**项目总结**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 学号 |  | 姓名 |  | 手机号码 | |  | |
| 班级 |  | 实验室 |  | | 邮箱 | |  |
| 项目  成果  摘要 | 项目名称： | SpeedCloud-Android | | | | | |
| 伴随着手机行业的蓬勃发展，越来越多的应用为了用户的粘性和更好的用户体验，不再局限于web端，都开始开发起了自己的手机端应用。本项目也秉承着这一理念，特此开发SpeedCloud安卓版。  本项目为用户提供文件云存储服务和文件预览服务。主要实现了用户登录、用户注册、忘记密码、文件查看、文件下载、文件上传、文件删除、文件移动、文件重命名、文件分享等一些功能。本软件使用的所有网络接口为web实训后端提供。  以下分用户操作和文件操作介绍具体功能。首先，用户可以在账户界面切换登录、注册或忘记密码三个面板，输入正确的账号密码即可进入网盘主界面，同时，若用户勾选了自动登录功能，下次打开应用时会自动登录，进入主界面后选择“我的”选项卡可以查看网盘容量使用情况，还可以退出当前账户。其次，进入网盘主界面后，可以下拉刷新文件列表，也可以按右下角的圆形加好添加新文件夹或上传本地文件，还可以长按文件列表项呼出文件工具栏，文件工具栏上有例如文件下载、文件删除、文件移动、文件重命名、文件分享等功能。点击右上角图标还能查看当前正在上传或下载的文件记录，或查看已经上传完毕或下载完毕的文件历史记录。  签名：2022年1月9日 | | | | | | |
| 教师评分 | 签名：2022年 1月 日 | | | | | | |

**SpeedCloud-Android项目总结报告**

2022年1月9日

**1 项目开发的意义**

**1.1 主要用途和价值**

本项目为用户提供文件云存储服务和文件预览服务。主要实现了用户登录、用户注册、忘记密码、文件查看、文件下载、文件上传、文件删除、文件移动、文件重命名、文件分享等一些功能。首先，用户可以在账户界面切换登录、注册或忘记密码三个面板，输入正确的账号密码即可进入网盘主界面，同时，若用户勾选了自动登录功能，下次打开应用时会自动登录，进入主界面后选择“我的”选项卡可以查看网盘容量使用情况，还可以退出当前账户。其次，进入网盘主界面后，可以下拉刷新文件列表，也可以按右下角的圆形加好添加新文件夹或上传本地文件，还可以长按文件列表项呼出文件工具栏，文件工具栏上有例如文件下载、文件删除、文件移动、文件重命名、文件分享等功能。点击右上角图标还能查看当前正在上传或下载的文件记录，或查看已经上传完毕或下载完毕的文件历史记录。

**1.2 社会意义和经济价值**

伴随着手机行业的蓬勃发展，越来越多的应用为了用户的粘性和更好的用户体验，不再局限于web端，都开始开发起了自己的手机端应用。本项目也秉承着这一理念，特此开发SpeedCloud安卓版。

**1.3 项目开发的可行性**

本项目开发工具及平台如下：

开发工具：Android Studio Arctic Fox | 2020.3.1 Patch 2 Build #AI-203.7717.56.2031.7678000, built on August 27, 2021

开发平台：

操作系统Windows 10\_20H2\_64位

处理器Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz 2.81 GHz

内存16GB

虚拟机运行环境：

Runtime version: 11.0.10+0-b96-7249189 amd64

VM: OpenJDK 64-Bit Server VM by Oracle Corporation

开发语言：Kotlin

**2需求分析与数据库设计**

**2.1 需求分析**

通过网络调研及结合自身体验现有的网盘软件，现提出如下需求：

1、用户可以通过邮箱获取验证码注册账号，还能通过邮箱发送验证码找回密码，还能通过用户名和密码登录；

2、用户可以查看自己网盘容量的使用情况；

3、用户可以查看网盘文件、在线浏览文件、下载文件、上传文件、删除文件、分享文件、移动文件、重命名文件等一系列文件操作；

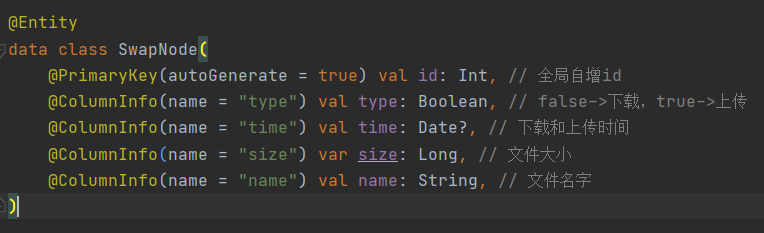
**2.2 功能设计重点**

1、基本功能：用户注册、登录、用户查看文件、上传文件、下载文件

2、附加功能：忘记密码、删除文件、分享文件、移动文件、重命名文件、回收站

**2.3 数据字典**

数据库采用SQLite，表结构如下，采用Android中的Room作为中间层



**3软件功能设计**

**3.1 软件架构设计**

软件主要分为3个界面，分别为用户登录界面、网盘主界面、下载上传界面。

网盘主界面采用singleTask模式。界面流程为先进入用户登录界面，然后进入网盘主界面，根据点击能跳转到下载上传界面。用户登录界面中登录、注册、忘记密码三个界面的切换通过fragment实现，网盘主界面和下载上传界面也是通过fragment和tabLayout以及ViewPage2实现。

