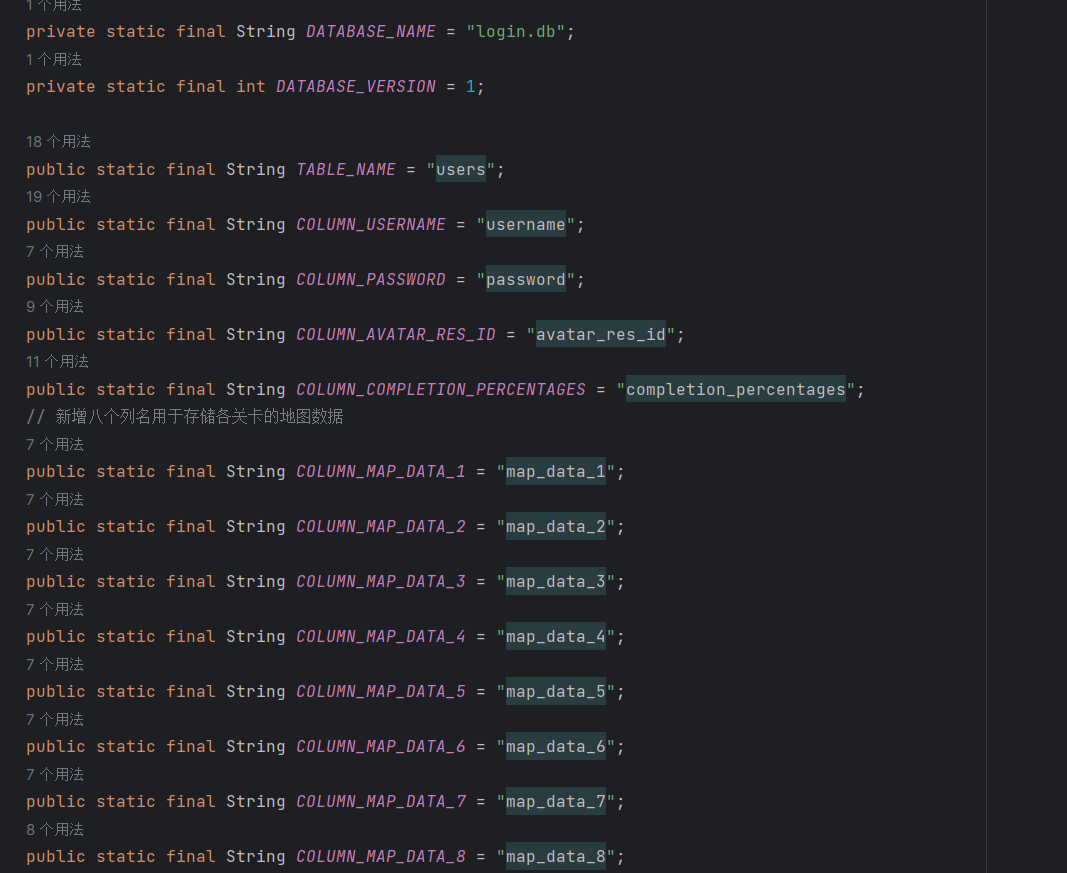
玩家可以根据自己需要选择不同的模式，两种模式的主要玩法相同，区别是联网模式下需要连接我们的服务器才可以登录在线账号游玩，离线模式则是作为一款单机游戏，适合家庭或社区等共用一台设备游玩。

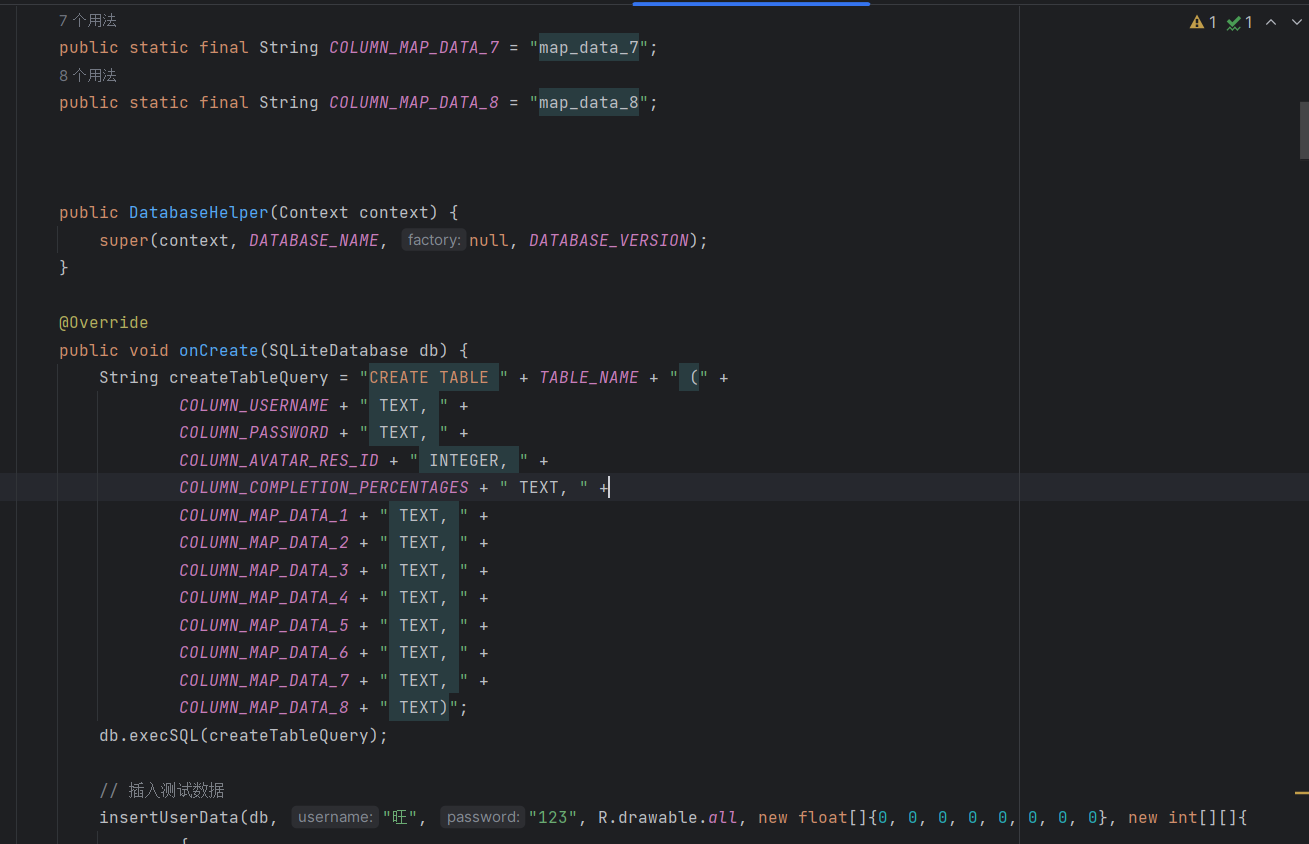
两种模式实现的不同点在于联机模式下使用的是服务器上的MySQL数据库，通过springboot设计好各种请求，对MySQL进行增删改查等操作，离线模式下则使用sqlite进行数据交互，所以两种模式的账号互不干扰，在离线模式下注册的用户不能在联机模式下登录。此外其他部分的实现效果基本相同，通过在不同的activity中书写，采取类似的实现方法。如下图：

xxxActivity是离线模式下的方法，而xxxActivity2则是联机模式下的方法。采取类似的命名方式，更加便于我们在编写代码时寻找到对应位置。下面时对各个部分实现的详细介绍。

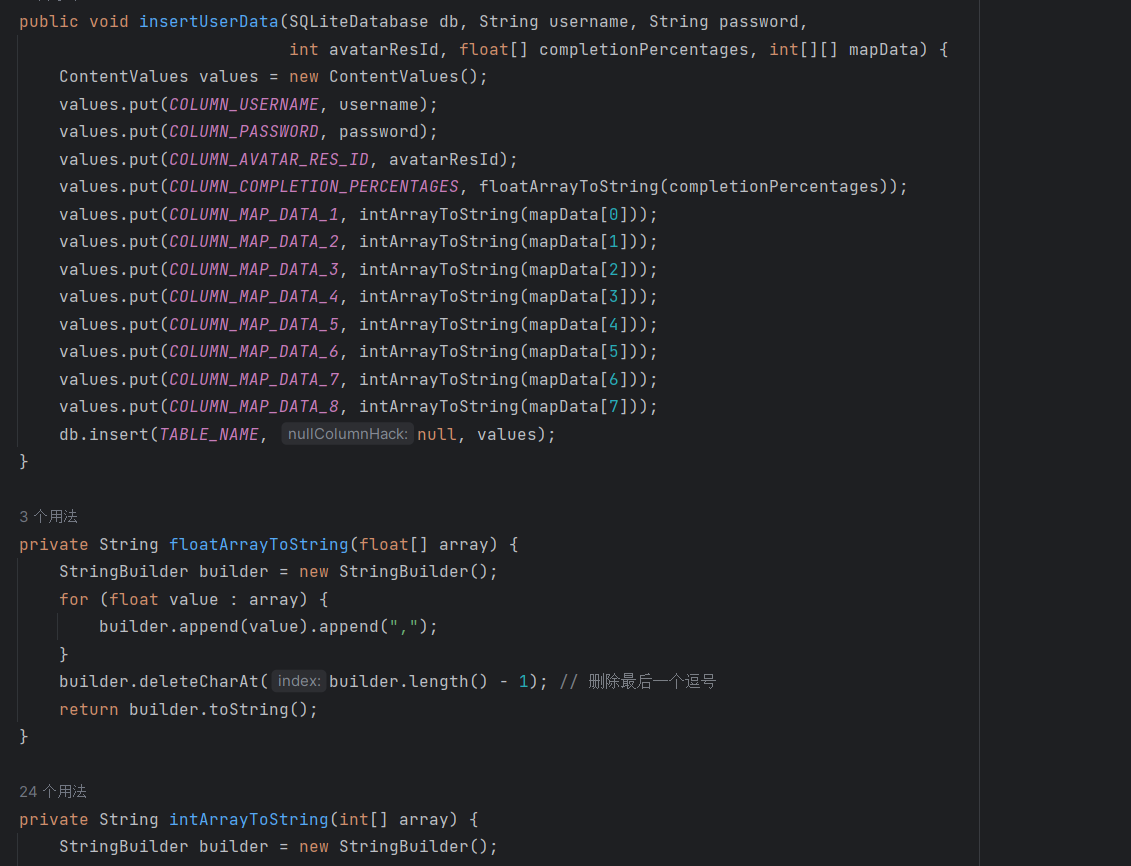
**（二）、离线模式。**

1.数据交互

采用SQLite存储离线数据。首先我们需要建立一个数据库表，根据游戏设计的需要我们创建了一个名为users的表，表中有用户名，密码，头像ID，八关游戏的完成率，和八关的地图，同时创好两个默认账号，并写好一些对数据库处理的一些函数以供之后数据交互时调用。

