登陆注册设计与实现说明文档

完成人：曾德明 学号：20172131138

完成时间：2019 月 10 月 20 号

一、软件名称

模仿 TIM 客户端的登陆注册 demo

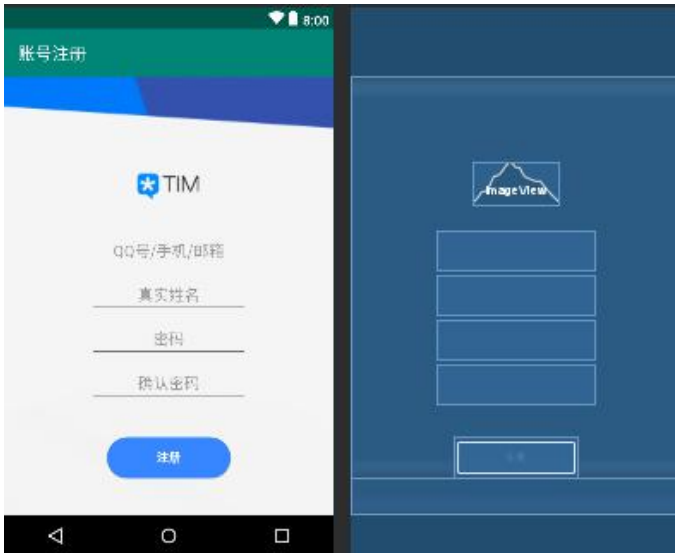
软件 github 链接：https://github.com/PCSKY/android-simpleTIM

二、软件内容简介

这是一个使用 Android studio 编写的模仿 TIM 客户端的登陆注册 demo，整个 app 界面参考了 TIM 的界面风格，使用简介风格进行配置，有独立的登录，注册与主页面。

App 的登录界面我选择使用 LinearLayout 布局作为主布局。首先在主布局里面添加一个 ScrollView 滚动视图，滚动视图是指当拥有很多内容，屏幕显示不完时，需要通过滚动跳来显示的视图，这里主要用于软键盘升起时界面变化。ScrollView 只支持垂直滚动，只接受一个子元素，所以我在里面再嵌套一个 LinearLayout 布局放置头像，信息输入框与登录按钮。信息输入框使用了 TextInputLayout 组件，且设置与两边有距离，显得更加美观，模拟头像显示则使用 ImageView 组件。最后再加入一个 LinearLayout 布局来放置底部的“新用户注册”与“无法登录？”按钮

App 的注册界面我同样选择使用 LinearLayout 布局作为主布局。在主布局里面添加一个 ScrollView 滚动视图放置注册信息输入框与注册按钮，信息输入框使用了 TextInputLayout 组件，且设置与两边有距离



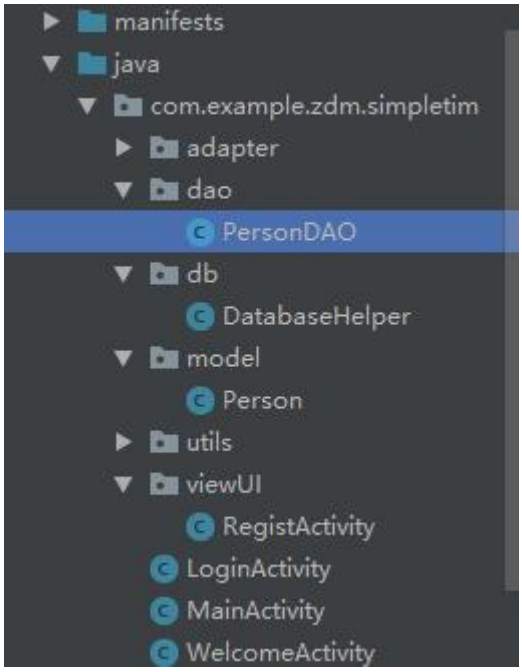
在功能的实现方面，我根据网络上了解的相关知识，实现了 app 的登陆注册功能以及页面的跳转，同时根据自己的理解，参考网上的相关 demo，做出了一个简单的主页。在有关算法的代码方面，我是在参考了网络上的 Android 与 SQLite 的相关知识后进行模仿与实现的，所以有可能代码并不是最简洁的，但是测试起来是可以完成的。