**3.2 功能分析和模块划分**

大体上分为用户模块和文件模块。

1、用户模块：

a)注册账号：用户可以通过邮箱在本系统注册账号，在发起注册请求时需要通过邮箱进行验证，输入正确的验证码，方可完成注册。

b)登录：用户可以通过用户名和密码登录系统。

c)重置密码：用户可以通过邮箱重置密码，在发起重置密码请求时需要通过邮箱进行验证，输入正确的验证码，方可完成重置密码。

2、文件模块

a)上传文件：用户可以上传文件或文件夹存储在系统中。

b)下载文件：用户可以从系统上下载存储的文件或文件夹，保留目录结构。

c)预览文件：用户可以在系统上无需下载，在线预览图片或视频。

d)删除文件：用户可以在系统上删除不需要存储的文件，删除的文件进入回收站，10天内不还原则由系统自动永久删除不可还原，10天内可选择还原文件。

e)分享文件：用户可以分享自己存储在系统上的文件，将文件和其他用户共享，其他用户可以通过生成的唯一URL链接访问到该用户分享的文件

f)保存分享文件：用户可以通过唯一URL链接访问到其他用户生成的共享文件，将共享文件保存到自己的目录结构中。

**4软件开发与实现**

包括使用具体的技术、开发工具，重要功能的编程实现方法、软件测试等。

**4.1 开发工具**

开发工具：Android Studio Arctic Fox | 2020.3.1 Patch 2 Build #AI-203.7717.56.2031.7678000, built on August 27, 2021

开发平台：

操作系统Windows 10\_20H2\_64位

处理器Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz 2.81 GHz

内存16GB

虚拟机运行环境：

Runtime version: 11.0.10+0-b96-7249189 amd64

VM: OpenJDK 64-Bit Server VM by Oracle Corporation

开发语言：Kotlin

**4.2 重要功能的编程实现**

**4.2.1 用户界面的切换**

利用fragment的show、hide和add实现fragment懒加载

    /\*\*

     \* 切换登录、注册、忘记密码的fragment

     \* 通过add、hide、show切换

     \*/

     private fun setFragment(type: FragmentType) {

        setTextButton(type)

        val mFragmentManager = supportFragmentManager

        val mTransaction = mFragmentManager.beginTransaction()

        for (i in iFragment.indices) {

            iFragment[i]?.let {

                mTransaction.hide(it)

            } // 不为null才隐藏

        }

        if (iFragment[type.ordinal] == null) { // 未创建实例，则创建并添加

            iFragment[type.ordinal] = cFragment[type.ordinal]()

            mTransaction.add(R.id.frameLayout, iFragment[type.ordinal]!!)

        } else { // 已创建则显示

            mTransaction.show(iFragment[type.ordinal]!!)

        }

        mTransaction.commitAllowingStateLoss()

    }

初始化的变量

private lateinit var iFragment: Array<Fragment?> // Fragment实例

private lateinit var cFragment: Array<() -> Fragment> //newInstance的函数引用

iFragment = Array(FragmentType.values().size) { null } // 初始化为null

cFragment = arrayOf(

    LoginFragment::newInstance,

    RegisterFragment::newInstance,

    ForgetFragment::newInstance

)

最后只需在切换文本上添加监听调用setFragment跳转

**4.2.2 自动登录**

将上次登录的账号密码保存到共享参数中，每次创建登录界面时会先从共享参数中拿去账号和密码，若勾选了自动登录，则会调用登录按钮所绑定的点击事件

// 获取共享参数中的信息

val username = SharedUtils.readString("username", "")

val password = SharedUtils.readString("password", "")

val check = SharedUtils.readBoolean("autoLogin", false)

root.findViewById<EditText>(R.id.username).setText(username)

root.findViewById<EditText>(R.id.password).setText(password)

root.findViewById<CheckBox>(R.id.autoLogin).isChecked = check

// 勾选自动登录且用户名和密码不为空，就自动登录

if (check && !username.isNullOrEmpty() && !password.isNullOrEmpty()) {

    clickButton()

}

**4.2.3 网盘查看文件及回退**

文件列表fragment中维护了一个backStack返回栈，每次点击文件夹进入会将文件夹信息存入返回栈，并重设RecyclerView的列表项。

    /\*\*

     \* 返回能否回退上一层，能就直接回退上一层。注意：如需调用getSelectedItem()，需在调用back()前

     \* @return true为可以回退，false为已经到达根目录，无法回退，即将退出应用

     \*/

     fun back(): Boolean {

        if (adapter.selectStatus) {

            adapter.cancelSelect()

            swipeRefresh.isEnabled = true

            fileToolbarWindow.dismiss()

            fileActionbarWindow.dismiss()

            floatingActionButton.show()

            return true

        } // 处于select模式

        if (backStack.size <= 1) return false

        backStack.removeLast()

        refreshPath()

        return true

    }

然后重写activity的onBackPressed就可以实现目录的回退

override fun onBackPressed() {

    // 不在文件页面或已经退到根目录

    if (binding.tabLayout.selectedTabPosition != 0 || !fileFragment.back()) super.onBackPressed()

}

长按列表项进入编辑模式，编辑模式中的两个弹窗由popupWindow实现

    /\*\*

     \* 实现长按列表项的事件

     \*/

     inner class MyOnItemLongClickListener : RecyclerListener.OnItemLongClickListener {

        override fun onItemLongClick(view: View, position: Int) {

            fileToolbarWindow.showAsDropDown(root) // 显示底部工具栏

            fileActionbarWindow.showAtLocation(root, Gravity.START or Gravity.TOP, 0, 0) // 显示顶部操作栏