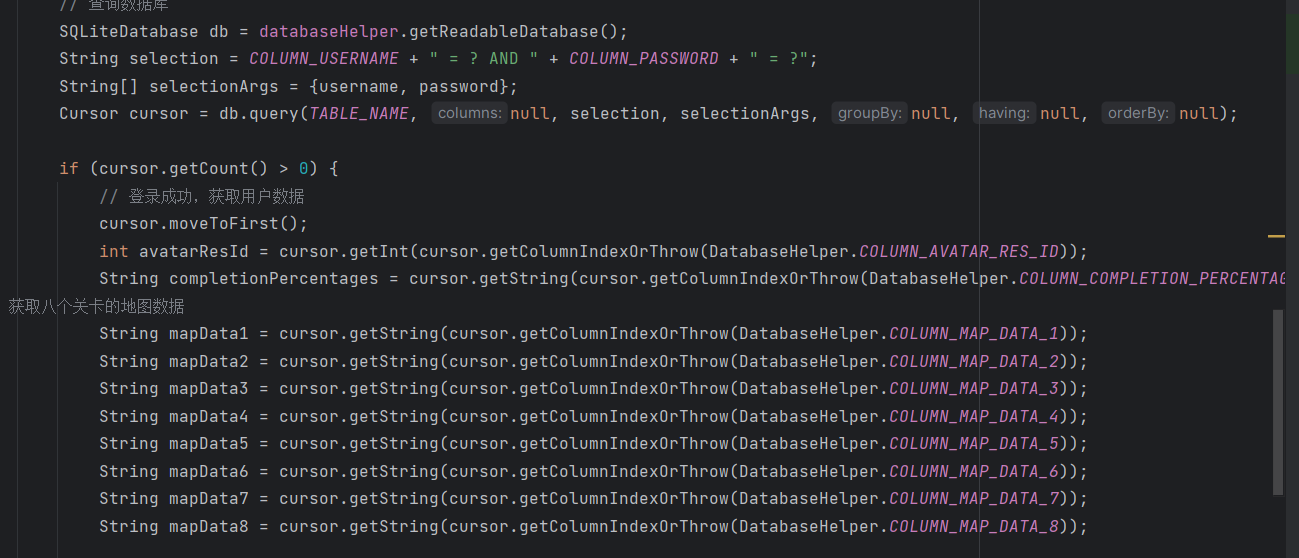


然后在数据交互的时候进行一些数据库指令：

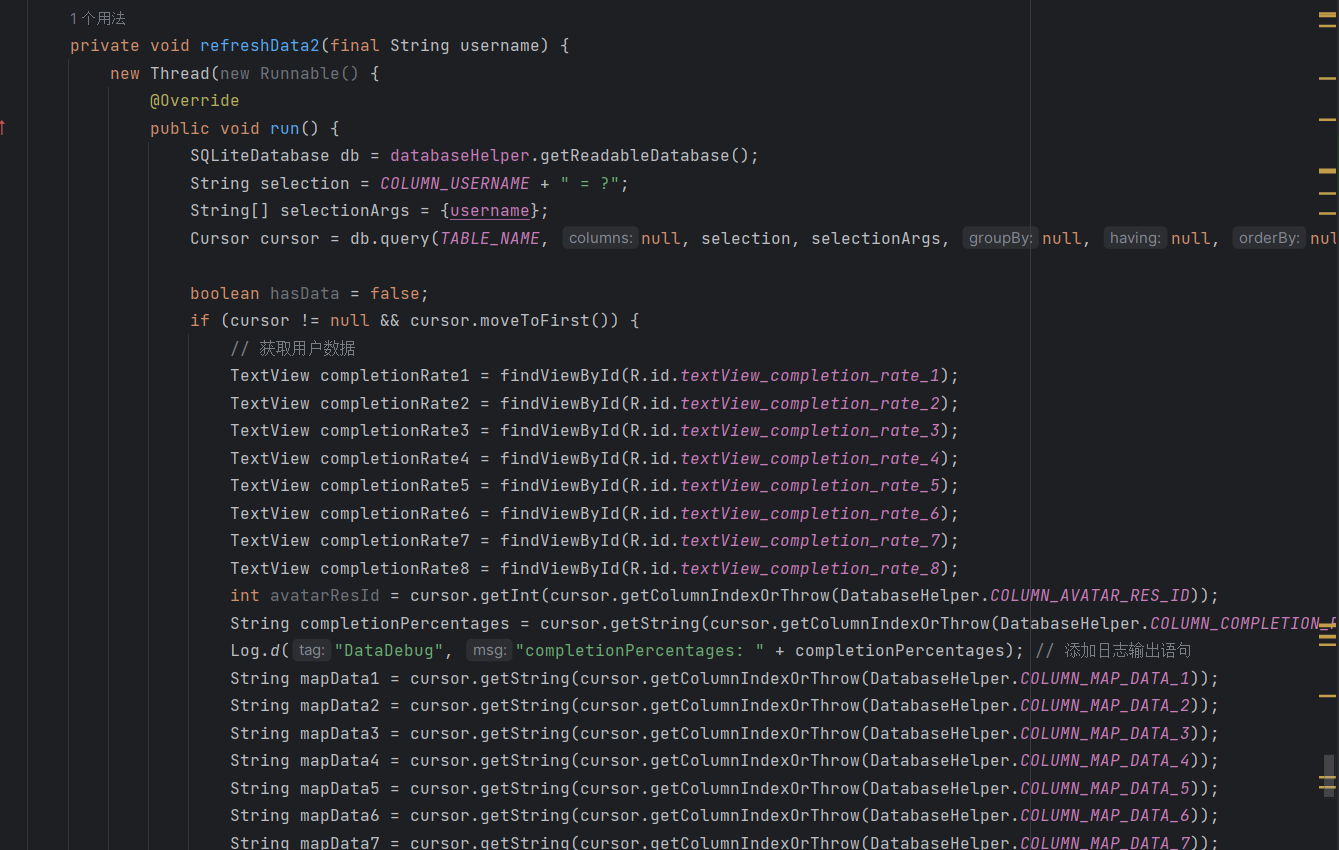
注册用户向数据库插入数据：



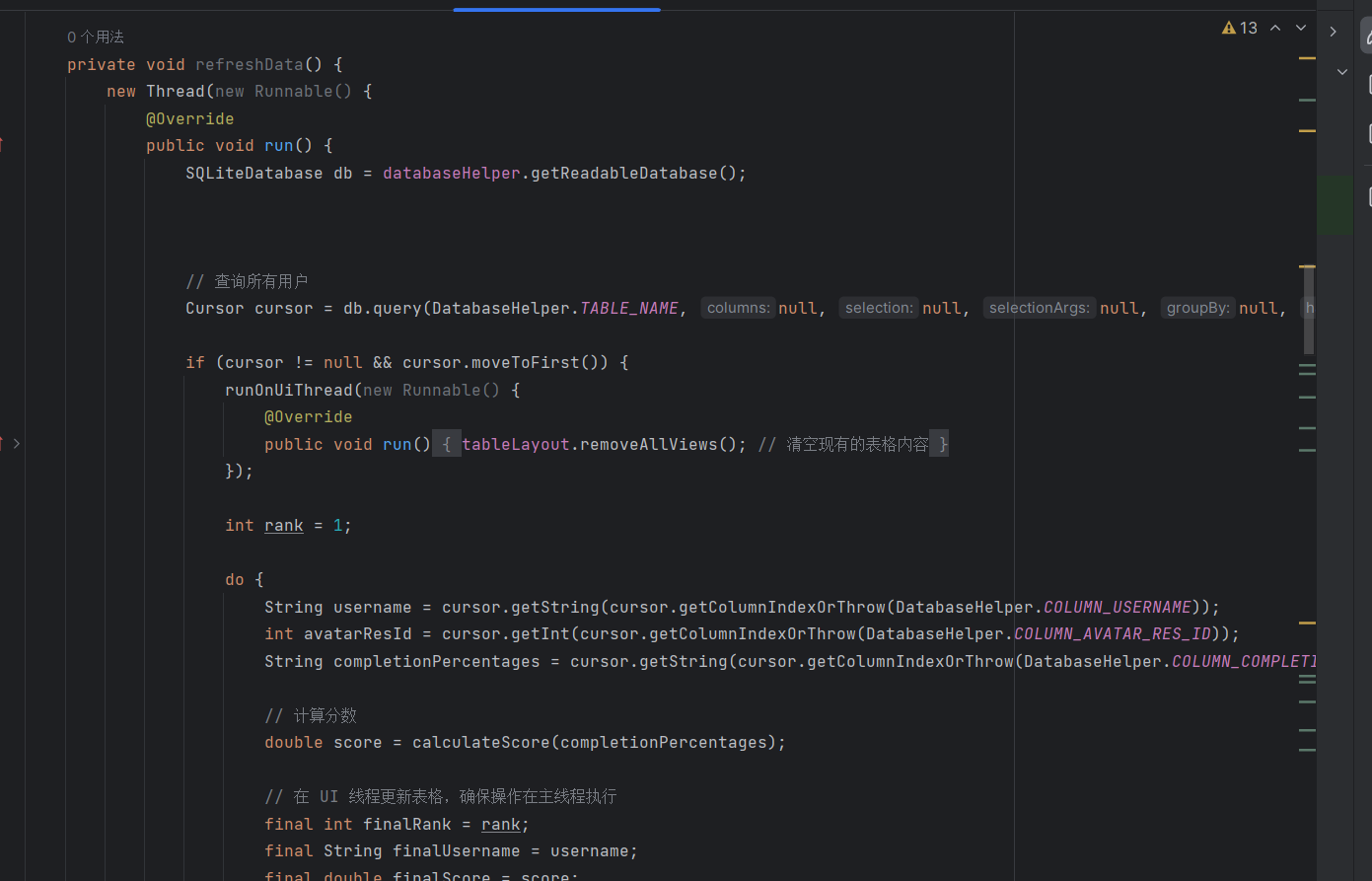
登录时查询数据库：



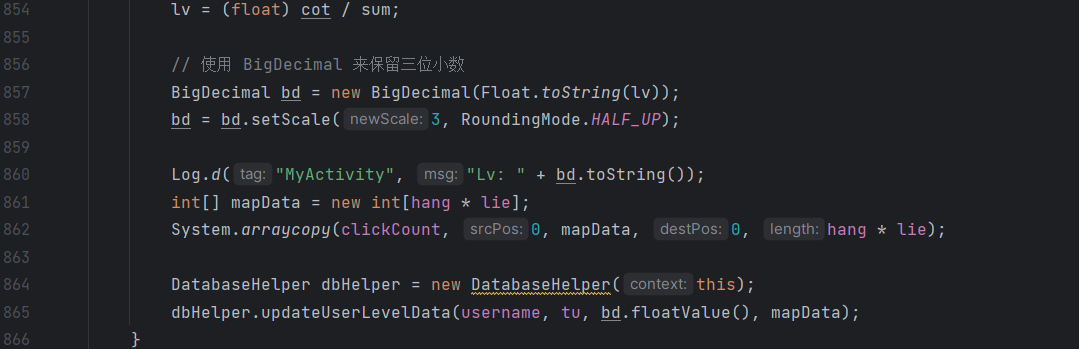
刷新游戏数据时查询数据库：



点击排行榜时查询数据库：



保存游戏数据时更新数据库：



1. 其他部分，如登录，注册，游戏界面等均与联机模式实现方式类似。

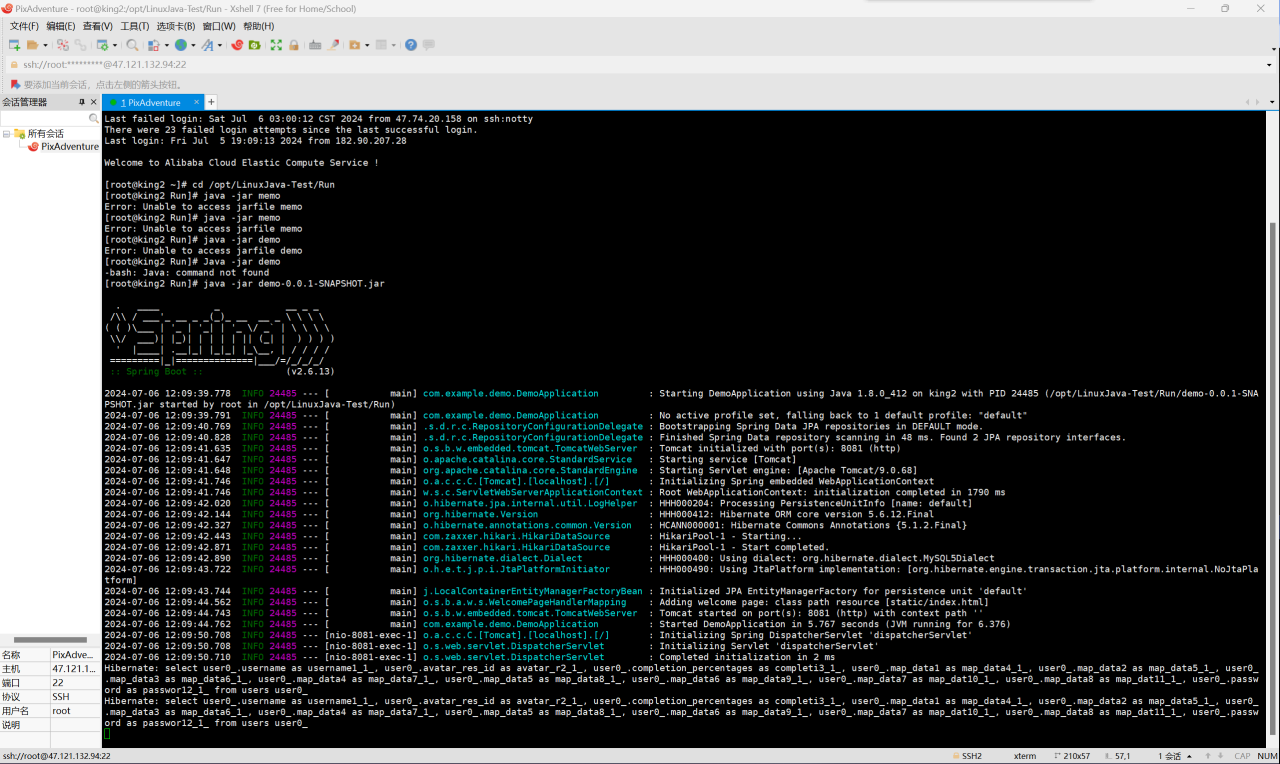
**（三）、联机模式**

1.数据交互

连接服务器的数据库，先在Linux上创建一个users表，表中有用户名，密码，头像ID，八关游戏的完成率，和八关的地图，然后通过使用idea编写springboot框架，编写对数据库的增删改查操作：

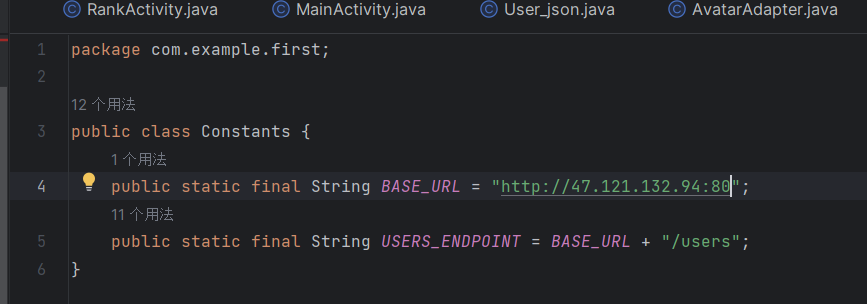
|  |
| --- |
| package com.example.demo.controller;  import com.example.demo.entity.User;  import com.example.demo.service.UserService;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.http.ResponseEntity;  import org.springframework.web.bind.annotation.\*;  import java.util.List;  @RestController  @RequestMapping("/users")  public class UserController {  @Autowired  private UserService userService;  @GetMapping  public List<User> getAllUsers() {  return userService.getAllUsers();  }  @GetMapping("/{username}")  public User getUserByUsername(@PathVariable String username) {  return userService.getUserByUsername(username);  }  @PostMapping  public User createUser(@RequestBody User user) {  return userService.saveUser(user);  }  @DeleteMapping("/{username}")  public void deleteUser(@PathVariable String username) {  userService.deleteUser(username);  }  @DeleteMapping("/deleteAll")  public void deleteAllUsers() {  userService.deleteAllUsers();  }  @PutMapping("/{username}")  public ResponseEntity<User> updateUser(@PathVariable String username, @RequestBody User updatedUser) {  User existingUser = userService.getUserByUsername(username);  if (existingUser != null) {  // 更新现有用户的数据  existingUser.setPassword(updatedUser.getPassword()); // 如果需要更新密码的话  existingUser.setAvatarResId(updatedUser.getAvatarResId());  existingUser.setCompletionPercentages(updatedUser.getCompletionPercentages());  existingUser.setMapData1(updatedUser.getMapData1());  existingUser.setMapData2(updatedUser.getMapData2());  existingUser.setMapData3(updatedUser.getMapData3());  existingUser.setMapData4(updatedUser.getMapData4());  existingUser.setMapData5(updatedUser.getMapData5());  existingUser.setMapData6(updatedUser.getMapData6());  existingUser.setMapData7(updatedUser.getMapData7());  existingUser.setMapData8(updatedUser.getMapData8());  // 保存更新后的用户数据  User updated = userService.saveUser(existingUser);  return ResponseEntity.ok(updated);  } else {  // 如果用户不存在，返回404 Not Found  return ResponseEntity.notFound().build();  }  }  } |

然后生成jar包，将jar包放到服务器上运行（服务器通过Xshell连接，我们使用的是阿里云服务器）：



然后在Android上点击相关按钮后对数据库进行相关操作。

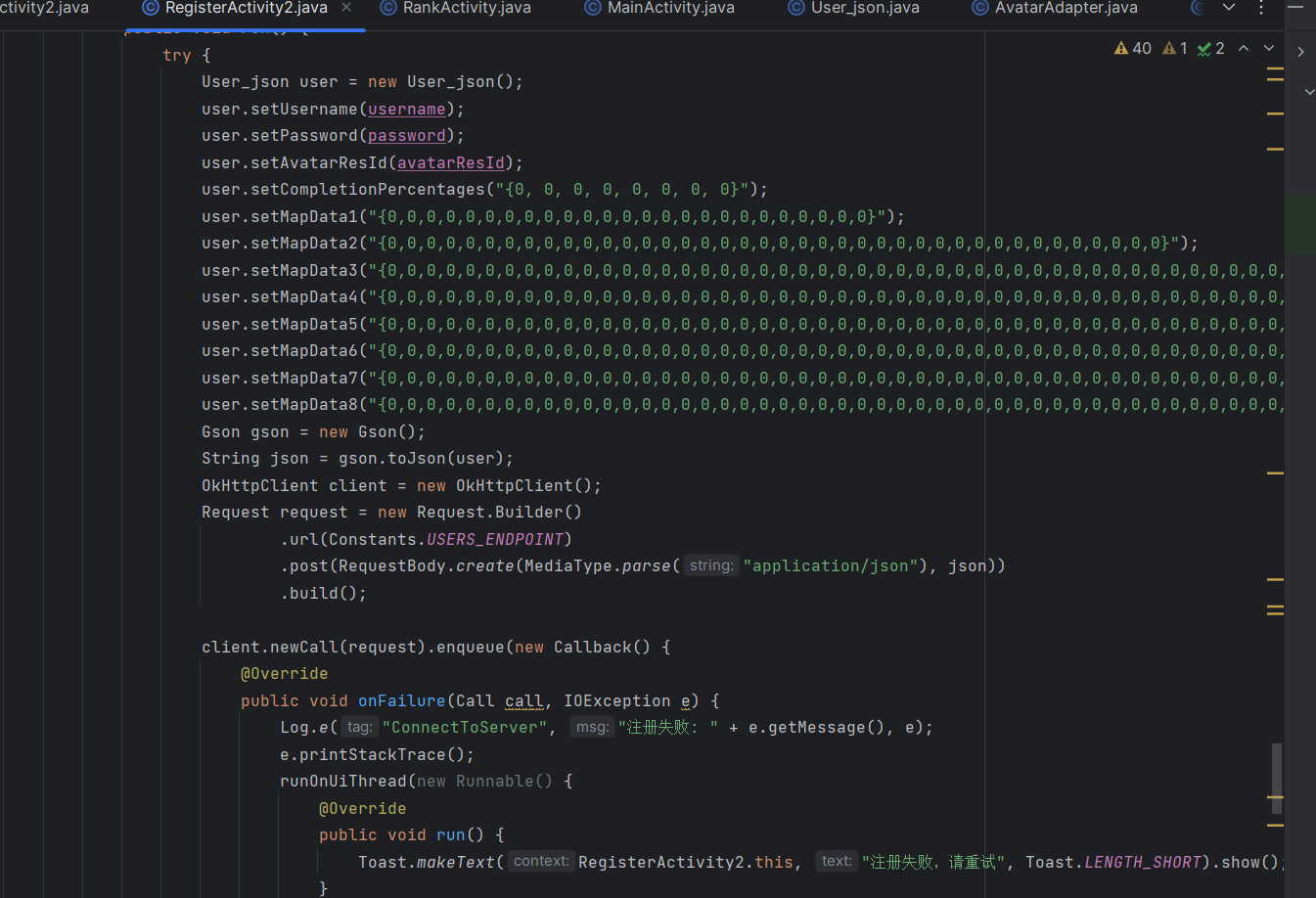
先创建一个类保存IP地址和端口号以便使用到的时候不用在输入url，也方便在测试的时候只用改一个地方，便于管理：