具体的算法与数据结构实现方面，我首先设计了了一个存储用户信息的数据表以及一个专门处理数据库的类 DatabaseHelper，继承于 SQLiteOpenHelper，然后在打开 welcome 界面的时候就顺便创建数据库。打开 app 时，会进入一个欢迎界面，欢迎界面的模式设计成 simgleTask，用于后面安全设置中直接退出 app，在 WelcomeActivity 中设置一个定时器 timer，设定欢迎界面只停留两秒，然后转移到登录界面。在登录模块中，用户在界面填入用户名和密码，当点击登录时，就会调用自己写的函数 attemptLogin()进行登陆信息的确认以及界面的转换，注册的流程和登录的流程基本一样，区别在于注册是把信息写入数据库然后跳转回登录界面，登录则是把数据与数据库中比较，判断登录成功后跳转到主页面。

- tb_person数据表

序号	字段名	字段类型	说明	备注
1	username	nvarchar(30) not null	账号(用户名)	关键字
2	password	nvarchar(30) not null	密码	
3	name	nvarchar(30) not null	真实姓名	

整个软件的逻辑结构大概包含了 model 层，dao 层和 activity 层，还有一些基本的工具类。Model 层用于表示数据库中元素，dao 层用于和数据库进行相应的交互，activity 层则与 layout 进行交互，最终形成一个完整的结构。



三、界面设计

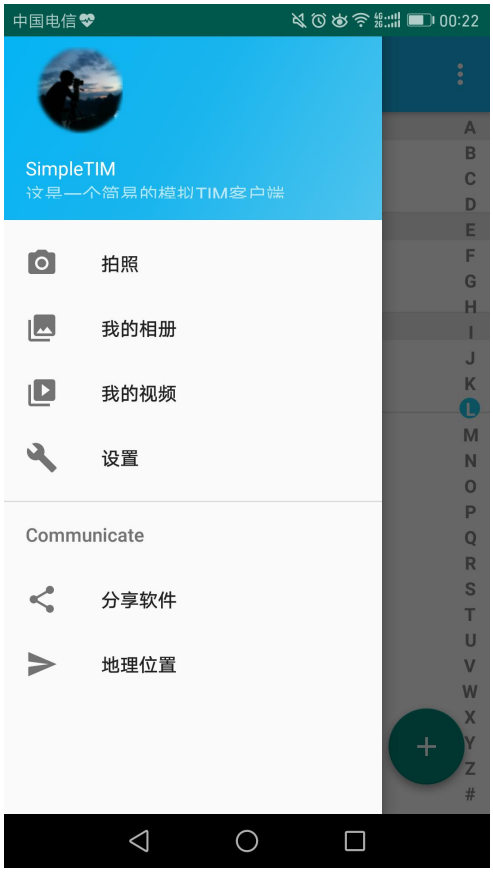
（一）登录界面：

（二）注册界面：

（三）主界面：


中国电信
00:22
好友列表
立即注册
安全退出
L
老王
M
马化腾
X
小明
写说说
看动态
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #

（四）侧拉界面：


中国电信
00:22
SimpleTIM
这是一个简易的模拟TIM客户端
拍照
我的相册
我的视频
设置
Communicate
分享软件
地理位置
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #

五、代码设计

（一）登录界面布局代码 activity_login.xml：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/rootViewLogin"
    android:background="@drawable/loginbg"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    tools:context="com.example.zdm.simpletim.LoginActivity">

    <!-- Login progress  进度条:  progressBarStyleLarge （大圆形）-->
    <ProgressBar
        android:id="@+id/login_progress"
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:visibility="gone"/>

    <!--ScrollView 滚动视图是指当拥有很多内容，屏幕显示不完时，需要通过滚动跳来显示的视图。ScrollView 只支持垂直滚动，只接受一个子元素。-->
    <ScrollView
        android:id="@+id/login_form"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="10">

        <LinearLayout
            android:id="@+id/email_login_form"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
            android:orientation="vertical">