            swipeRefresh.isEnabled = false // 隐藏下拉刷新

            floatingActionButton.hide() // 隐藏右下角新建按钮

        }

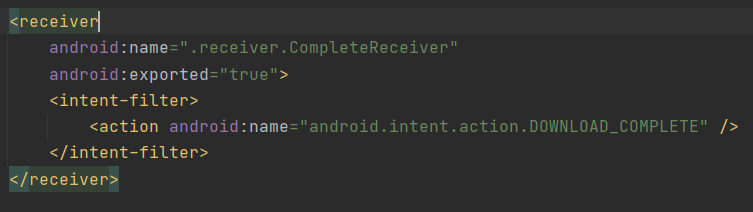
    }

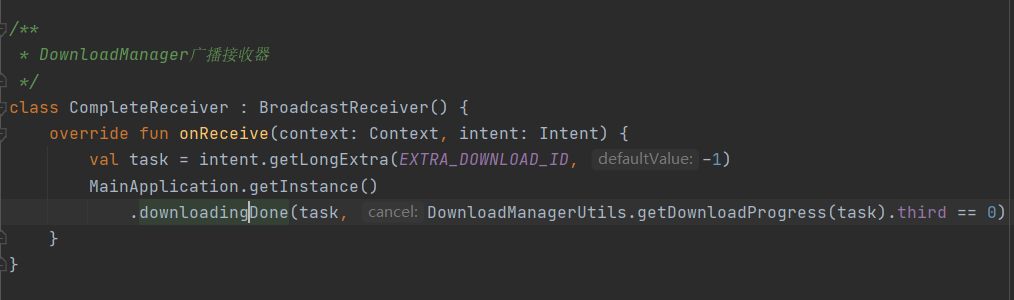
**4.2.4 文件下载**

采用DownloadManager，将下载数据存入数据库，然后一秒查询一次正在下载的文件，当有下载完成时会更新已经下载完的信息。



注册广播，接收DownloadManager完成任务的广播



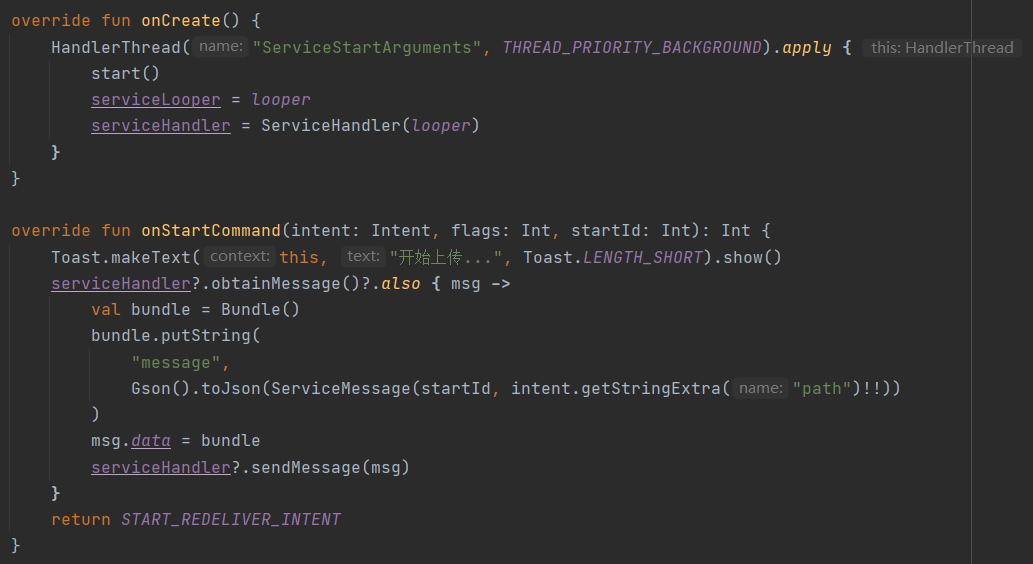