接下来是与离线模式类似的操作，只是更换了操作的数据库对象。

网络请求使用okhttp3的方法：

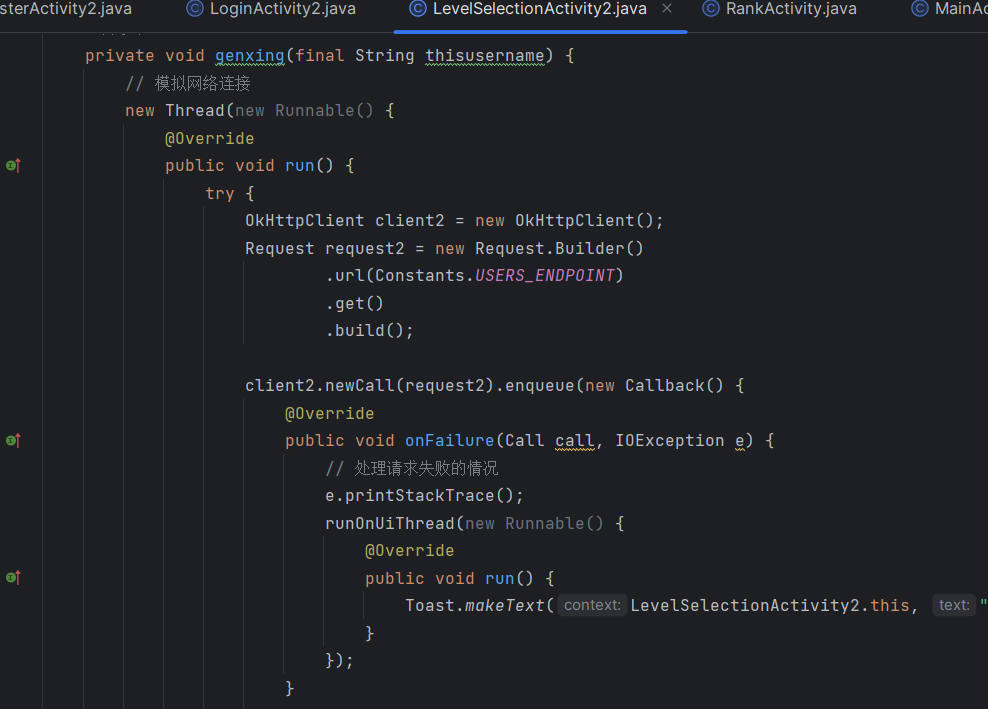
注册时向数据库插入数据



登录时查询数据库：



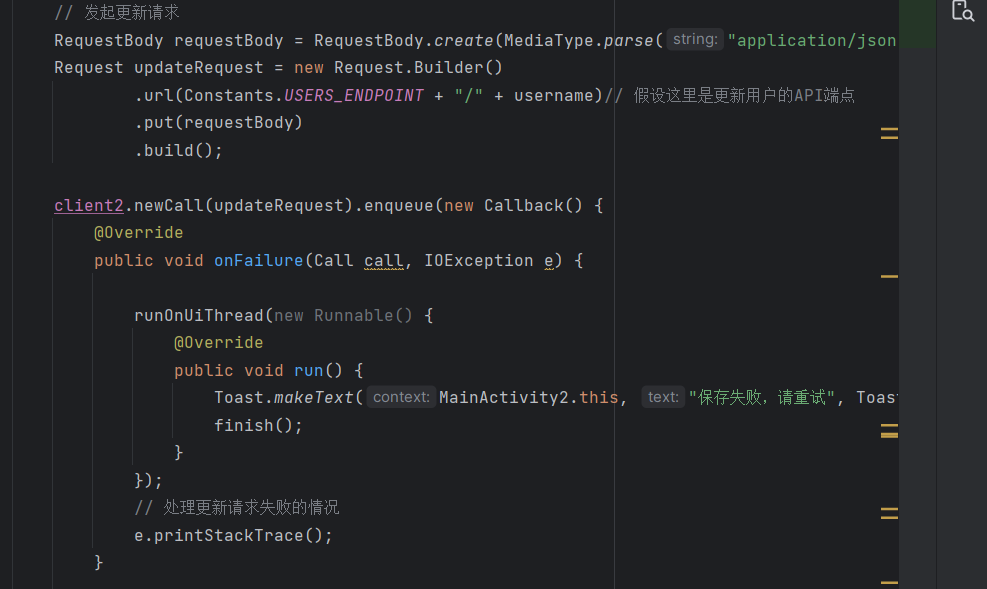
刷新时查询数据库：



点击排行榜查询数据库：



保存地图更新数据库：

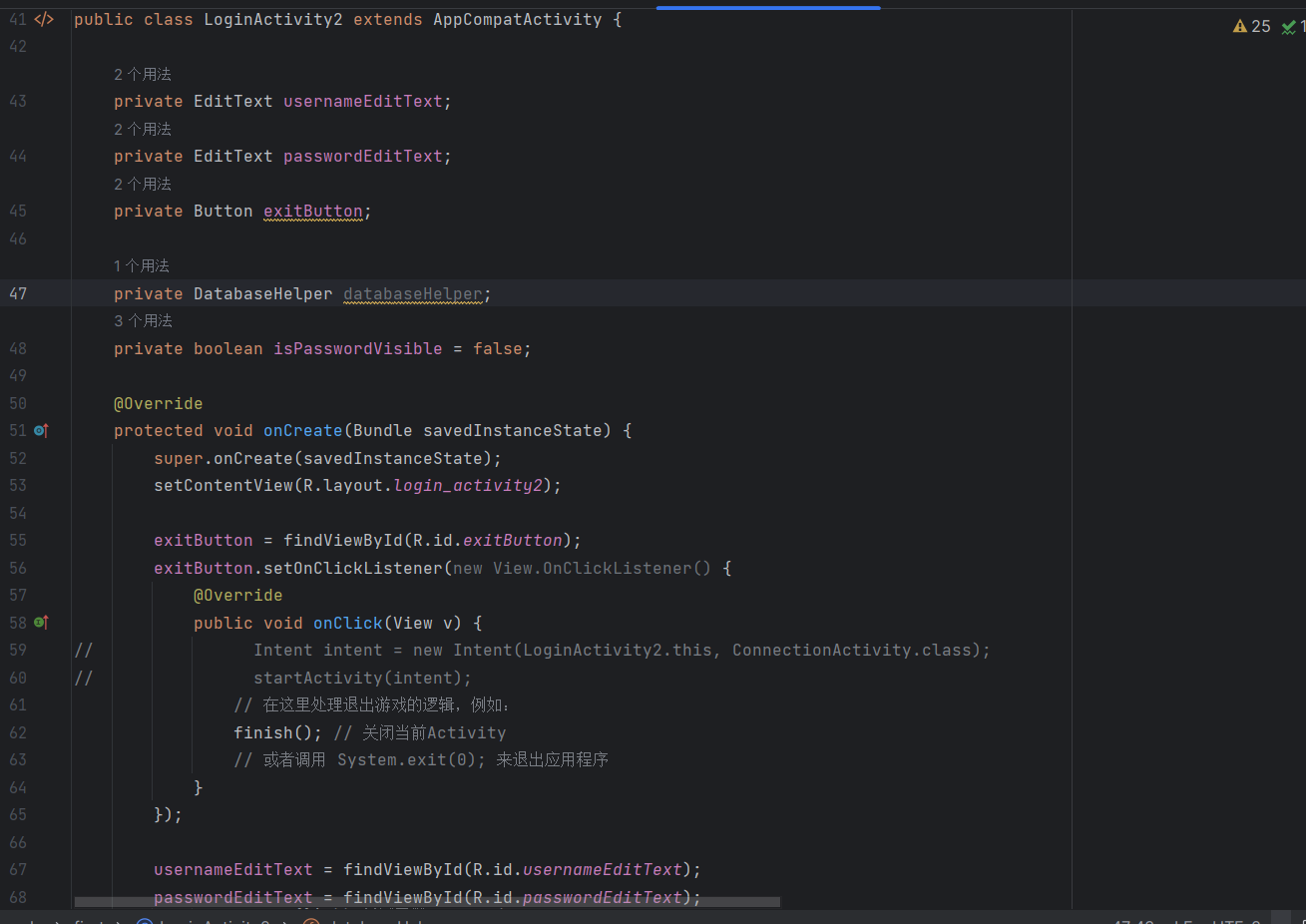


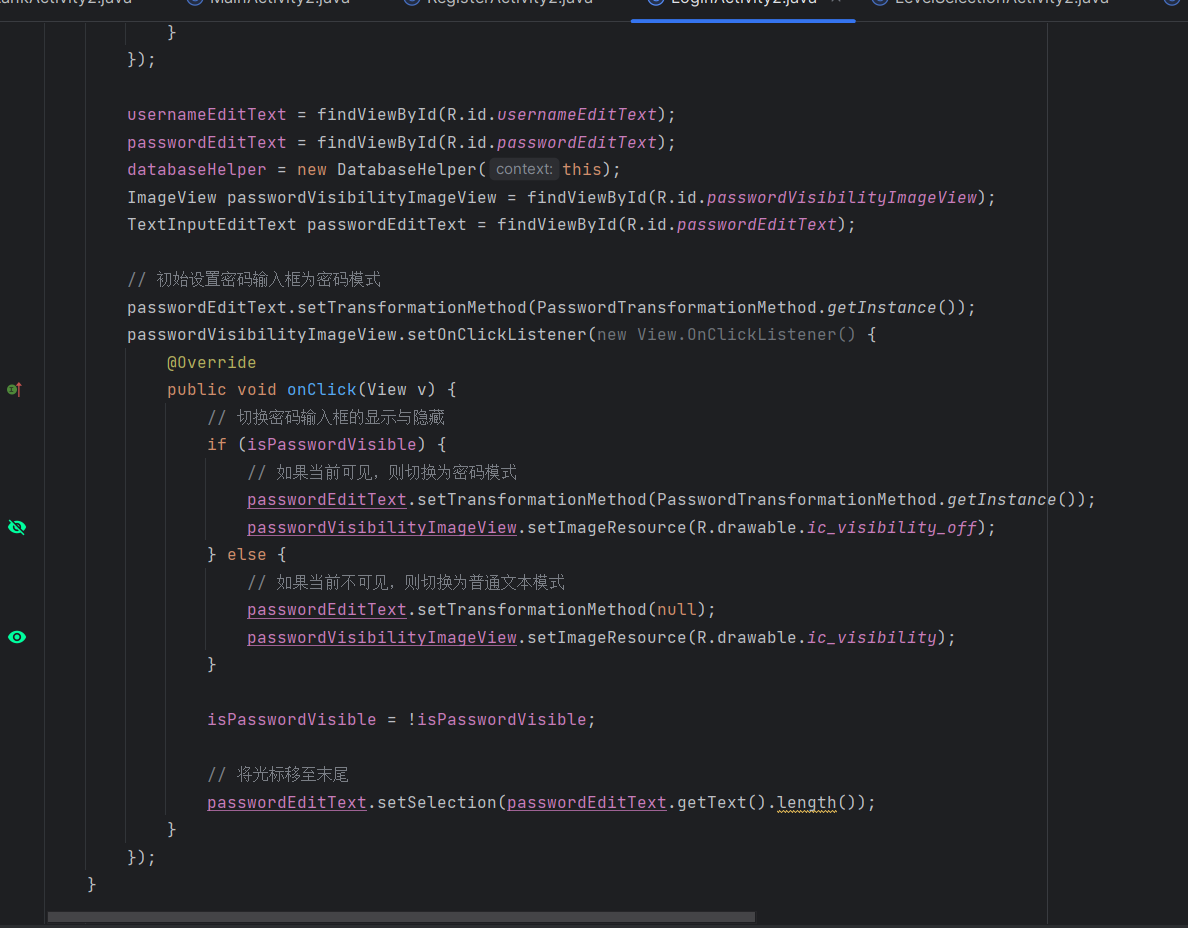
2.登录与注册

登录与注册采取类似的ui设计：

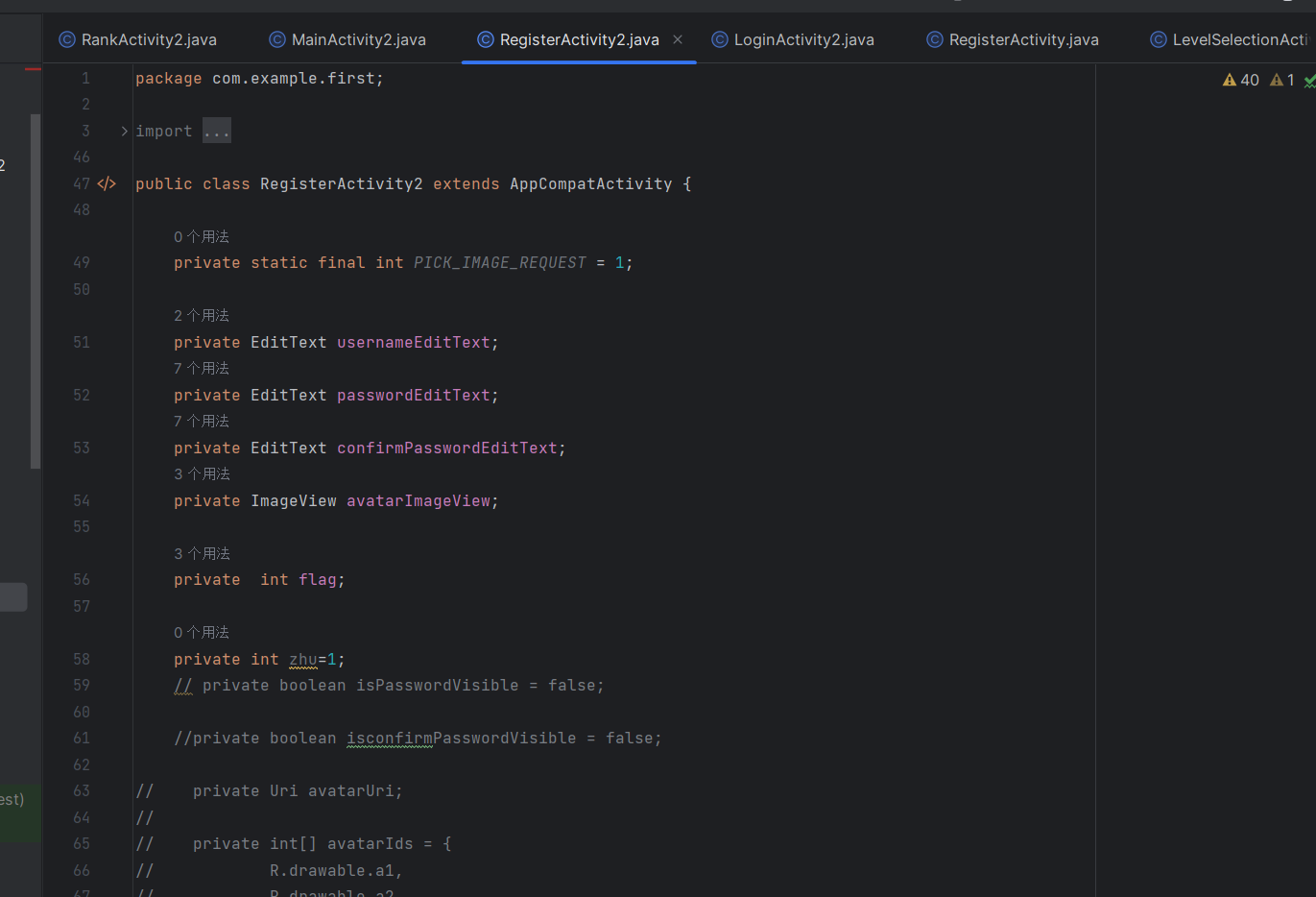
在登录界面下，输入数据库中已经存在的用户和对应的密码则可以进入到游戏的菜单界面，这里我们设计了人性化的提示，登录的时候会提示正在登录中，登录成功会有登录成功的提示，并且会为不同的登录失败情况进行不同的提示，如因为网络错误无法登录则提示“无法连接服务器”，如果是用户名密码没有输入则提示“请输入用户名和密码”，如果密码错误则提示“密码错误”。注册的时候也会有类似的提示，要求两次输入的密码一样，且注册的用户名不能在数据库中已存在。密码的输入类似其他手机软件，可以点按右边小标志来选择隐藏或查看密码，提高安全性。