            <!--模拟头像显示-->
            <ImageView
                android:id="@+id/symbol"
                android:layout_width="70sp"
                android:layout_height="70sp"
                android:layout_marginTop="60sp"
                android:background="@drawable/symbol"
                />

            <!--账号框-->
            <android.support.design.widget.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="60sp"
                android:layout_marginLeft="100sp"
                android:layout_marginRight="100sp">

                <AutoCompleteTextView
                    android:id="@+id/email"
                    android:layout_width="match_parent"
```

```
                android:layout_height="wrap_content"
                android:gravity="center_horizontal"
                android:hint="@string/prompt_email"
                android:inputType="textEmailAddress"
                android:maxLines="1"
                android:singleLine="true" />

        </android.support.design.widget.TextInputLayout>

        <!--密码框-->
        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10sp"
            android:layout_marginLeft="100sp"
            android:layout_marginRight="100sp">

            <EditText
                android:id="@+id/password"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center_horizontal"
                android:hint="@string/prompt_password"
                android:imeActionLabel="@string/action_sign_in_short"
                android:imeOptions="actionDone"
                android:inputType="textPassword"
                android:maxLines="1"
                android:singleLine="true" />

            <!--
            android:imeActionId 设置 IME 动作 ID，在 onEditorAction 中捕获判断进行逻辑操作。
            android:imeActionLabel  设置 IME 动作标签。但是不能保证一定会使用，猜想在输入法扩展的时候应该有用。
            android:imeOptions  设置软键盘的 Enter 键。
           -->

        </android.support.design.widget.TextInputLayout>

        <!--登录按钮-->
        <Button
            android:id="@+id/email_sign_in_button"
            style="?android:textAppearanceSmall"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="100sp"
            android:layout_marginRight="100sp"
            android:layout_marginTop="40sp"
            android:background="@drawable/btn_selector"
            android:text="@string/action_sign_in"
            android:textColor="#ffffff"
            android:textStyle="bold" />

    </LinearLayout>

</ScrollView>

<!--底部按钮-->
<LinearLayout
```

```
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:orientation="horizontal">

        <!--注册-->
        <Button
            android:id="@+id/regist"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:background="#00ffffff"
            android:gravity="center"
            android:text="@string/new_register"
            android:textColor="#468fba"
            android:textStyle="bold" />

        <Button
            android:id="@+id/nothing"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:background="#00ffffff"
            android:gravity="center"
            android:textColor="#454343"
            android:textStyle="bold" />

        <!--帮助-->
        <Button
            android:id="@+id/login_help"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:background="#00ffffff"
            android:gravity="center"
            android:text="@string/login_help"
            android:textColor="#468fba"
            android:textStyle="bold" />

    </LinearLayout>

</LinearLayout>

    <!--
        使用 TextInputLayout 必须手动引入 Design Support Library，在 build.gradle
中
       控件 AutoCompleteTextView 客户端保存搜索历史自动提示
        AutoCompleteTextView 常用属性

        android:completionHint              设置出现在下拉菜单中的提示标题
        android:completionThreshold             设置用户至少输入多少个字符才会显示
提示
        android:dropDownHorizontalOffset    下拉菜单于文本框之间的水平偏移。默认与
文本框左对齐
        android:dropDownHeight              下拉菜单的高度
        android:dropDownWidth               下拉菜单的宽度
        android:singleLine                  单行显示
        android:dropDownVerticalOffset      垂直偏移量
    -->
```

（二）LoginActivity：

```
package com.example.zdm.simpletim;

import android.content.Intent;
import android.os.Handler;
import android.os.Message;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.KeyEvent;
import android.view.View;
import android.view.inputmethod.EditorInfo;
import android.widget.AutoCompleteTextView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.example.zdm.simpletim.dao.PersonDAO;
import com.example.zdm.simpletim.utils.APPglobal;
import com.example.zdm.simpletim.viewUI.RegistActivity;

/**
 * User: ZDM
 * DateTime: 2019/10/14
 * Description: app 登录页面
 */
public class LoginActivity extends AppCompatActivity {

    private static boolean isExit=false;        // 判断是否直接退出程序
    private AutoCompleteTextView mEmailView;    // 用户名
    private EditText mPasswordView;             // 密码
    private int count = 0;                      // 判断按钮次数