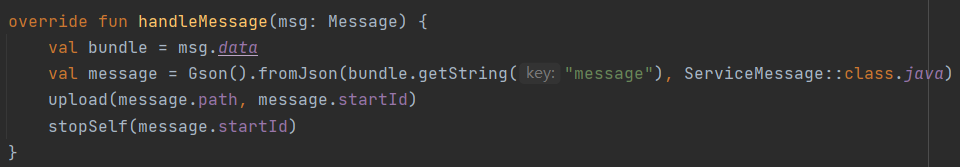
**4.2.5 文件上传**

文件上传采用安卓服务的形式

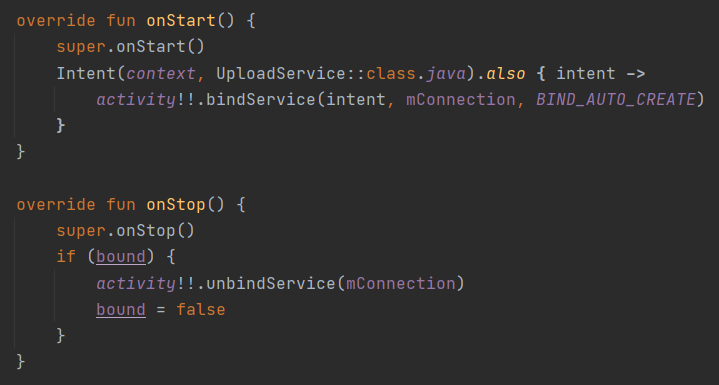


在服务中通过intent接收文件信息并开启handler进行任务下载

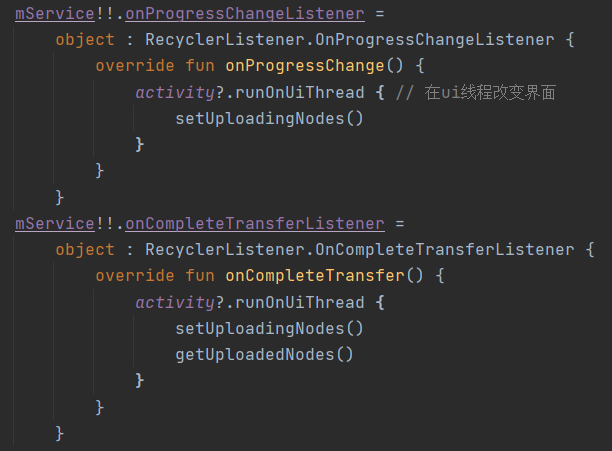




进入上传界面，采用绑定服务进行监听的方式获取最新下载信息



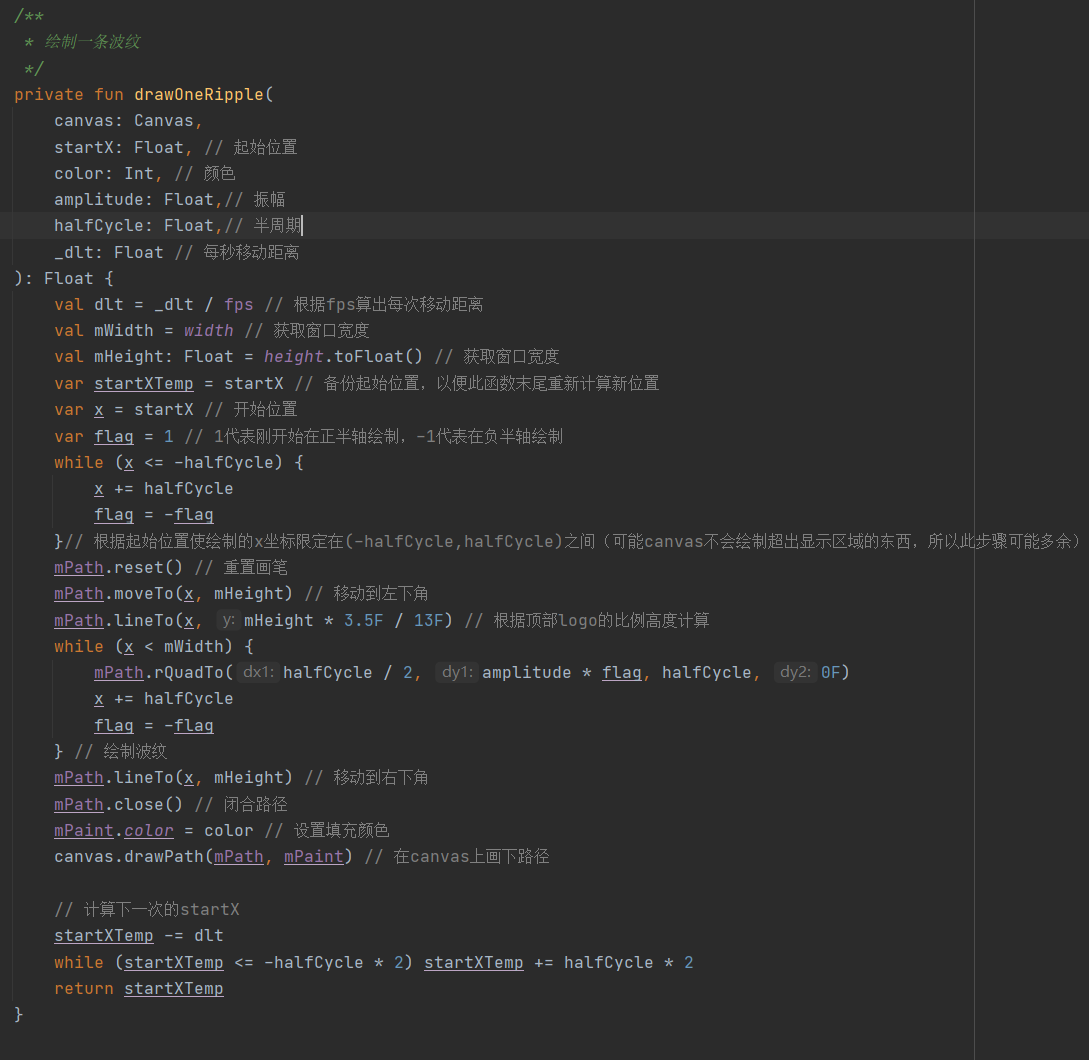
一个是进度条监听，一个是任务完成监听



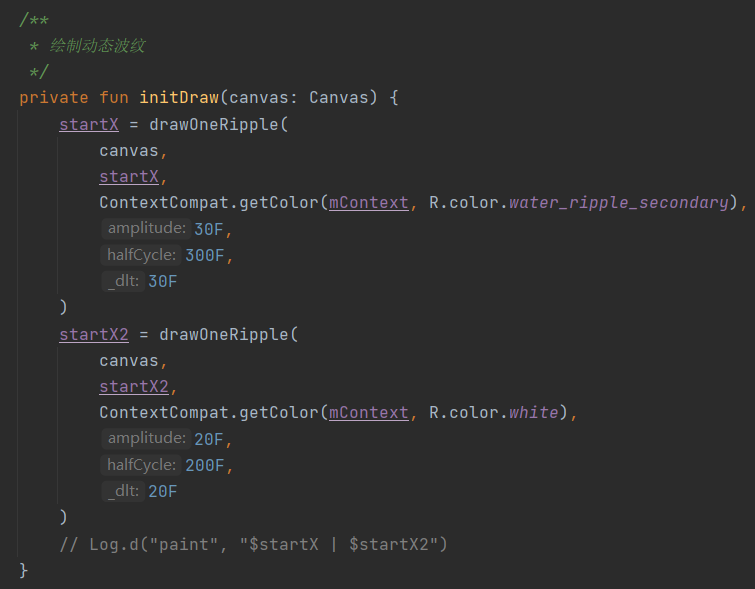
**4.2.6 用户界面波纹背景**

自定义组件RippleLinearLayout，继承自LinearLayout。