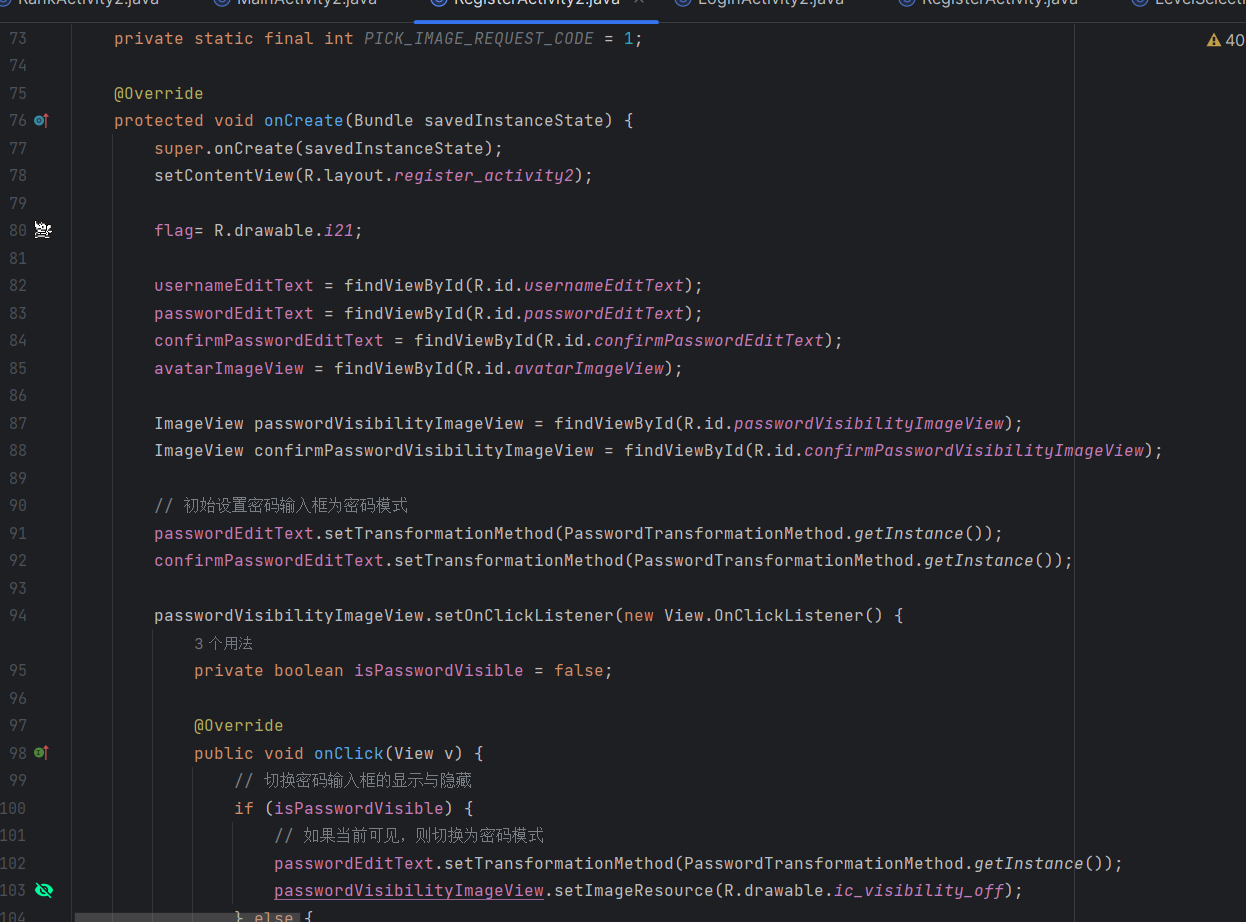
登录部分代码：



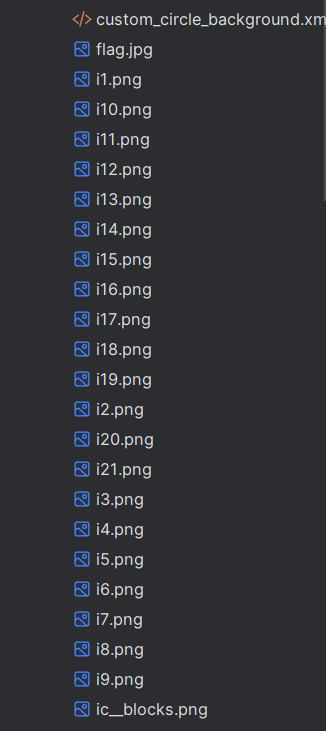


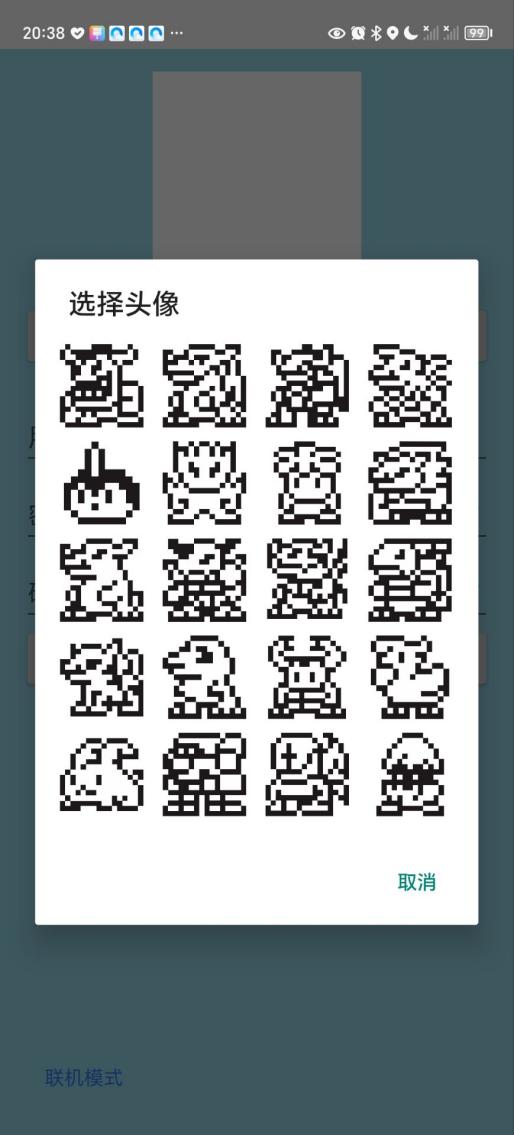
注册部分代码：



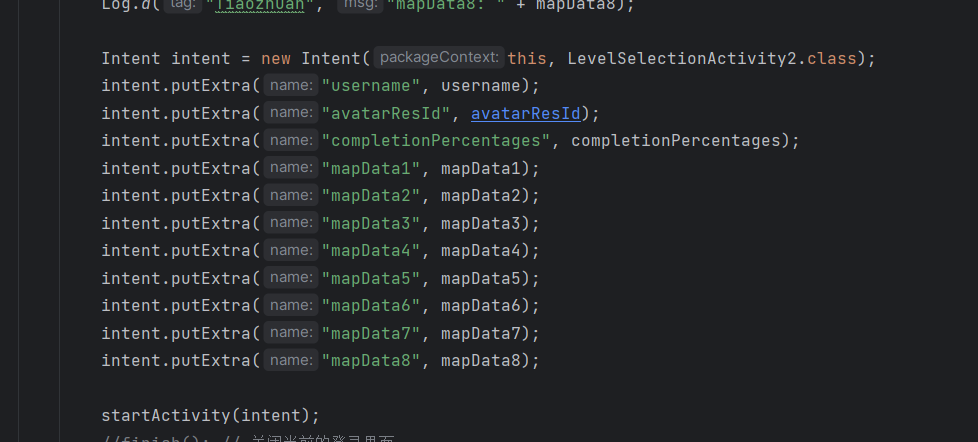


其中注册使用的头像是提前存在目录drawable下的:



UI效果：

登录成功之后要将对应用户的数据传到游戏菜单界面：



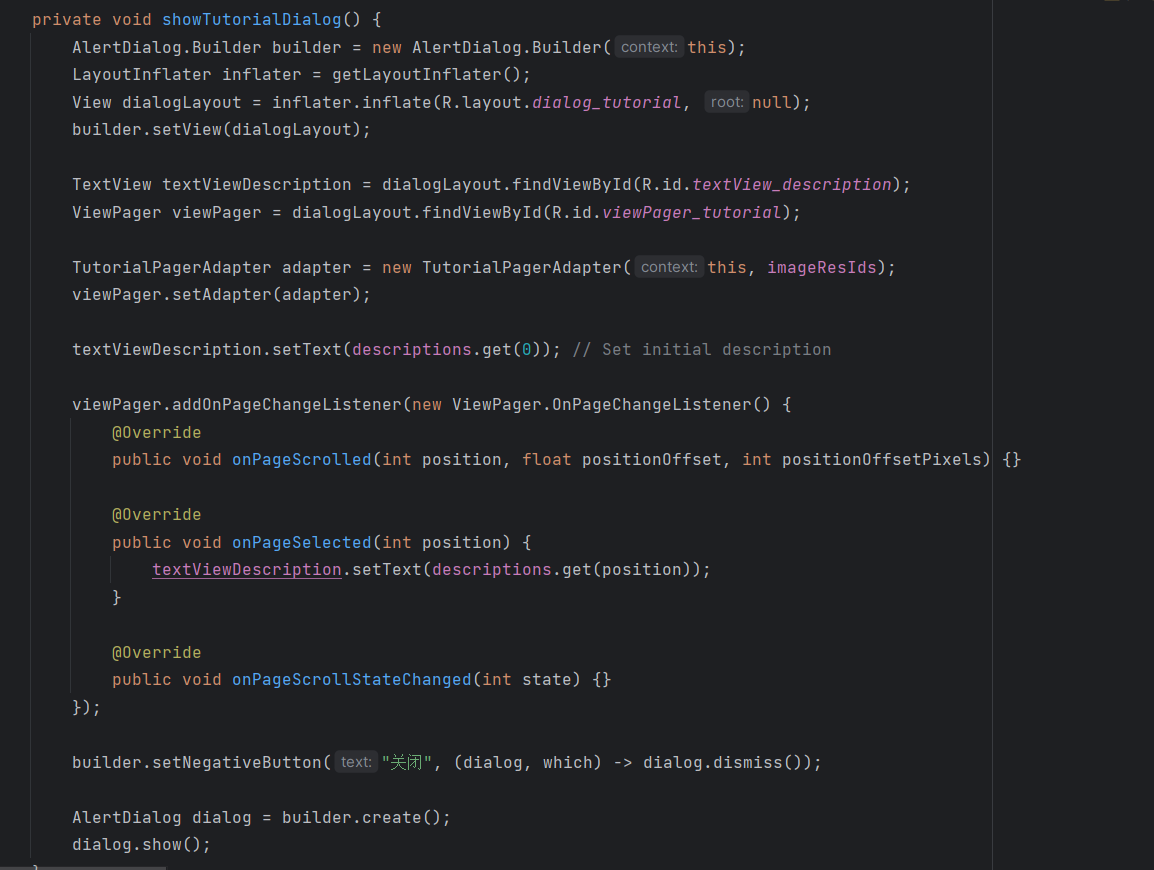
3.游戏主菜单

UI展示：

如图所示，主菜单的左上角是用户头像和姓名，类似QQ等手机应用，这个头像点击后会呼出侧拉框，侧拉框的功能与实现会在下文详细介绍。

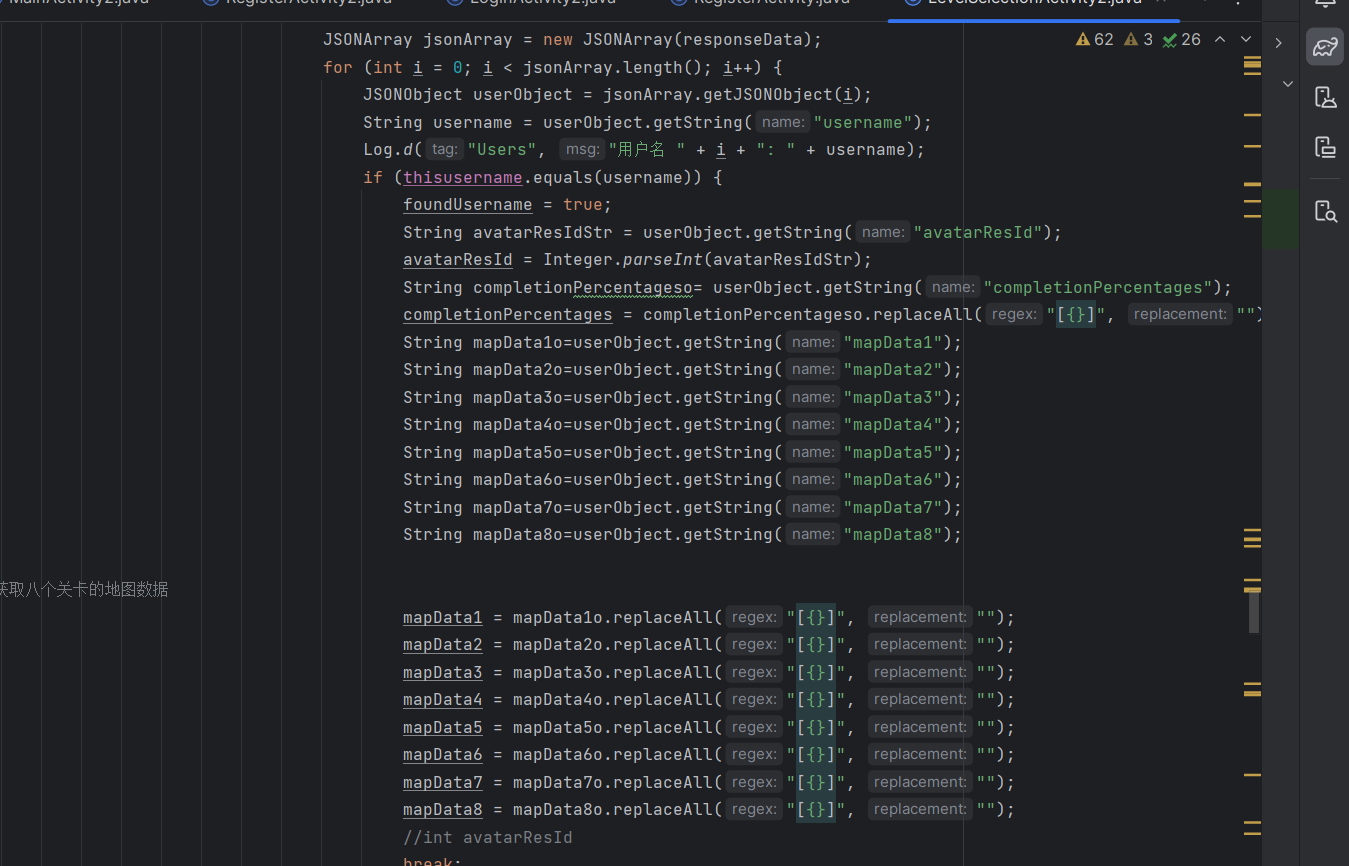
主菜单的中上方有一个新手教程按钮，点击后会弹出对话框：

如图所示，点击“新手教程”后会弹出一个对话框，配有文字和说明显示信息。该提示框可以按住左右滑动切换上一张下一张，新玩家在阅读之后能很快上手游戏玩法。

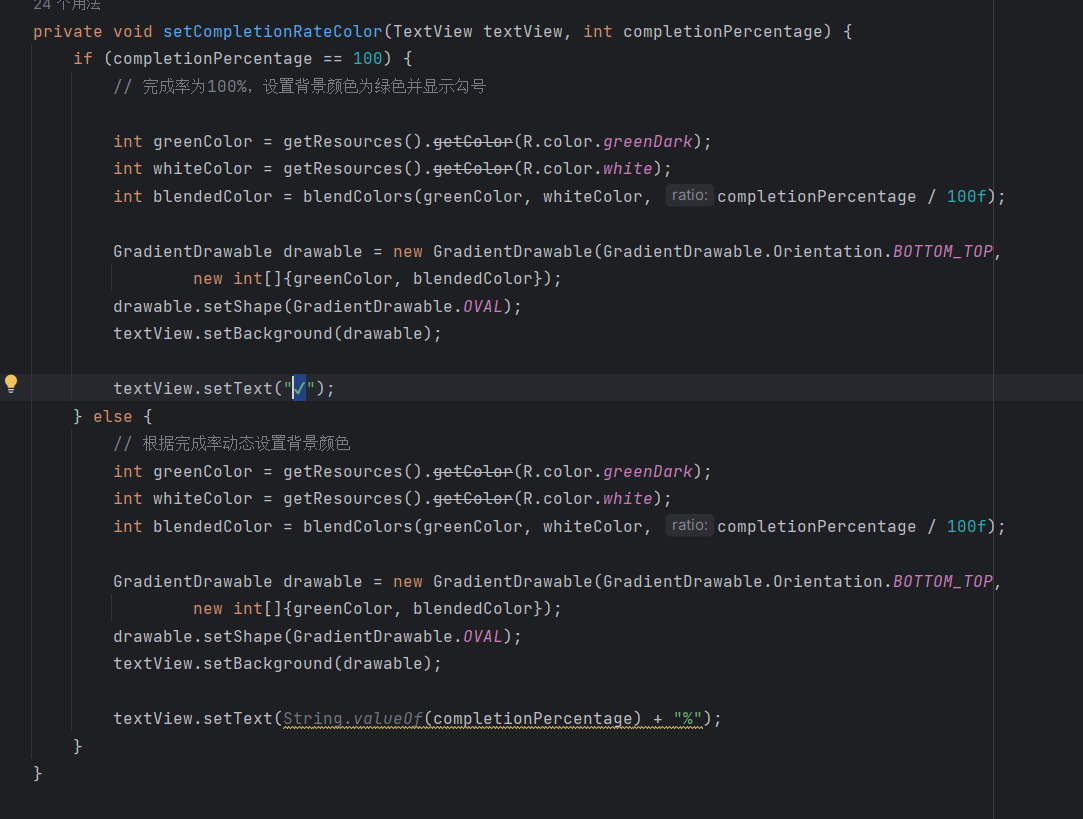
实现代码：

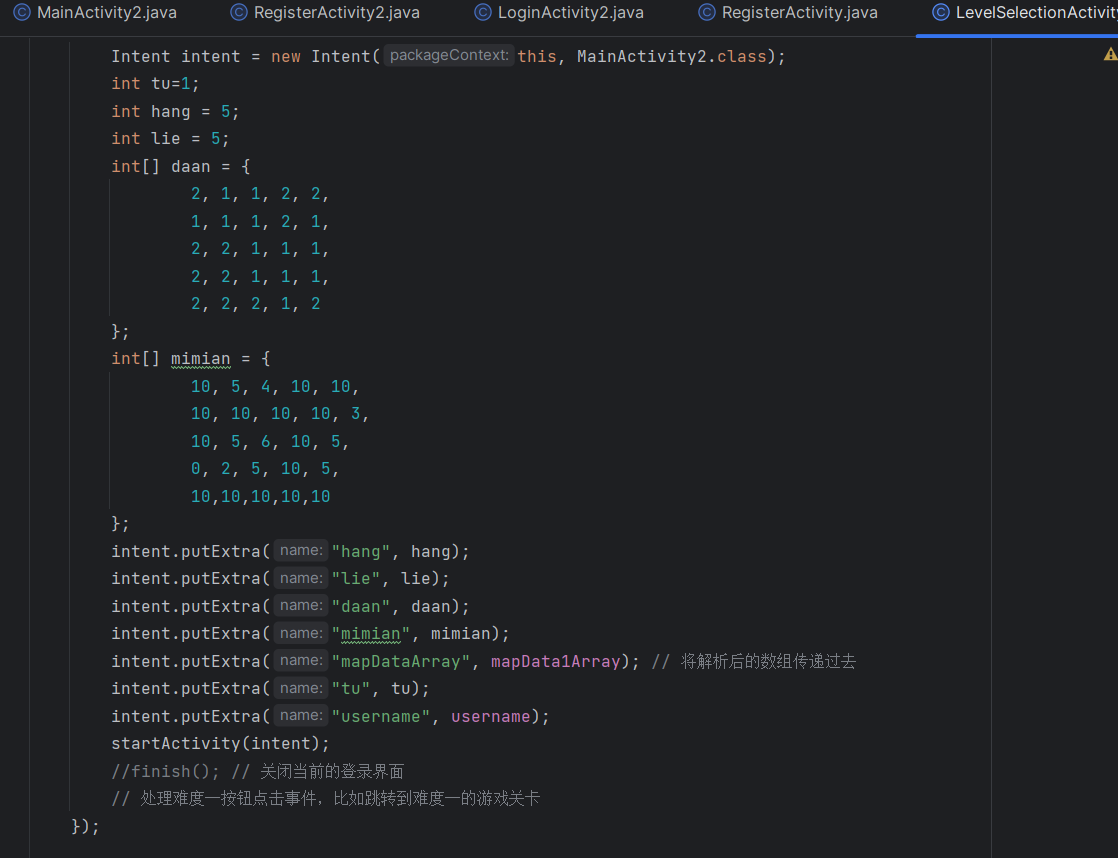
主菜单的右上方是一个音乐的播放暂停开关和一个刷新按钮，进入主菜单时应用会自动播放背景音乐。其实现方法主要是创建mediaplayer。代码较为简单，这里就不详细介绍了。

背景音乐开关旁边是刷新按钮，点击刷新按钮后会重新读取当前用户的相关数据并刷新界面展示最新数据。

实现代码：

再下面的八行按键表示八关游戏的进入按钮，以及对应关卡的完成程度。点按关卡的进入按钮后进入正式游戏，完成程度示数会根据每个关卡的不同，从数据库里查询对应的数据读出，再依据其值赋值给对应的百分比图标，图标通过drawable的xml代码，设计出了渐变的效果。完成度为百分之百时则会展示✔，美化游戏的同时也提升了玩家游戏正反馈。

实现代码：  


点击关卡按钮后会进入主要游戏界面，同时传入对应的数据到游戏界面：

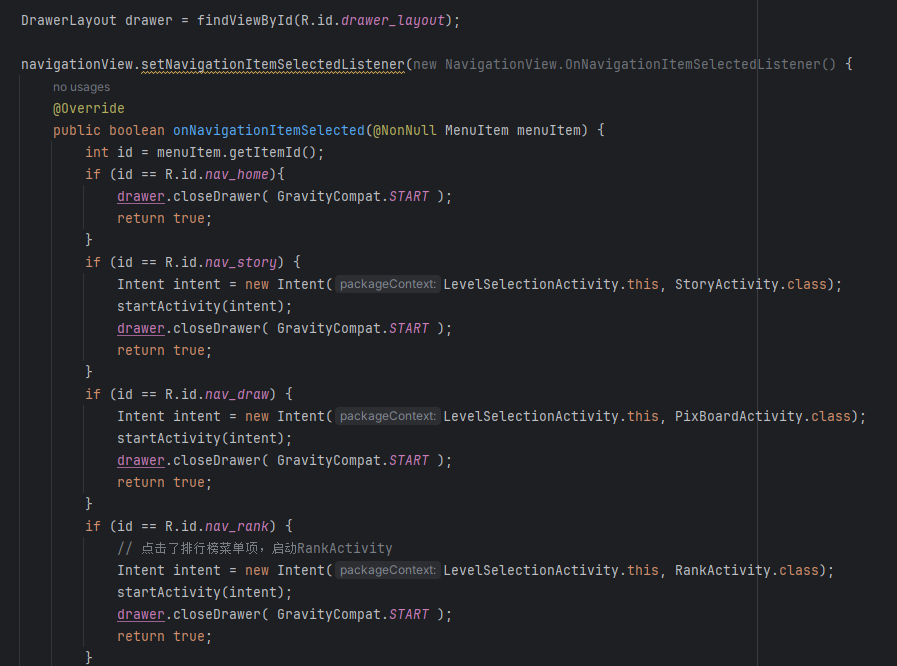
4.侧拉框

UI效果：

为了使我们的安卓应用集成更多功能，以及为日后开发预留空间，我们参照了QQ等日常生活中常用的手机应用，在我们的项目中实现出了一个可以在主菜单呼出的drawer侧边栏。点击左上方的头像即可呼出，符合大多数用户的直觉，同时设计了提示引导用户点击头像。呼出侧边栏后，会展示我们应用的更多功能，用户点按对应功能的选项即可进入该功能的页面。

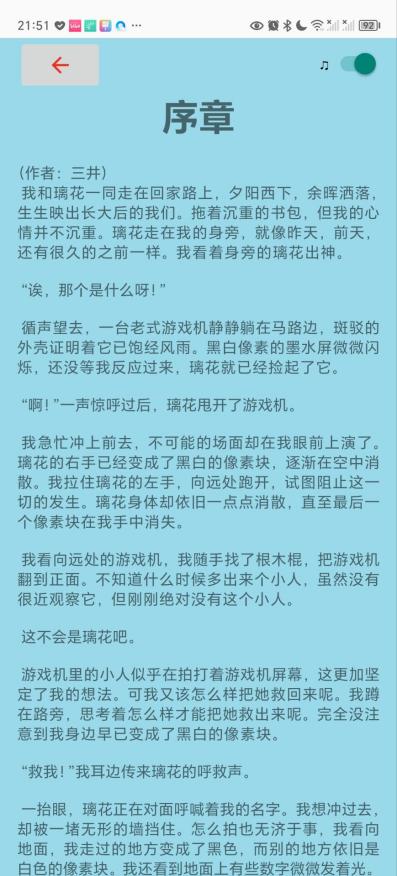
在侧边栏的设计上，我们编写了很多drawable文件来实现诸如渐变色背景、各个选项的图标等。代码繁多例如pathData和shape等，均以视觉设计为主，在此不再赘述。

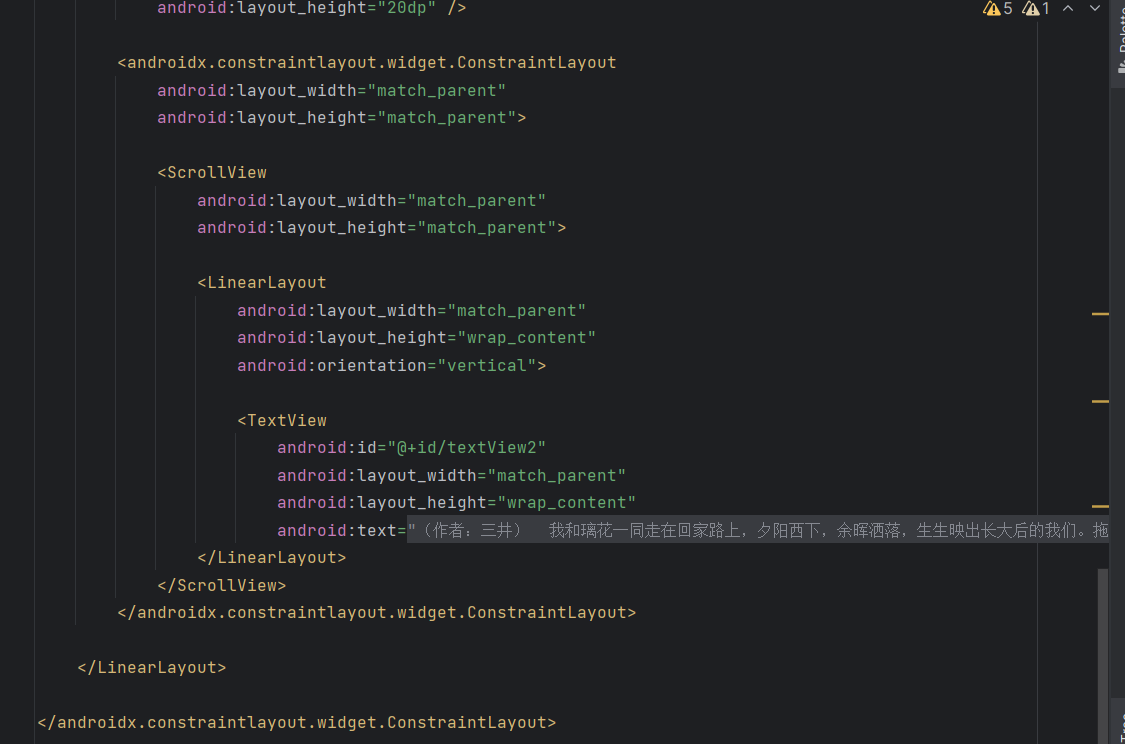
侧边栏的功能实现代码主要如图：



点击侧边栏的关卡选择后会回到游戏主菜单，即关闭侧边栏进行关卡选择。

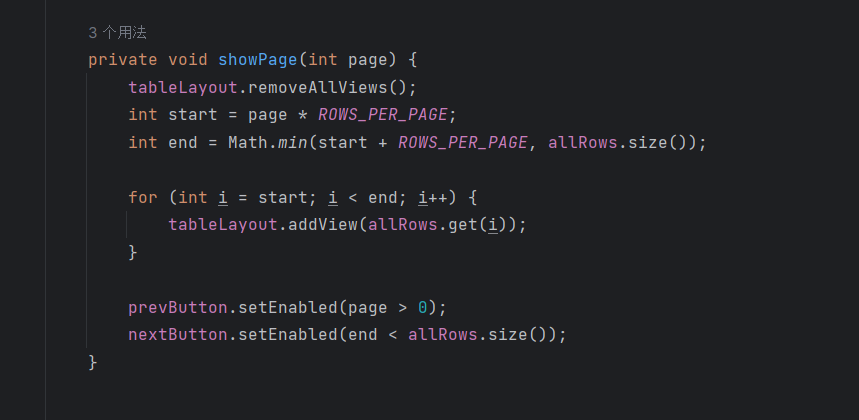
点击剧情后会跳到剧情界面，这里有我们设定的游戏剧情背景文章以供玩家阅读。文本保存在res/values/strings.xml中，由于文本较长，所以我们使用了ScrollView来实现上下滑动阅读，同时该页面的背景音乐与主菜单不同，同样的可以点按右上角音乐开关来自由选择播放与否。