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        mEmailView = (AutoCompleteTextView) findViewById(R.id.email);    // 获得用户名控件
        mPasswordView = (EditText) findViewById(R.id.password);          // 获得密码控件

        // 定义输入密码时软键盘的 enter 键功能(即 activity_login.xml 中设置的 IME 动作)
        // setOnEditorActionListener 这个方法在我们编辑完之后点击软键盘上的回车键才会触发
        mPasswordView.setOnEditorActionListener(new TextView.OnEditorActionListener() {
            @Override
            public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
                if(actionId == EditorInfo.IME_ACTION_DONE ) {
                    ++count;
                    if(count >= 3) {
                        //finish();
```

```
                        //System.exit(0);
                        Intent intent = new Intent(LoginActivity.this,
WelcomeActivity.class);
                        //传递退出所有 Activity 的 Tag 对应的布尔值为 true
                        intent.putExtra(WelcomeActivity.EXIST, true);
                        //启动 WelcomeActivity
                        startActivity(intent);
                    }
                    attemptLogin();    // 调用函数检查登陆信息是否合法
                  return true;
                }
                return false;
            }
        });

        // 登录按钮
       Button mEmailSignInButton = (Button)
findViewById(R.id.email_sign_in_button);
        mEmailSignInButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ++count;
                if(count >= 3) {
                    //finish();
                    //System.exit(0);
                    Intent intent = new Intent(LoginActivity.this,
WelcomeActivity.class);
                    //传递退出所有 Activity 的 Tag 对应的布尔值为 true
                    intent.putExtra(WelcomeActivity.EXIST, true);
                    //启动 WelcomeActivity
                    startActivity(intent);
                }
                attemptLogin();    // 调用函数检查登陆信息是否合法
           }
        });

        // 注册按钮
       Button mRegisterButton = (Button) findViewById(R.id.regist);
        mRegisterButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent=new Intent();
                intent.setClass(LoginActivity.this,RegistActivity.class);    //
跳转到注册页面
              LoginActivity.this.startActivity(intent);
            }
        });

        // 无法登陆按钮
       Button mLoginHelpButton = (Button) findViewById(R.id.login_help);
        mLoginHelpButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(LoginActivity.this, "未完成",
Toast.LENGTH_SHORT).show();
            }
        });

    }
```

```
    Handler handler = new Handler(){
        @Override
        public void handleMessage(Message msg) {
            super.handleMessage(msg);
            isExit=false;
        }
    };

    /**
     * 登录信息确认
     */
    private void attemptLogin() {
        // 初始化错误信息为 null
        mEmailView.setError(null);
        mPasswordView.setError(null);

        // 获取输入信息.
        String email = mEmailView.getText().toString();
        String password = mPasswordView.getText().toString();

        boolean cancel = false;   // 是否是非法信息
       View focusView = null;

        // 检查密码是否有效
       if ( (!TextUtils.isEmpty(password) && !isPasswordValid(password)) ||
TextUtils.isEmpty(password) ) {
            mPasswordView.setError(getString(R.string.error_invalid_password));
            focusView = mPasswordView;
            cancel = true;
        }

        // 检查邮箱
       if ( TextUtils.isEmpty(email) ) {
            mEmailView.setError(getString(R.string.error_field_required));
            focusView = mEmailView;
            cancel = true;
        }

        if ( cancel ) {//非法信息
           focusView.requestFocus();//标签用于指定屏幕内的焦点 View。
       }
        else { //合法信息
           // 登录跳转逻辑
           PersonDAO personDAO = new PersonDAO();
            boolean success = personDAO.chechLogin(email,password);
            if(success) { // 信息合法
               APPglobal.NAME = PersonDAO.findNameByUsername(email);   // 根据
账号找到当前账号真实名字
               APPglobal.USERNAME = email;
                Intent intent=new Intent();
                intent.setClass(LoginActivity.this,MainActivity.class);   //
跳转到 app 主页面
               LoginActivity.this.startActivity(intent);
            }
            else {
                Toast.makeText(LoginActivity.this, "用户名或密码错误",
Toast.LENGTH_SHORT).show();
            }
```