如下函数绘制一个矩形，其中矩形的一条边为正弦函数模样，并填充颜色。

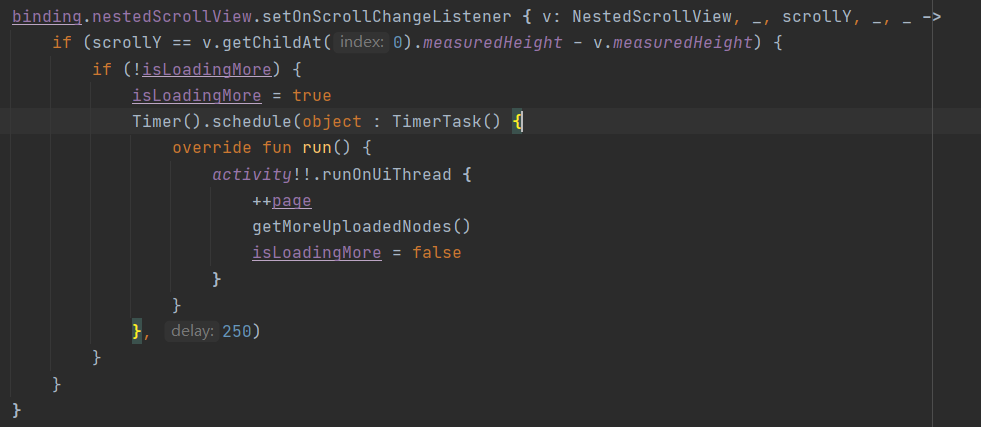


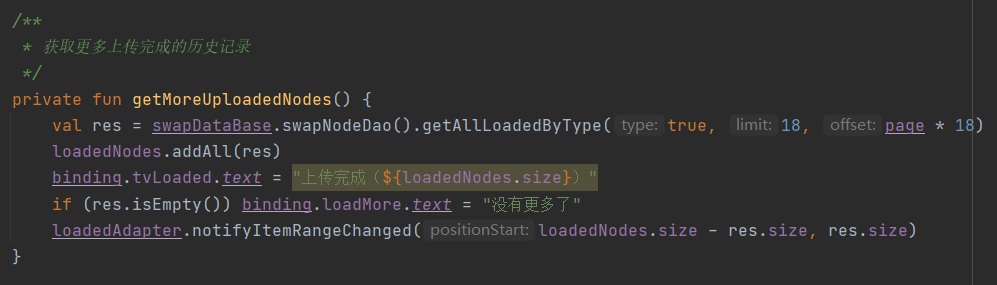
改变起始的x坐标即可使两条波纹动起来



**4.2.7 分页查询**

每次到底部时，从数据库获取更多历史记录



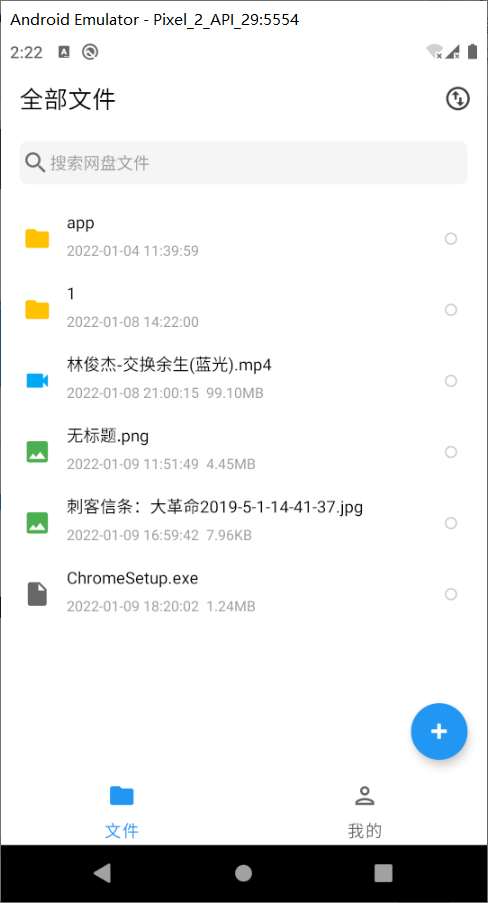
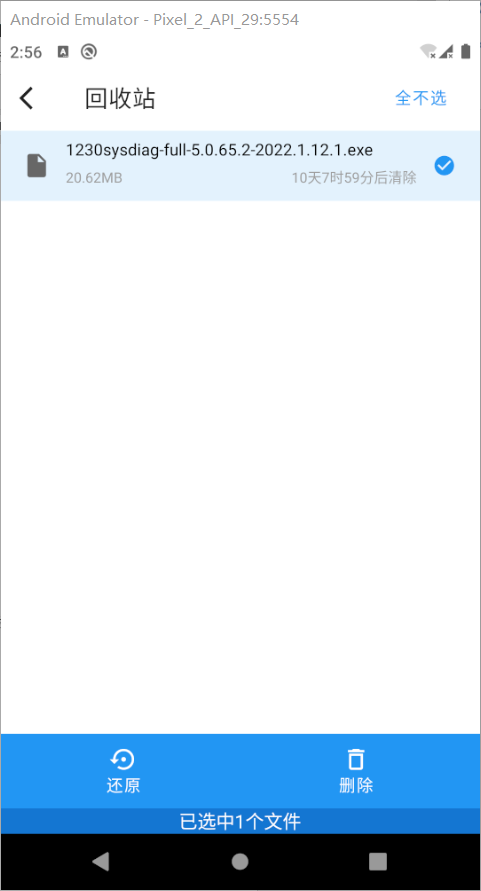


**5项目成果简介**

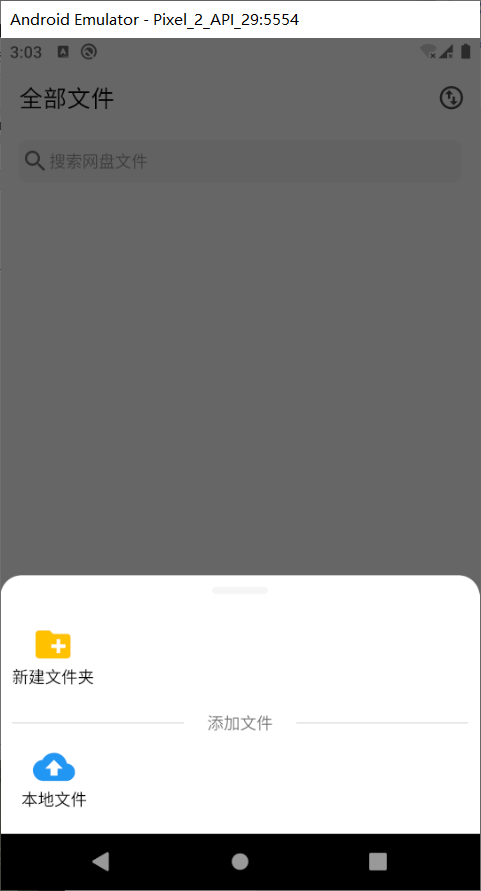
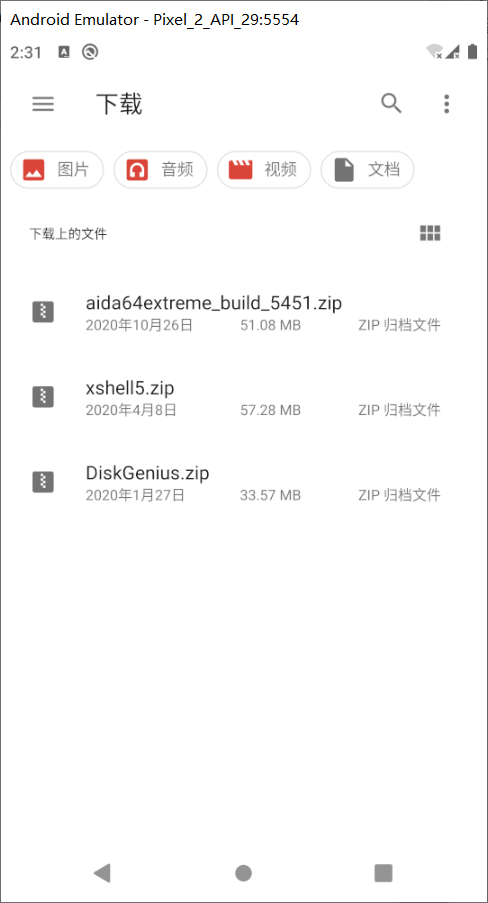
用户登录、注册、忘记密码