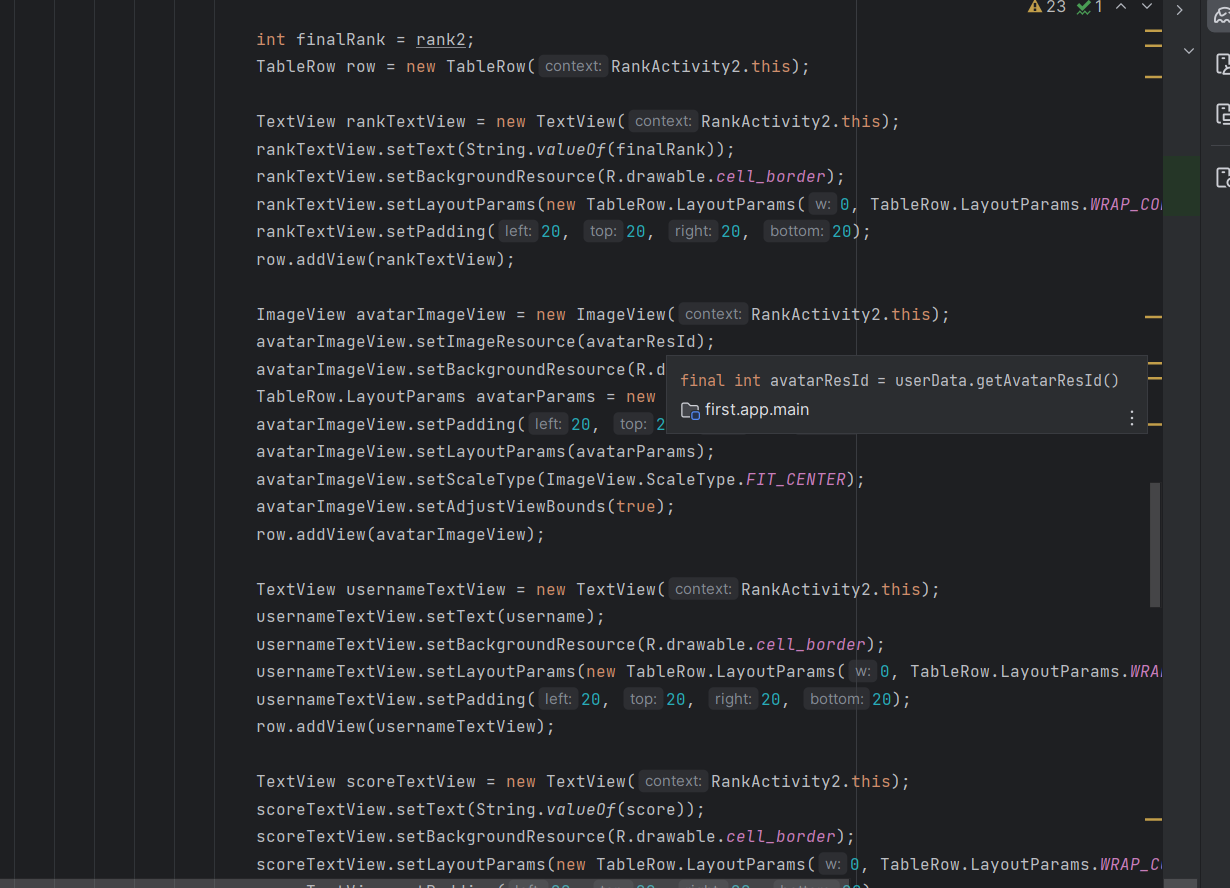
UI界面如下图：

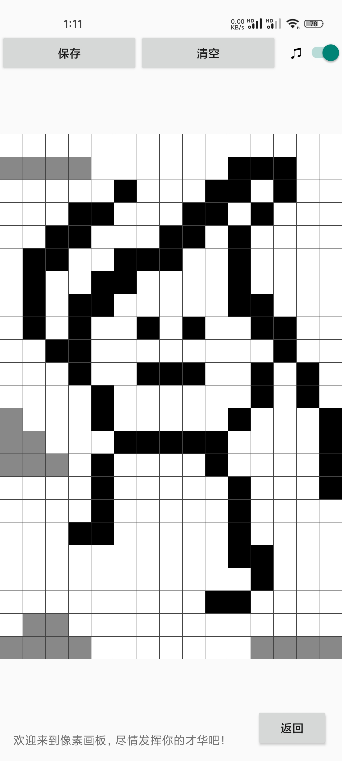
实现代码：

点击排行榜后会进入排行榜界面，这里会根据游戏分数排名由高到低排列所有用户，动态生成每一行TableView：

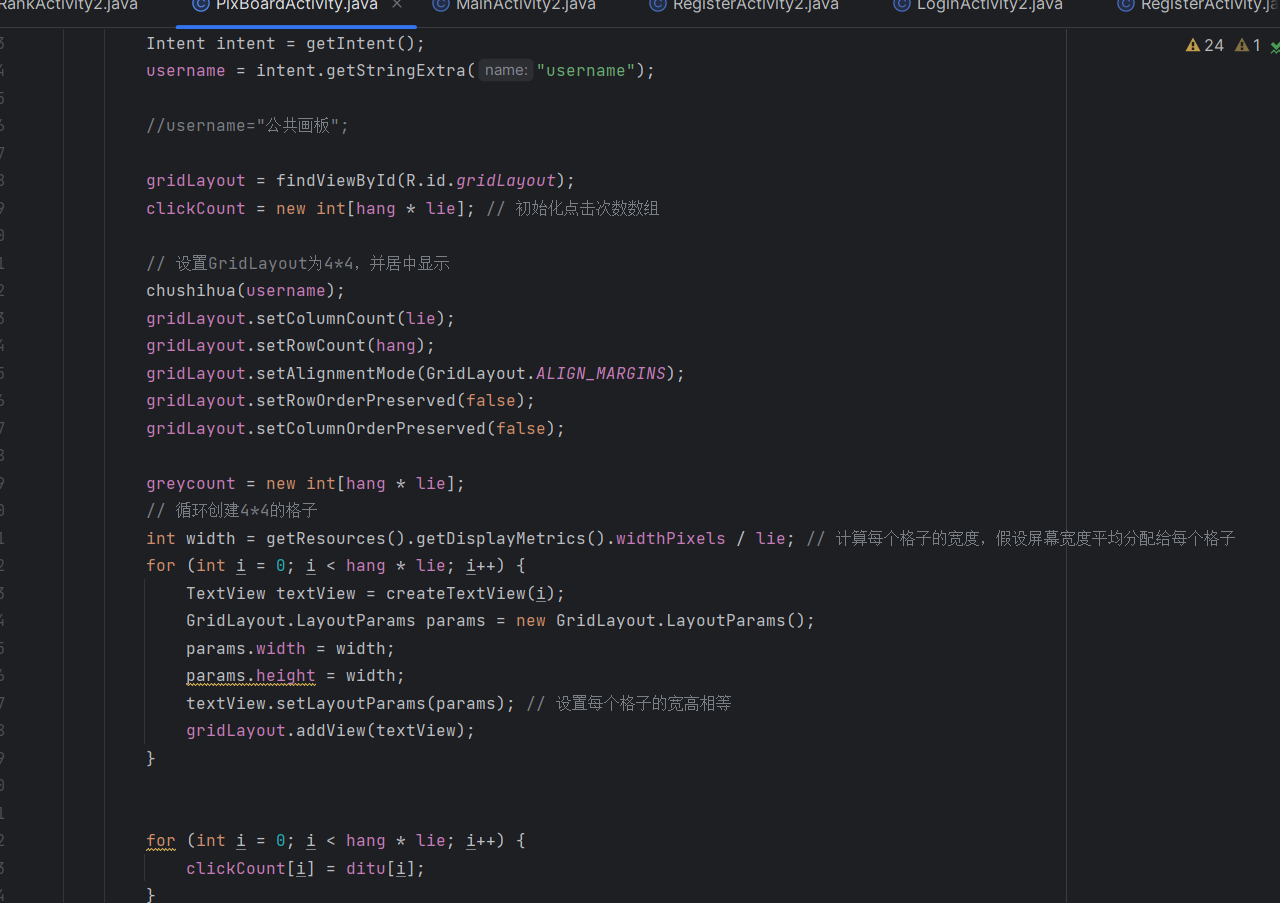
这里实现主要是通过查询数据库，根据得分进行排名，当显示用户超过20个时会自动新建页面，并且可以点按按钮进行翻页操作。

实现代码如下：

点击每周关卡后会弹出一个弹窗，这个功能我们正在开发中。侧边栏的优势正体现在拥有足够的功能拓展空间，日后我们开发者增添修改这些功能时十分便捷。

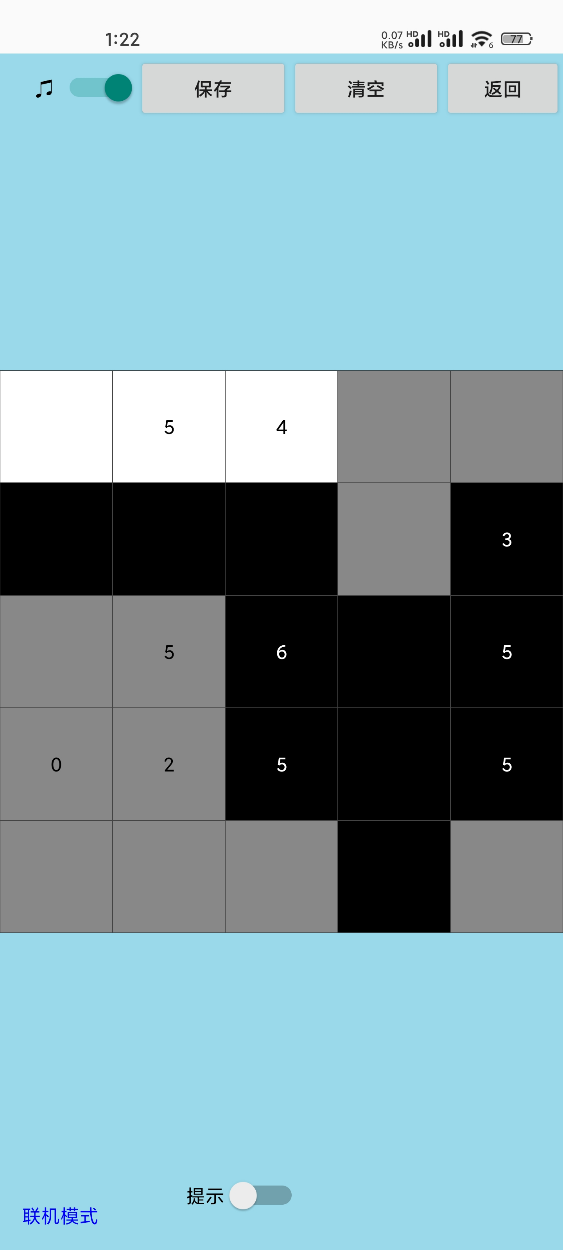
点击像素画板后，应用会跳转到自由画板界面，点按操作与实际游戏时一致，但是没有规则限制，玩家可以自由涂画作品，像素画板保存后的数据会保存到本地。