```
        }
    }

    /**
     * 密码是否和非法，至少需要4位
     * @param password
     * @return
     */
    private boolean isPasswordValid(String password) {
        return password.length() >= 4;
    }
}
```

（三）注册界面布局代码 activity_regist.xml：：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/rootViewRegist"
    android:background="@drawable/registbg"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    tools:context=".viewUI.RegistActivity">

    <ScrollView
        android:id="@+id/regist_form"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:id="@+id/email_regist_form"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
            android:orientation="vertical">

            <!--模拟 logo 显示-->
            <ImageView
                android:id="@+id/logo"
                android:layout_width="100sp"
                android:layout_height="50sp"
                android:layout_marginTop="100sp"
                android:background="@drawable/timlogo"
                />

            <!--账号-->
            <android.support.design.widget.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="30sp"
                android:layout_marginLeft="100sp"
                android:layout_marginRight="100sp">

                <AutoCompleteTextView
                    android:id="@+id/email"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
```

```
            android:gravity="center_horizontal"
            android:hint="@string/prompt_email"
            android:inputType="textEmailAddress"
            android:maxLines="1"
            android:singleLine="true"/>

    </android.support.design.widget.TextInputLayout>

    <!--真实姓名-->
    <android.support.design.widget.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="6sp"
        android:layout_marginLeft="100sp"
        android:layout_marginRight="100sp">

        <EditText
            android:id="@+id/rename"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
            android:hint="@string/real_name"
            android:imeOptions="actionUnspecified"
            android:inputType="text"
            android:maxLines="1"
            android:singleLine="true"/>

    </android.support.design.widget.TextInputLayout>

    <!--密码-->
    <android.support.design.widget.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="6sp"
        android:layout_marginLeft="100sp"
        android:layout_marginRight="100sp">

        <EditText
            android:id="@+id/password"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
            android:hint="@string/prompt_password"
            android:imeOptions="actionUnspecified"
            android:inputType="textPassword"
            android:maxLines="1"
            android:singleLine="true"/>

    </android.support.design.widget.TextInputLayout>

    <!--确认密码-->
    <android.support.design.widget.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="6sp"
        android:layout_marginLeft="100sp"
        android:layout_marginRight="100sp">

        <EditText
            android:id="@+id/repassword"
```

```
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:gravity="center_horizontal"
                    android:hint="@string/confirm_password"
                    android:imeActionLabel="@string/action_regist_short"
                    android:imeOptions="actionDone"
                    android:inputType="textPassword"
                    android:maxLines="1"
                    android:singleLine="true"/>

            </android.support.design.widget.TextInputLayout>


            <Button
                android:id="@+id/email_sign_in_button"
                style="?android:textAppearanceSmall"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginLeft="120sp"
                android:layout_marginRight="120sp"
                android:layout_marginTop="40sp"
                android:background="@drawable/btn_selector"
                android:textColor="#ffffff"
                android:text="@string/action_regist"
                android:textStyle="bold"/>

        </LinearLayout>

    </ScrollView>

</LinearLayout>
```