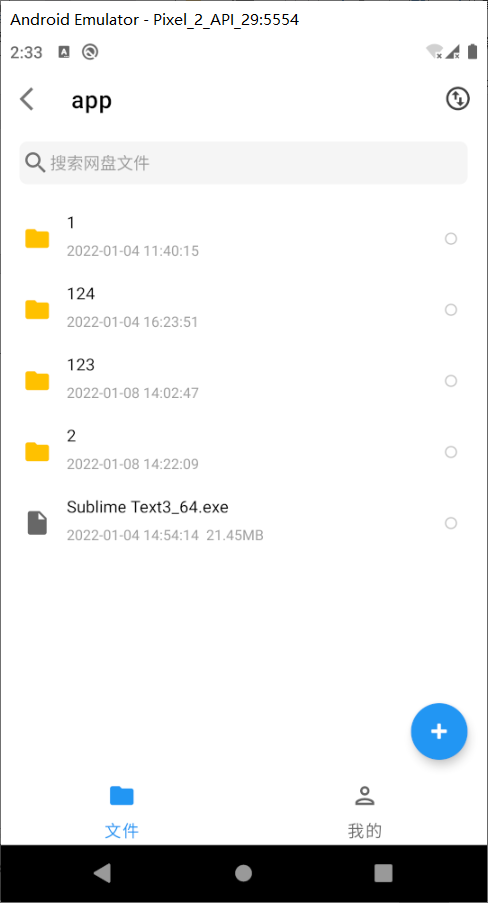
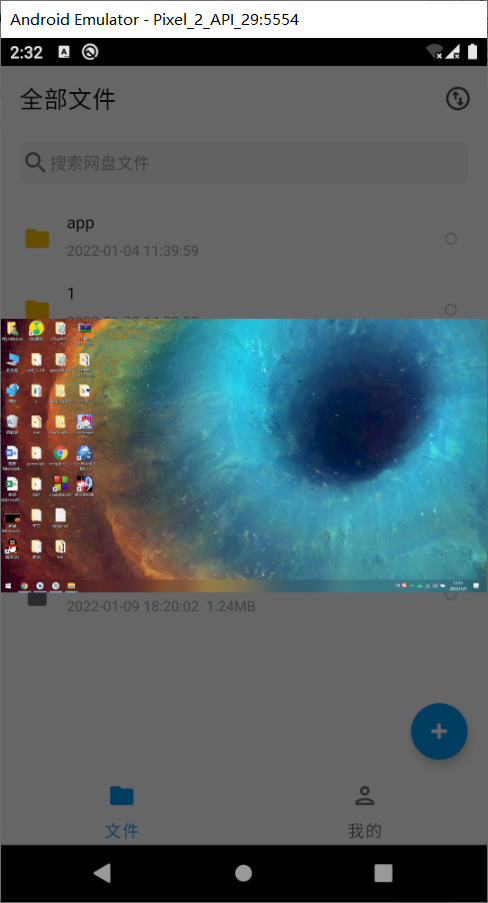
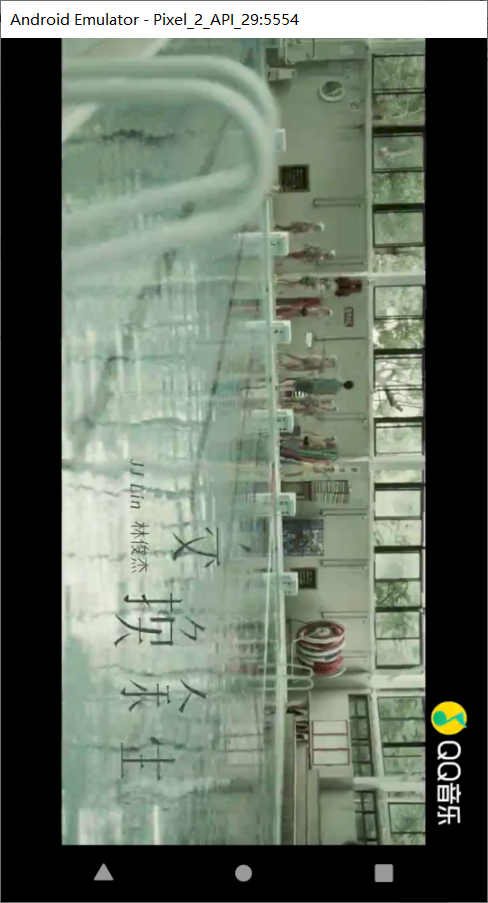
网盘文件、我的界面、回收站