实现代码：

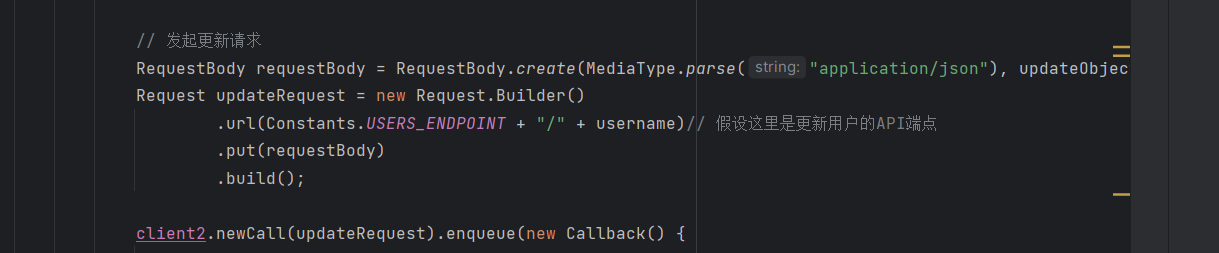


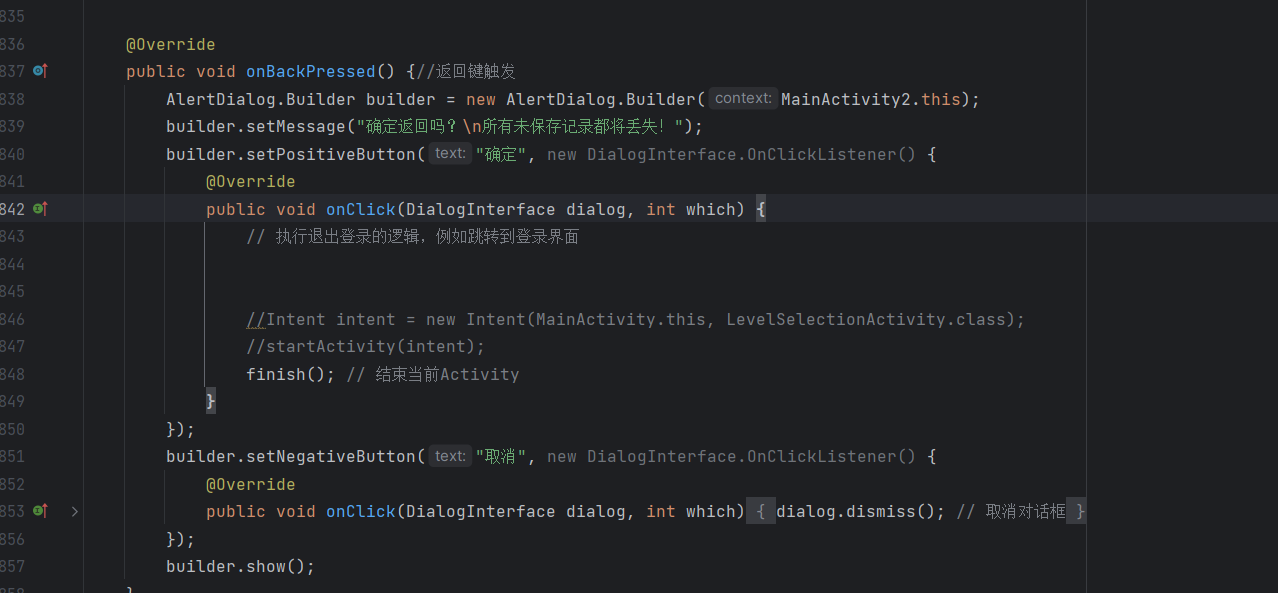
点击关于我们，会进入开发者的介绍：

5.游戏界面

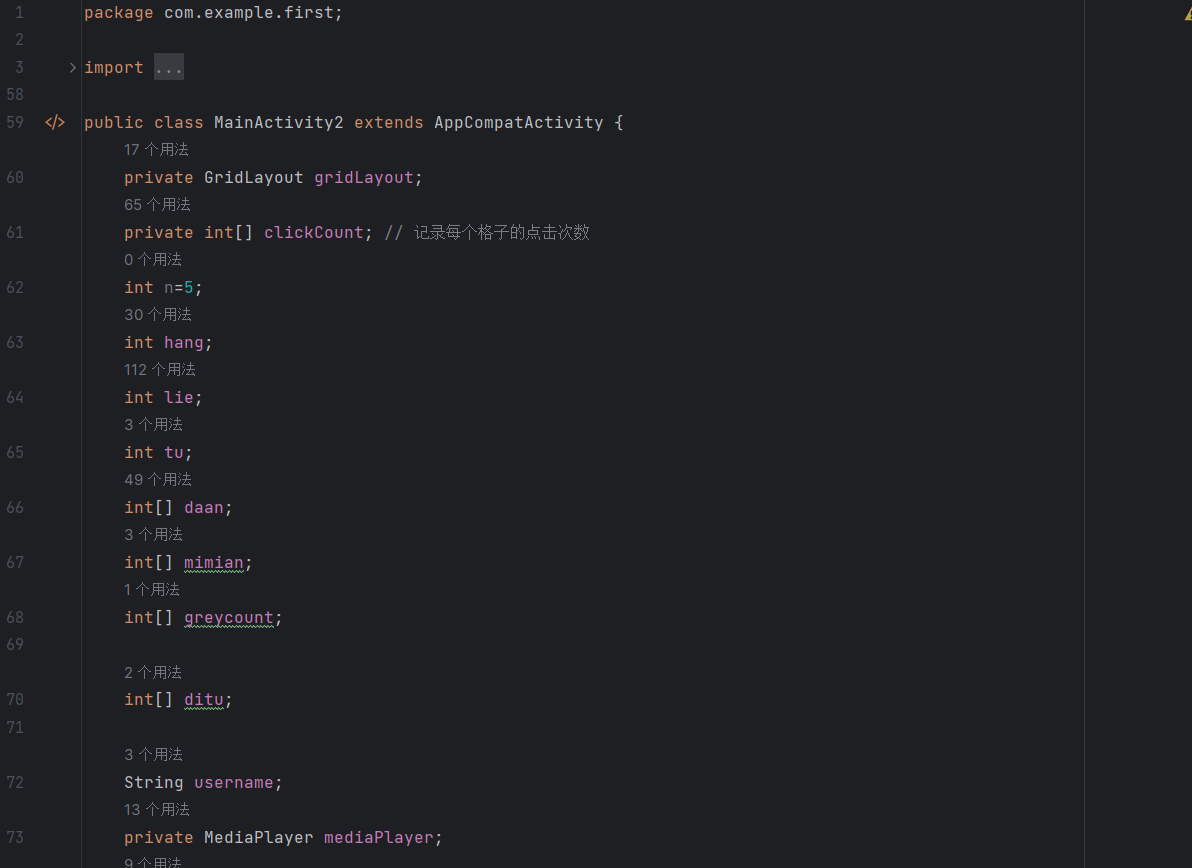
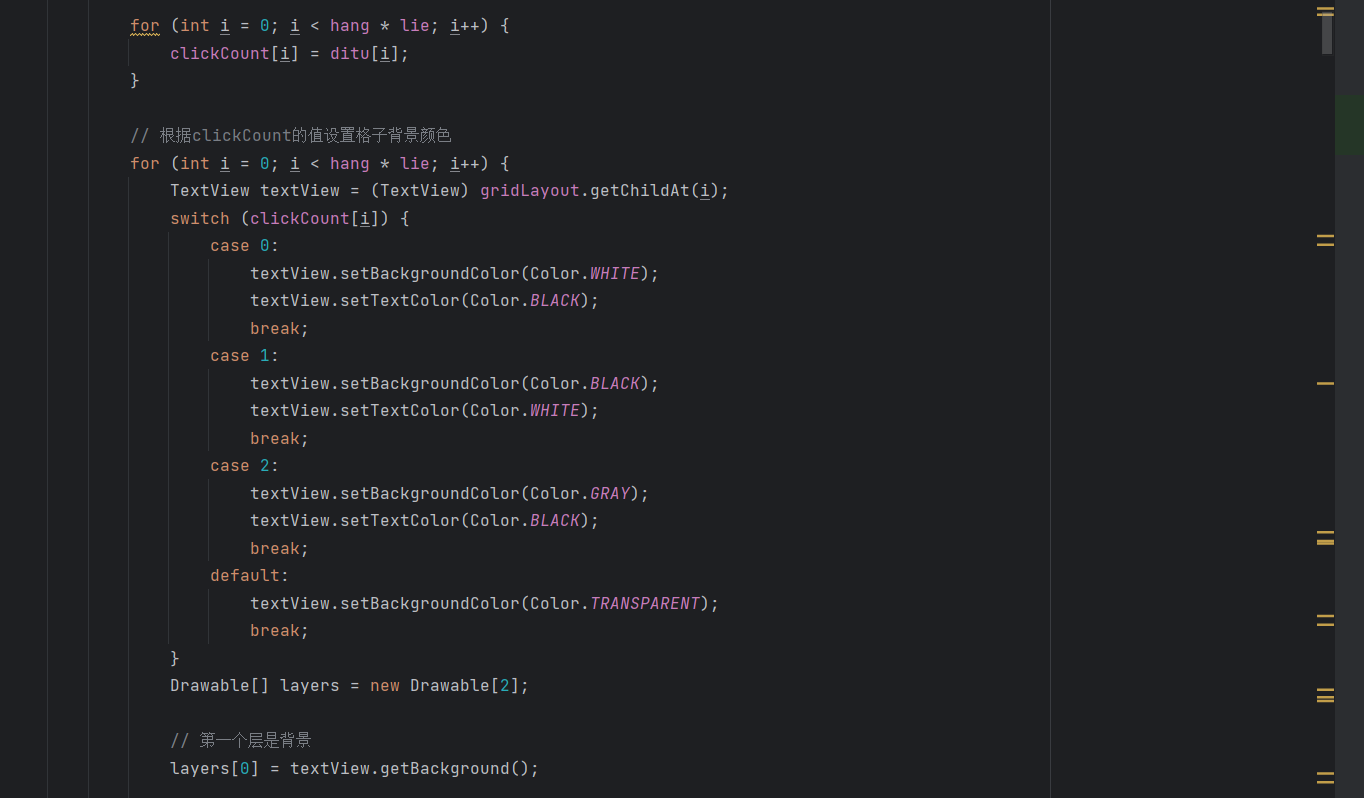
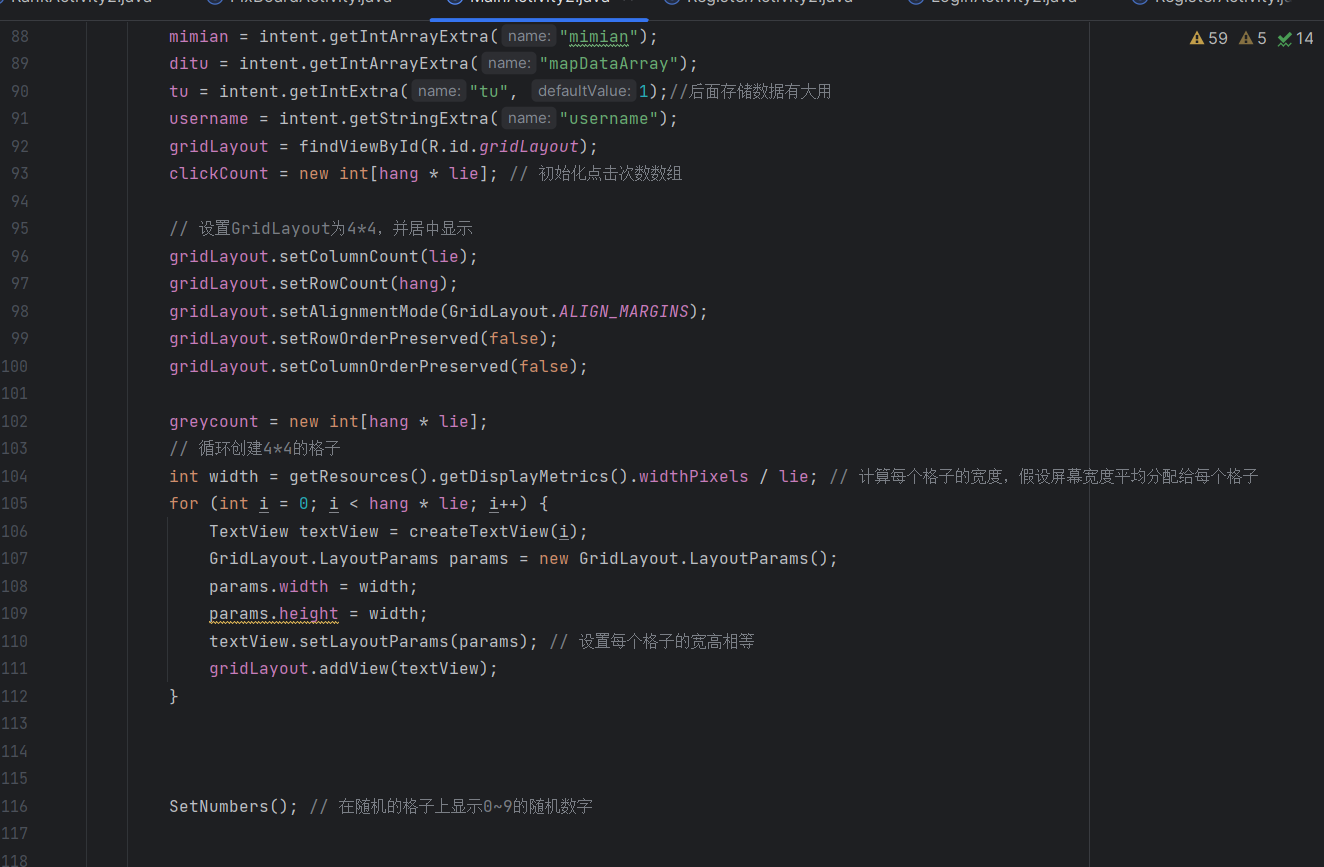
UI效果：

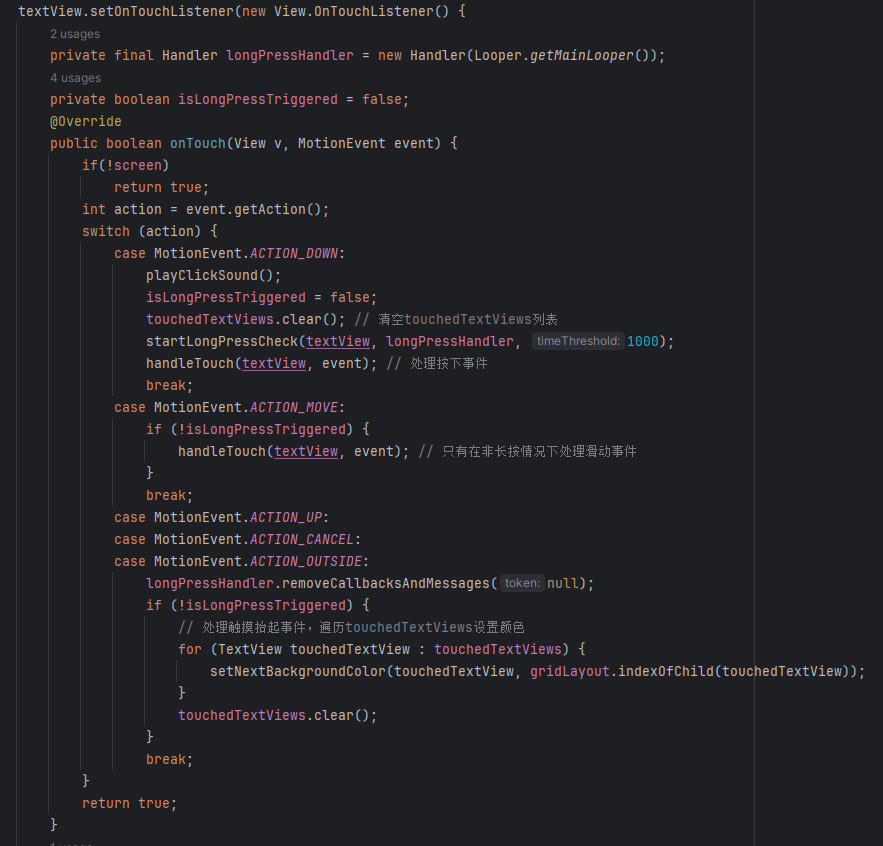
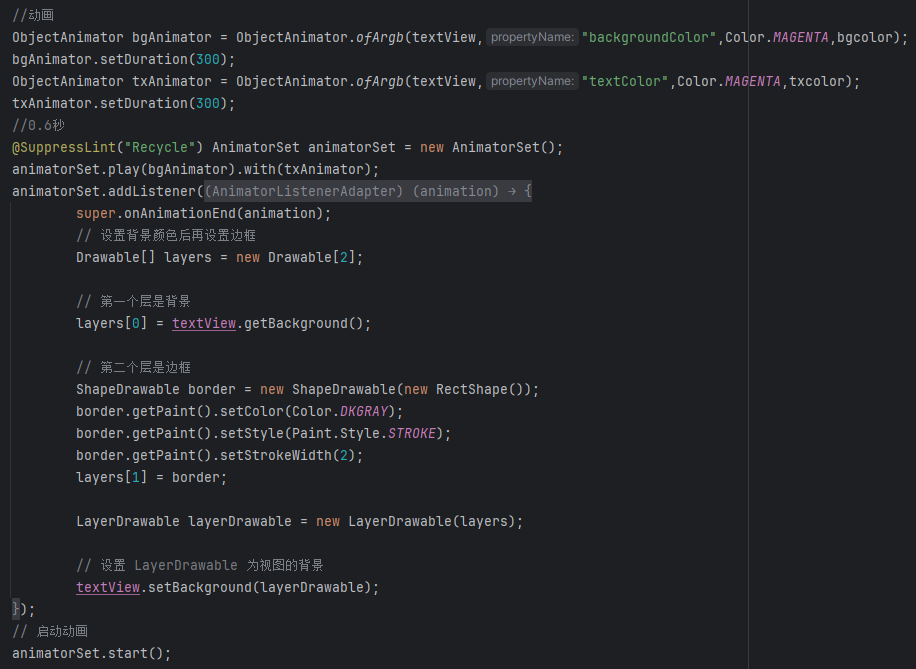
游戏界面上方分别是音乐开关，保存清空返回分别对应三个功能，点按保存按钮会将当前玩家填涂的进度储存到数据库中，保存成功后会弹出Toast提示，点击清空按钮时会弹出对话框询问玩家是否确认清空，防止误操作。点按返回会提醒用户保存好数据再返回，这部分功能我们充分考虑到了用户实际游玩的情况，防止数据丢失。

实现代码：

游戏界面的正中心的gridlayout就是进行填图的游戏部分，游戏规则和原版康思填色游戏一致，依据选择地图的不同，游玩页面下的谜面显示的的行列数据也不同。在正常游玩下，玩家可以自由短按点击或长按滑动正方形格子进行涂色。按下某个格子没有松开时格子会变成紫色，松开后会根据先前格子颜色的状态依次变成黑色，灰色，白色，手指拖动经过的格子也会和点击一样变为紫色，整个过程流畅且稳定。点击格子时会有点击音效和渐变动画效果。根据游戏规则，正确完成地图时会自动弹出通关提示。

实现代码：

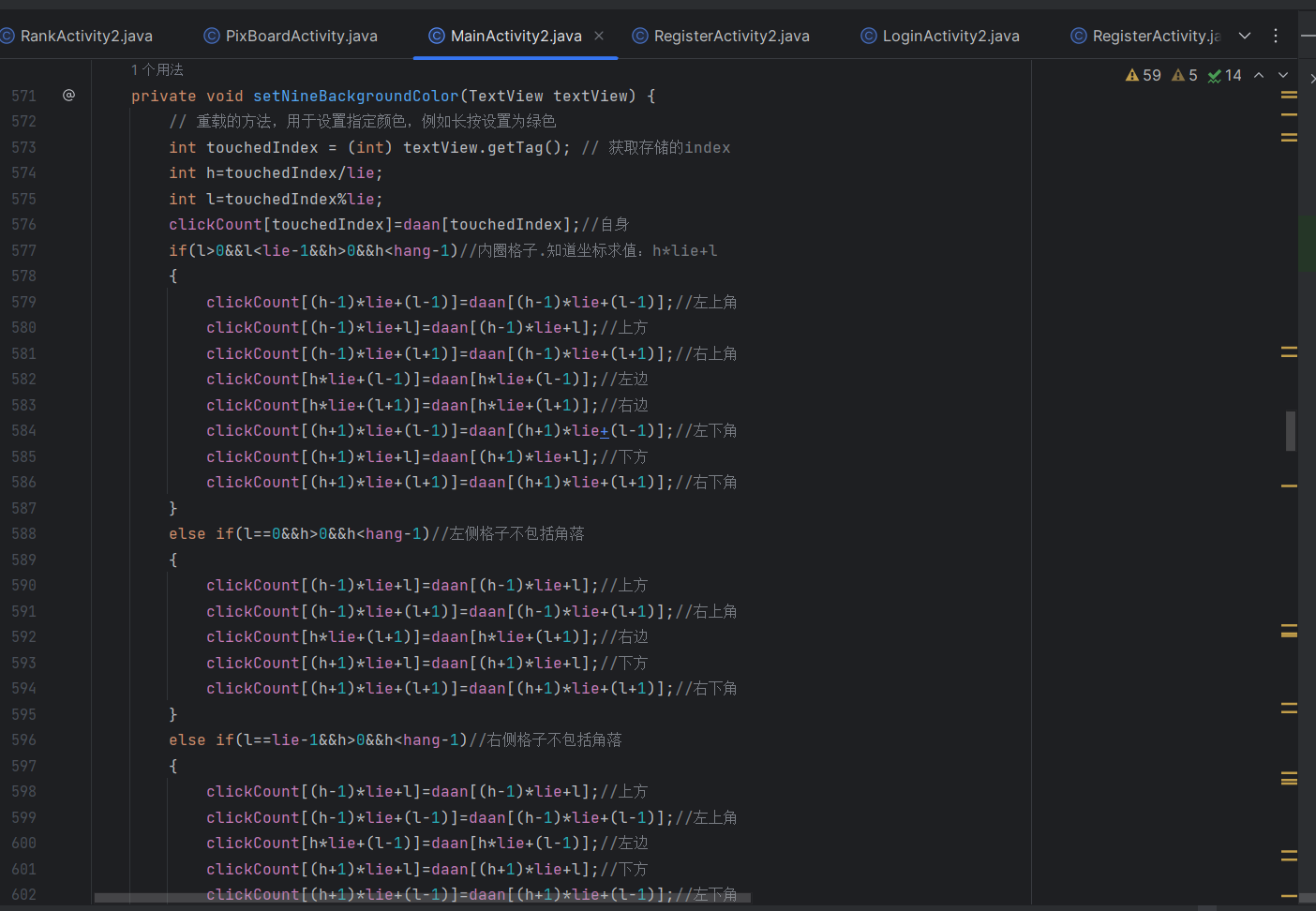
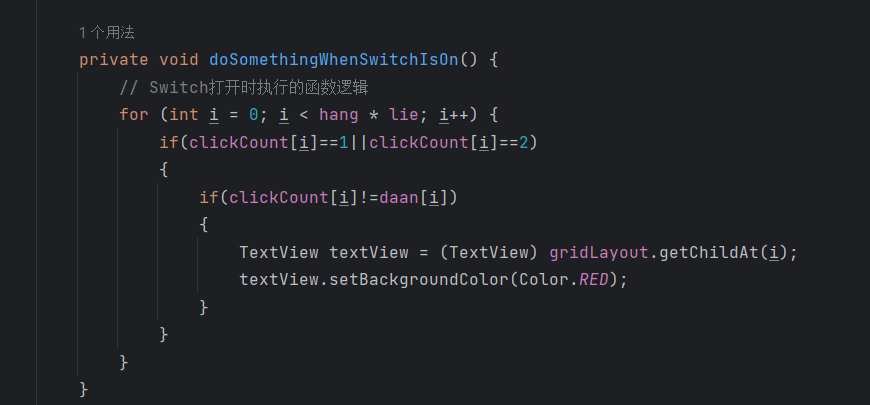