（四）RegistActivity:

```
package com.example.zdm.simpletim.viewUI;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.KeyEvent;
import android.view.View;
import android.view.inputmethod.EditorInfo;
import android.widget.AutoCompleteTextView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.example.zdm.simpletim.LoginActivity;
import com.example.zdm.simpletim.R;
import com.example.zdm.simpletim.dao.PersonDAO;
import com.example.zdm.simpletim.model.Person;

/**
 * User: ZDM
 * DateTime: 2019/10/14
 * Description: 注册界面
 */
```

```
public class RegistActivity extends AppCompatActivity {

    private AutoCompleteTextView mEmailView;  //用户名
    private EditText rename;                  //真实姓名
    private EditText mPasswordView;           //密码
    private EditText repassword;              //确认密码

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_regist);

        mEmailView = (AutoCompleteTextView) findViewById(R.id.email);//查找用
户名控件
        rename=(EditText) findViewById(R.id.rename);                  //真实姓名
控件
        mPasswordView = (EditText) findViewById(R.id.password);       //查找密码
控件
        repassword=(EditText) findViewById(R.id.repassword);          //重复密码
控件

        // 定义输入密码时软键盘的 enter 键功能(即 activity_login.xml 中设置的 IME 动
作)
        // setOnEditorActionListener 这个方法在我们编辑完之后点击软键盘上的回车键
才会触发
        repassword.setOnEditorActionListener(new
TextView.OnEditorActionListener() {
            @Override
            public boolean onEditorAction(TextView v, int actionId, KeyEvent
event) {
                if(actionId == EditorInfo.IME_ACTION_DONE ) {
                    attemptLogin();   // 调用函数检查登陆信息是否合法
                  return true;
                }
                return false;
            }
        });

        // 注册按钮
        Button mEmailSignInButton = (Button)
findViewById(R.id.email_sign_in_button);
        mEmailSignInButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                attemptLogin();   // 调用函数检查登陆信息是否合法
            }
        });

    }

    /**
     * 注册信息确认
     */
    private void attemptLogin() {
        // 初始化控件错误信息
        mEmailView.setError(null);
        mPasswordView.setError(null);
        repassword.setError(null);
```

```
        // 获取输入信息
        String email = mEmailView.getText().toString();
        String mname=rename.getText().toString();
        String password = mPasswordView.getText().toString();
        String mrepassword = repassword.getText().toString();

        boolean cancel = false;   // 是否是非法信息
        View focusView = null;

        // 检查密码是否有效
        if ( (!TextUtils.isEmpty(password) && !isPasswordValid(password)) ||
TextUtils.isEmpty(password) ) {
            mPasswordView.setError(getString(R.string.error_invalid_password));
            focusView = mPasswordView;
            cancel = true;
        }

        // 检查两次密码是否相等
        if(!mrepassword.equals(password)){
            repassword.setError("两次密码不一致");
            focusView = repassword;
            cancel = true;
        }

        // 检查邮箱
        if ( TextUtils.isEmpty(email) ) {
            mEmailView.setError(getString(R.string.error_field_required));
            focusView = mEmailView;
            cancel = true;
        }

        //检查真实姓名
        if(mname.equals("")){
            rename.setError("真实姓名不能为空");
            focusView = rename;
            cancel = true;
        }

        // 注册逻辑实现
        if ( cancel ) {
            focusView.requestFocus();
        }
        else {
            String mrename=rename.getText().toString();
            String memail=mEmailView.getText().toString();
            String mpassword=repassword.getText().toString();
            Person person=new Person(mrename,memail,mpassword);

            PersonDAO personDAO=new PersonDAO();
            boolean isSuccess=false;

            if(personDAO.checkUsername(memail)) { // 用户已存在
               Toast.makeText(RegistActivity.this, "用户名已存在，请重新输入",
Toast.LENGTH_SHORT).show();
            }
            else {
                isSuccess= personDAO.insert(person); // 添加到数据库
              if(isSuccess) { // 成功添加到数据库
```

```
                    Toast.makeText(RegistActivity.this, "注册成功，请登录",
Toast.LENGTH_SHORT).show();
                      Intent intent=new Intent();
                      intent.setClass(RegistActivity.this,LoginActivity.class);
// 转到登陆
                    RegistActivity.this.startActivity(intent);
                }
                else {
                    Toast.makeText(RegistActivity.this, "信息不合法，请确认输入",
Toast.LENGTH_SHORT).show();
                }
            }
        }

    }

    /**
     * 密码是否和非法，至少需要4位
     * @param password
     * @return
     */
    private boolean isPasswordValid(String password) {
        return password.length() >= 4;
    }
}
```

（五）button 的处理：

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <!-- 圆角的半径 -->
    <corners android:radius="40dp"/>

    <!-- 填充颜色 -->
    <solid android:color="#3686ff"/>

</shape>
<!--正常按钮样式-->
```

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <!-- 圆角的半径 -->
    <corners android:radius="40dp"/>

    <!-- 填充颜色 -->
    <solid android:color="#0662f5"/>

</shape>
<!--按下按钮样式-->
```