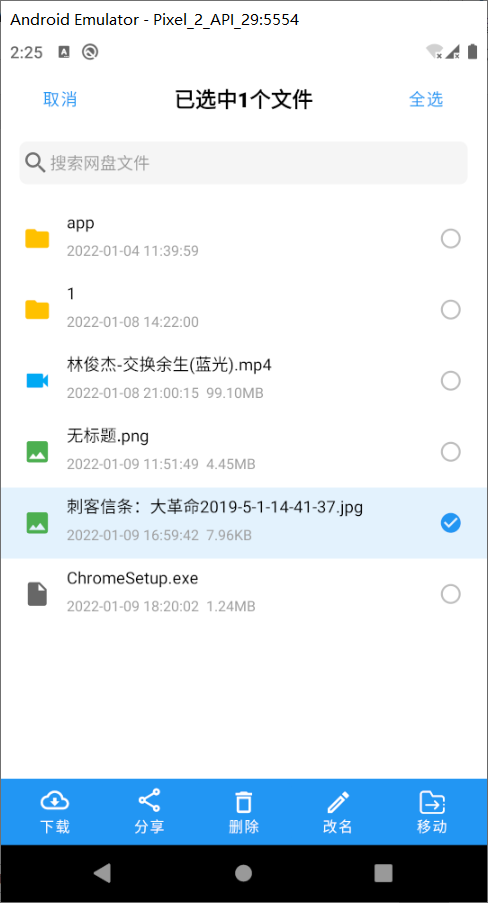
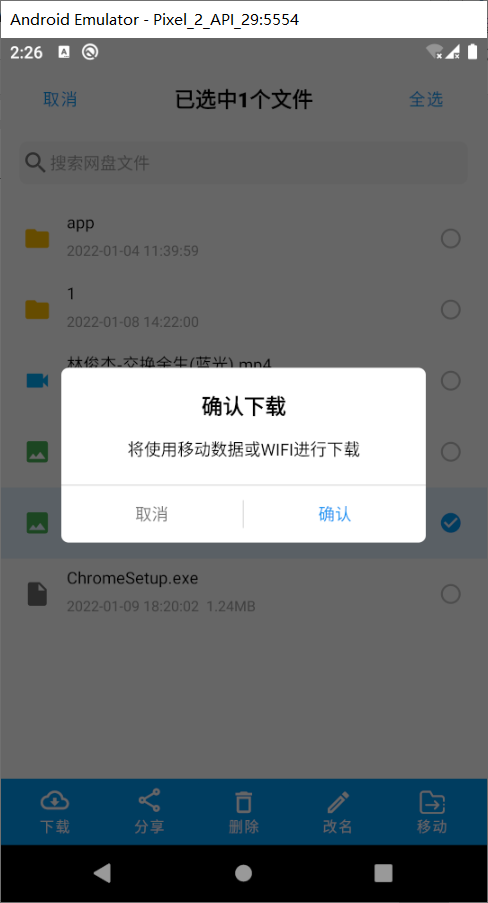
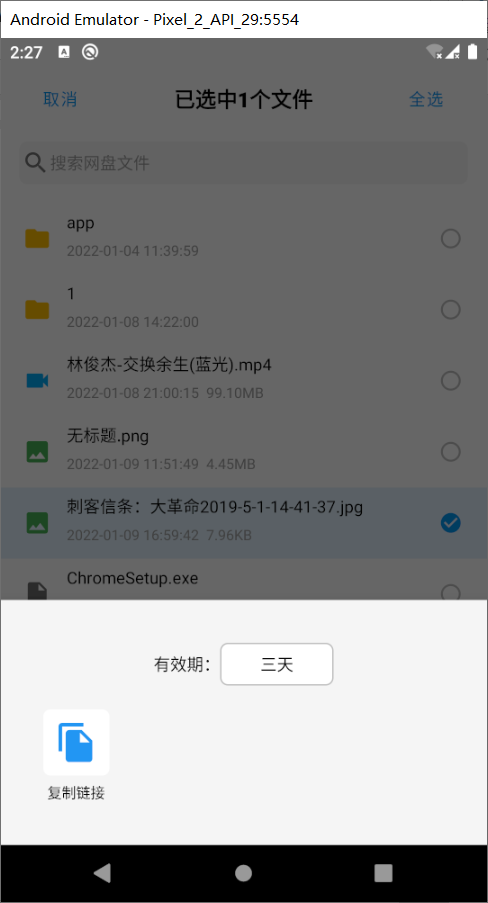
点击右下角新建加号，上传文件会跳转到系统自带的文件管理器中去选择文件