界面的下方有一个提示开关。开启提示功能后，会禁用滑动效果，同时显示辅助功能中的错误开关和答案按钮。玩家在提示功能开启时可以长按格子，应用会自动将点按的方格周围的九格区域涂上正确的颜色。开启错误功能后，会将当前已涂色的格子进行比对判断，玩家涂错的方格会变成红色。

点击答案按钮后，会将当前的地图的正确谜底以对话框的形式显示出来。

实现代码：

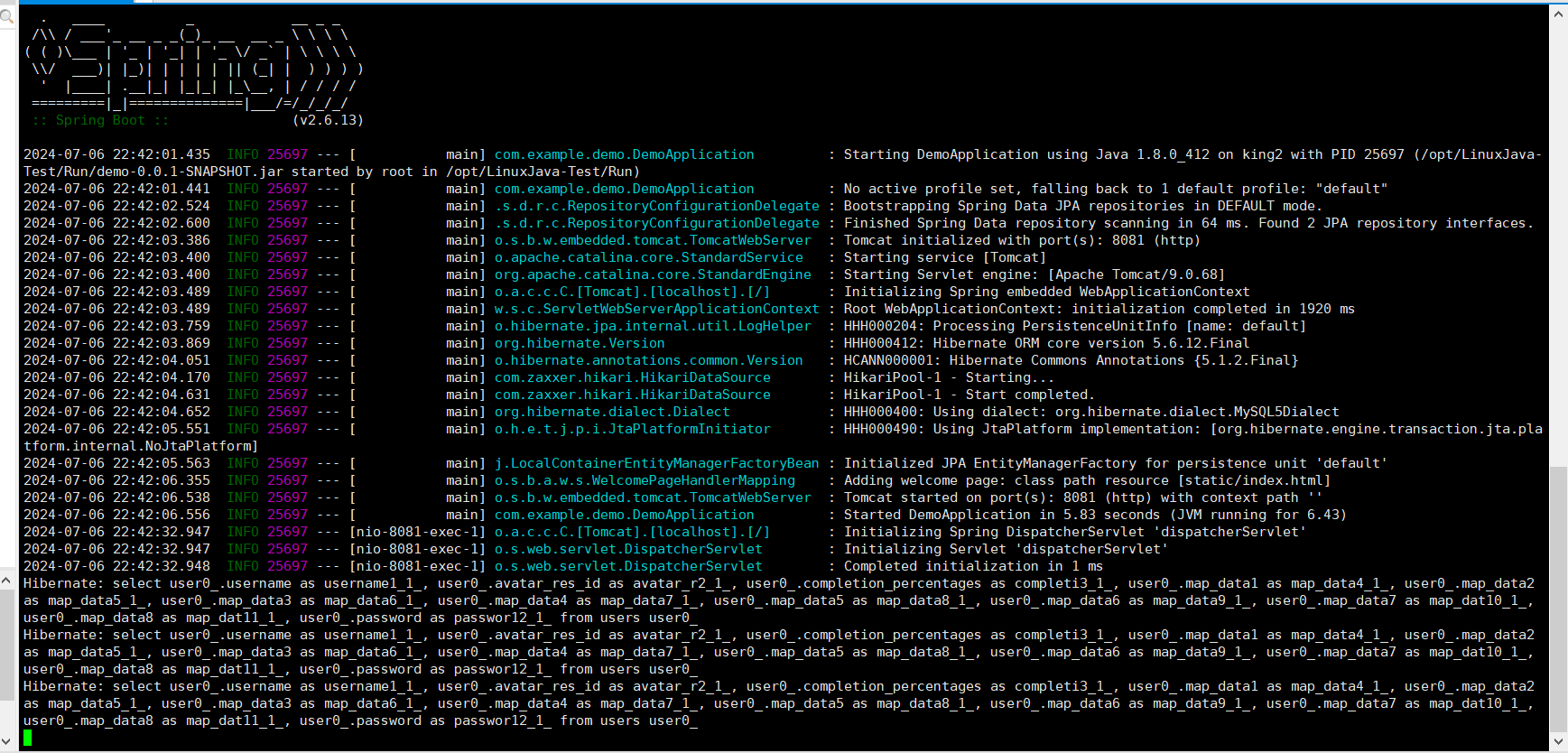


## 八. 系统测试

  由于离线模式和联机模式实现效果类似，在线功能的操作已涵盖离线功能，所以我们主要以对联机模式的测试进行展示。

1. 测试登录功能：

输入用户登录后，提示登录成功：

服务端接收到查询数据库指令：

1. 测试注册功能：

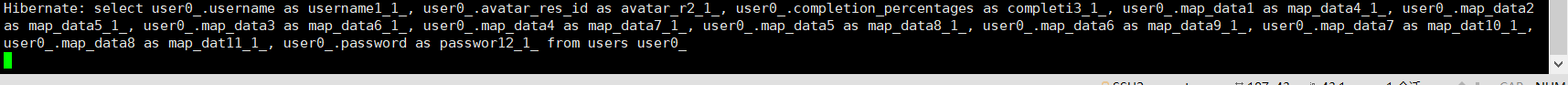
点击注册按钮后，弹回登录界面并提示注册成功：

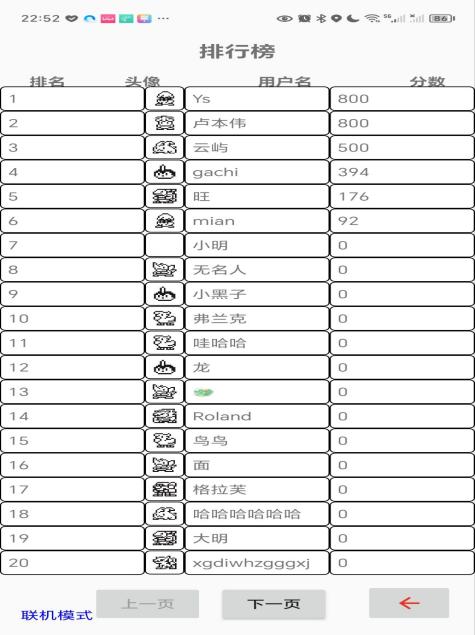
服务端接收到插入数据库指令：

同时客户端可以用新注册的用户进行登录：

1. 测试刷新功能：

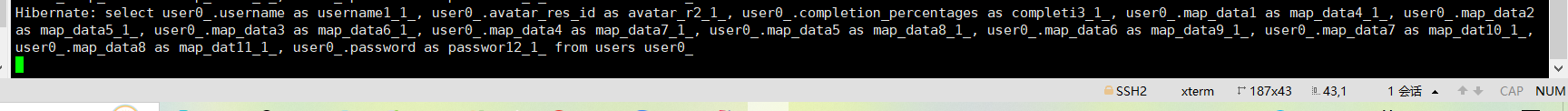
点击游戏菜单界面的刷新按钮，提示刷新成功：

服务端接收到查询数据库指令：

1. 测试排行榜功能；

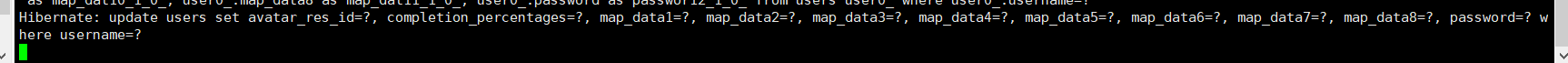
点击侧拉框中的排行榜按钮，进入排行榜界面，展现出当前的排名。

服务器接收到查询数据库指令：



1. 测试游戏保存功能。

点击保存按钮后，提示保存成功，再次回来后地图数据依然存在。

服务端接收到更新数据库指令：

系统测试结束，一切正常。

除此之外，我们还请来了十余位测试玩家来试玩我们游戏的在线模式。他们来自不同城市的不同网络环境，在他们的反馈下我们的在线模式工作稳定，服务器运作正常。

## 九． 遇到的问题、解决办法或改进思路

问题1：在开发初期制作游戏填图功能时，无法实现滑动效果，只能一个个点击，游戏体验效果不好。

解决办法：通过增加监听格子的按下方式，对滑动效果进行相关涂色和数据设计。

问题2：点击或滑动格子后边框消失了。

解决办法：通过检查代码，发现原因是每次点击后绘制背景色后会把边框覆盖消失。于是我们在修改背景颜色的同时每次都再次新建边框。

问题3：只有瞬间滑动，玩法还是太单调了。

改进思路：我们又在点击格子时增加了动画效果和音效，让用户体验更好。同时，在提示模式中实现出了类似原版康思的快速九宫格填色。

问题4：游戏提示模式下，长按需要检测周围九宫格的坐标。一维数组的横纵坐标位置较难获取，自动填色难以实现。

解决办法：通过反复分析，数学计算，取余取整等获取各个方格的位置坐标，并依据方格位置的不同进行不同的填涂操作。

问题5：Android访问通过服务器访问数据库较难实现。

解决办法：通过上网学习，我们学会了okhttp的使用方法，完成了数据交互。

问题6：无法修改默认主题的按钮颜色。

解决办法：在themes中修改为新的主题，实现自由修改各个ui的颜色，使得界面效果更加美观。

问题7：在中期开发中，我们发现在游戏游玩的过程容易出现操作bug，方格在颜色变化的过程中会卡在过渡状态紫色，这种bug难以消除。

解决方法：通过不断测试和和主动学习，我们最终找出了问题所在，在安卓应用的动作逻辑中，当控件中有第二个按下的手指时，前一个按下的手指会因为第二个手指未松开而无法触发松手为方格填色，故而卡在紫色过渡态。这实际上是我们早期代码的逻辑问题。通过不断查阅资料和优化代码，我们最终实现了很稳定的交互操作，bug由此消除。

问题8：在应用切换到后台再回到前台时，背景应用会直接打开，无论音乐switch开关与否。

解决办法：在activity的onResume()方法中增加判断条件，若开关为关闭状态则不会播放音乐。

问题9：在应用开发中期，我们发现了早期代码遗留的诸多问题。例如activity之间切换时造成应用闪退等。

解决方法：我们不断优化代码，致力于应用的更高稳定性。通过修改activity的切换逻辑，实现了进退页面不闪退。

问题10：在游玩过程中，我们发现了早期设计缺失的一些小细节。例如游玩过程中点按清空按钮时，游戏会直接清空当前地图而没有提示。

解决方法：我们不断追求更高的用户友好性，在进行例如返回和清空等操作时，应用会弹出一个对话框询问玩家是否确认操作，防止用户误操作导致游戏进度缺失。

问题11：应用开发后期，我们发现有很多玩家对游戏规则不熟悉，不能进行正常游戏。

解决方法：我们在主菜单设计出了一个新手教程，并在游玩界面提供辅助游戏选项，帮助新玩家更快更好上手。

问题12：仅仅是我们开发者试玩运行，难以找到所有应用中的细节问题。

解决方法：我们找到了十余位愿意试玩我们游戏的普通玩家进行测试游玩。这些玩家们为我们的游戏出谋划策，为我们的后期优化提供了诸多宝贵建议。玩家测试也是游戏开发的一个重要部分。

问题13：排行榜在账号数量过多时会溢出应用边界，并且不可滑动查看更多信息，导致信息显示不全。

解决办法：由于我们的排行榜信息是动态生成TableLayout，所以无法使用Android Studio自带的滑动视图。在经过讨论和查阅资料后我们找到了改进方案，一页显示20行玩家信息，通过点按上下页按钮进行翻页操作查看更多玩家信息。经过测试我们的代码已经能够稳定实现上述功能。

问题14：在开发过程中，Android Studio与实体安卓手机连接的adb总是频繁断联重启，影响了开发效率。

解决办法：通过查阅网络资料，我们找到的解决方法是更换更好的手机数据线，重启手机上的USB调试，在运行Android Studio之前提前运行adb进程等。

问题15：我们开发者三人在协同开发该应用时，遇到了因为gradle配置问题导致的无法正常运行编译工程的问题。

解决办法：积极查阅资料，我们明白了gradle文件中各项配置的含义，能够在不同电脑中交换运行同一个工程项目。

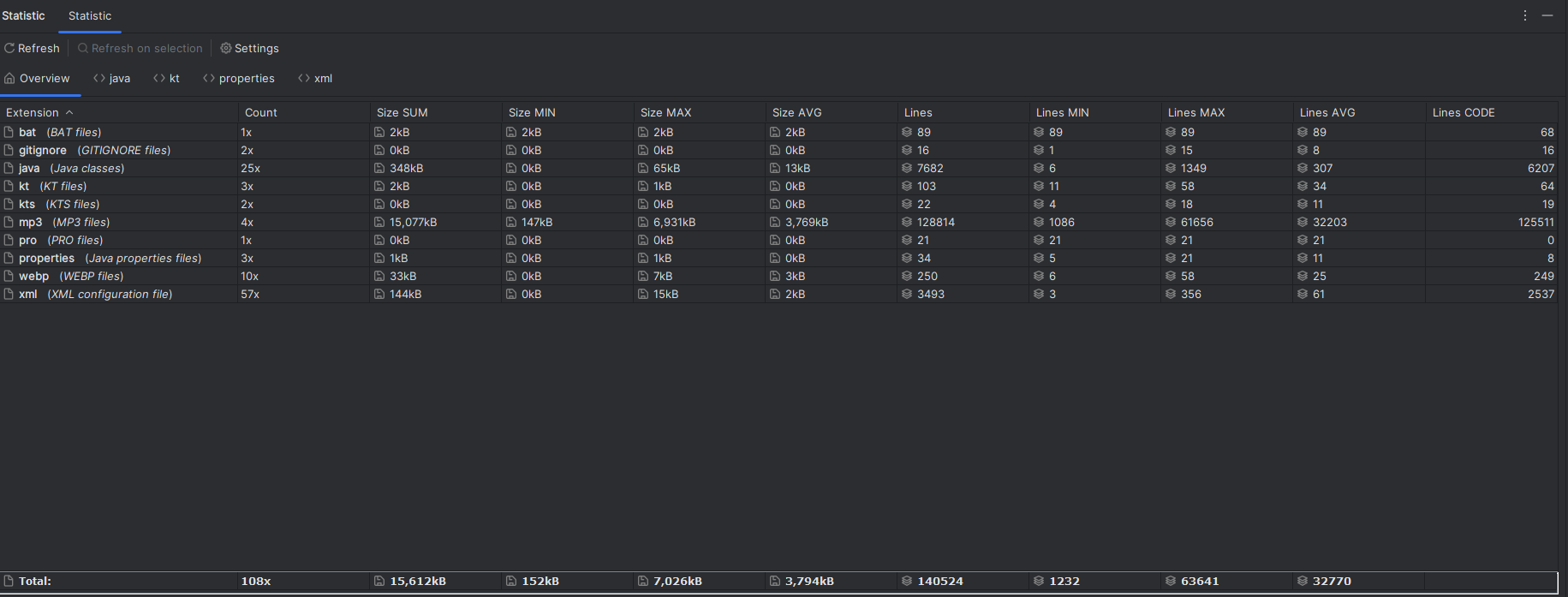
## 十． 总结

通过本次实验，我们成功开发了一款拥有离线和在线双模式的康思填色块游戏，并在应用操作和优化上不断刻苦钻研，给予玩家非常流畅的体验。

我们从本次应用开发实践中收获了大量知识。通过本次实验，我们对Android Studio的使用变得更加熟练，并深刻体会到其强大的功能。该开发软件能够实现许多功能，包括添加图像、音乐和动画效果。通过这次实验，我们掌握了Android Studio的使用方法，未来可以根据轻松制作所需的软件，实用性极高。

此外，本次实验使我们学会了多种方法和算法，以及不同的程序设计思路。我们通过不断实践和构思，交流与合作学习，最终实现了我们在Android平台上的应用程序设计。

实验过程中，我们也深刻体会到了团队合作的重要性。项目中的每个开发成员都是不可或缺的一部分，我们通过合理的分工和协作，最终成功完成了这款Android游戏。

本应用的工程最终代码量中，java代码共6207行，xml代码2537行。一切都尽力为了更好的实现效果。