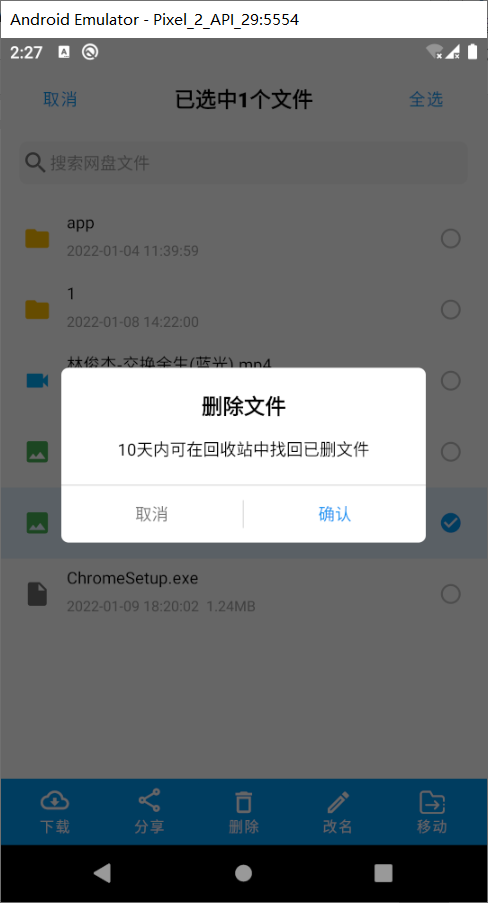
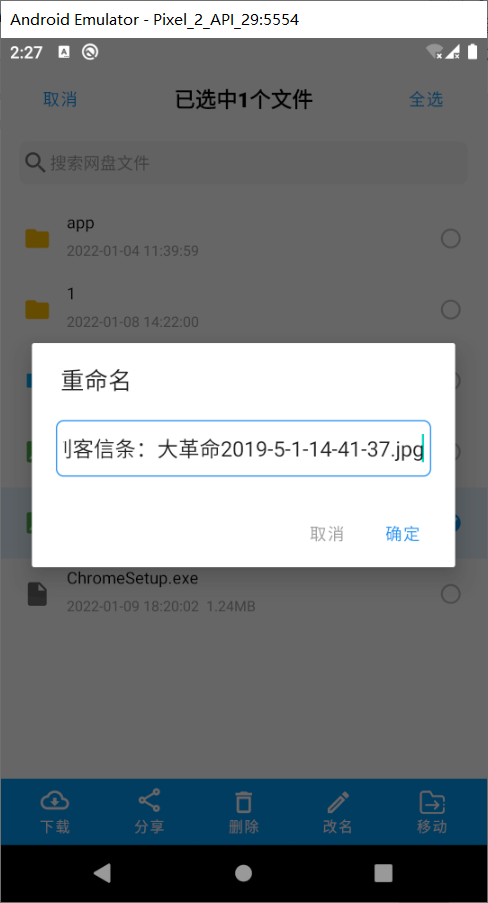
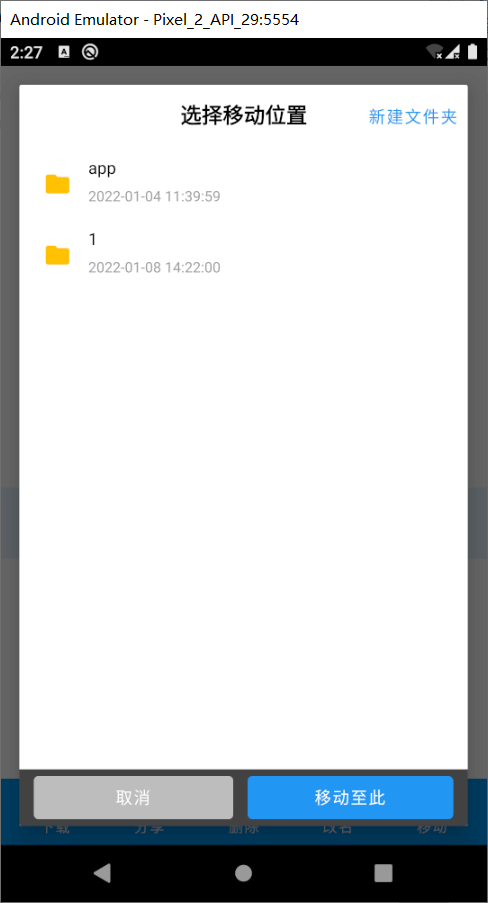
  

单击列表项，分别点击文件夹、图片、视频

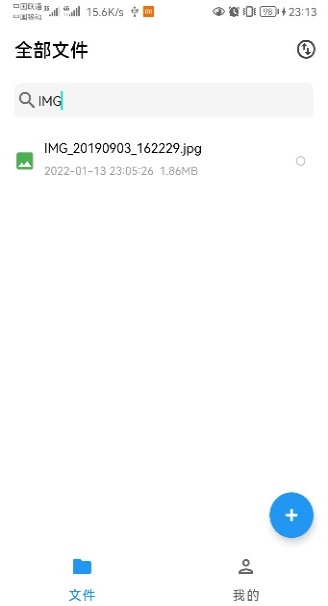
  

长按列表项

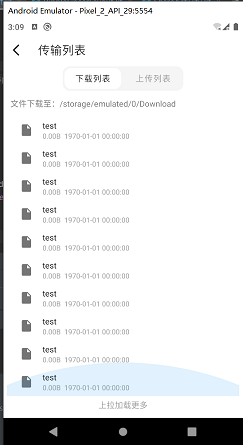
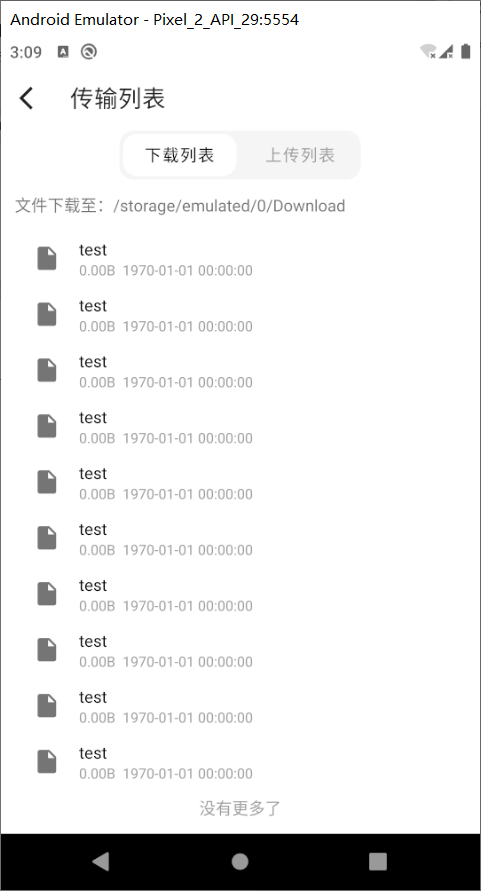
  

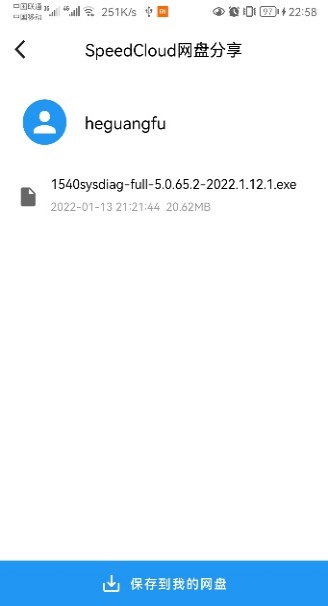
搜索、下载和上传页面

分页查询记录

分享保存文件界面

保存到我的网盘打开与移动文件对话框类似

**6 参考文献与网络资源**

1. Kotlin文档：https://book.kotlincn.net/text/basic-types.html
2. Android开发文档：https://developer.android.google.cn/guide?hl=zh\_cn
3. Android Studio开发工具：https://developer.android.google.cn/studio?hl=zh-cn
4. Kotlin教程：https://www.runoob.com/kotlin/kotlin-tutorial.html
5. 安卓Uri转真实文件路径：https://blog.csdn.net/qq\_42795723/article/details/113862211