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
```

```
    <!-- 正常状态 -->
    <item android:drawable="@drawable/btn_normal"
android:state_pressed="false"/>

    <!-- 按下状态 -->
    <item android:drawable="@drawable/btn_pressed"
android:state_pressed="true"/>

</selector>
<!--选择器，根据按钮按下或抬起来判断使用样式-->
```

（六）国际化设置：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">SimpleTIM</string>
    <string name="top_lable_name">好友列表</string>
    <string name="action_regist_now">立即注册</string>
    <string name="action_settings">安全退出</string>
    <string name="title_activity_login">账号登录</string>
    <string name="title_activity_regist">账号注册</string>
    <string name="description">这是一个简易的模拟 TIM 客户端</string>
    <string name="navigation_drawer_open">Open navigation drawer</string>
    <string name="navigation_drawer_close">Close navigation drawer</string>
    <!--LoginActivity-->
    <string name="new_register">新用户注册</string>
    <string name="login_help">无法登录?</string>
    <string name="prompt_email">QQ 号/手机/邮箱</string>
    <string name="prompt_password">密码</string>
    <string name="confirm_password">确认密码</string>
    <string name="real_name">真实姓名</string>
    <string name="action_sign_in">登录</string>
    <string name="action_sign_in_short">Sign</string>
    <string name="action_regist">注册</string>
    <string name="action_regist_short">regist</string>
    <string name="error_invalid_email">不是有效的账号</string>
    <string name="error_invalid_password">密码太短,至少输入 4 位</string>
    <string name="error_incorrect_password">密码无效</string>
    <string name="error_field_required">用户名无效</string>
</resources>
```

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">SimpleTIM</string>
    <string name="top_lable_name">friends list</string>
    <string name="action_regist_now">Register now</string>
    <string name="action_settings">Safety exit</string>
    <string name="title_activity_login">Account login</string>
    <string name="title_activity_regist">Account regist</string>
    <string name="description">这是一个简易的模拟 TIM 客户端</string>
    <string name="navigation_drawer_open">Open navigation drawer</string>
    <string name="navigation_drawer_close">Close navigation drawer</string>
    <!--LoginActivity-->
    <string name="new_register">new regist</string>
    <string name="login_help">help?</string>
    <string name="prompt_email">QQ 号/mobile/email</string>
    <string name="prompt_password">password</string>
```

```
    <string name="confirm_password">confirm password</string>
    <string name="real_name">real name</string>
    <string name="action_sign_in">login</string>
    <string name="action_sign_in_short">Sign</string>
    <string name="action_regist">regist</string>
    <string name="action_regist_short">regist</string>
    <string name="error_invalid_email">Not a valid account</string>
    <string name="error_invalid_password">Password too short, at least 4
digits</string>
    <string name="error_incorrect_password">Invalid password</string>
    <string name="error_field_required">Invalid user name</string>
</resources>
```

（七）模型 Person 类：

```
package com.example.zdm.simpletim.model;

import com.example.zdm.simpletim.utils.PinYinUtils;

/**
 * User: ZDM
 * DateTime: 2019/10/14
 * Description:好友实体类
 */
public class Person {

    private String name;       // 名字
    private String pinyin;     // 拼音
    private String headerChar; // 拼音首字母

    private String username;   // 用户名
    private String password;   // 密码

    public Person(){};

    public Person(String name) {
        this.name = name;
        this.password="";
        this.username="";
        this.pinyin = PinYinUtils.getPinyin(name);
        headerChar = pinyin.substring(0, 1);
    }

    public Person(String name, String username, String password) {
        this.name = name;
        this.username = username;
        this.password = password;
        this.pinyin = PinYinUtils.getPinyin(name);
        headerChar = pinyin.substring(0, 1);
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setUsername(String username) {
        this.username = username;
    }
```

```
    public void setPassword(String password) {
        this.password = password;
    }

    public String getName() {
        return name;
    }

    public String getPinyin() {
        return pinyin;
    }

    public String getHeaderChar() {
        return headerChar;
    }

    public String getUsername() {
        return username;
    }

    public String getPassword() {
        return password;
    }
}
```

（八）中间层 PersonDAO 类：

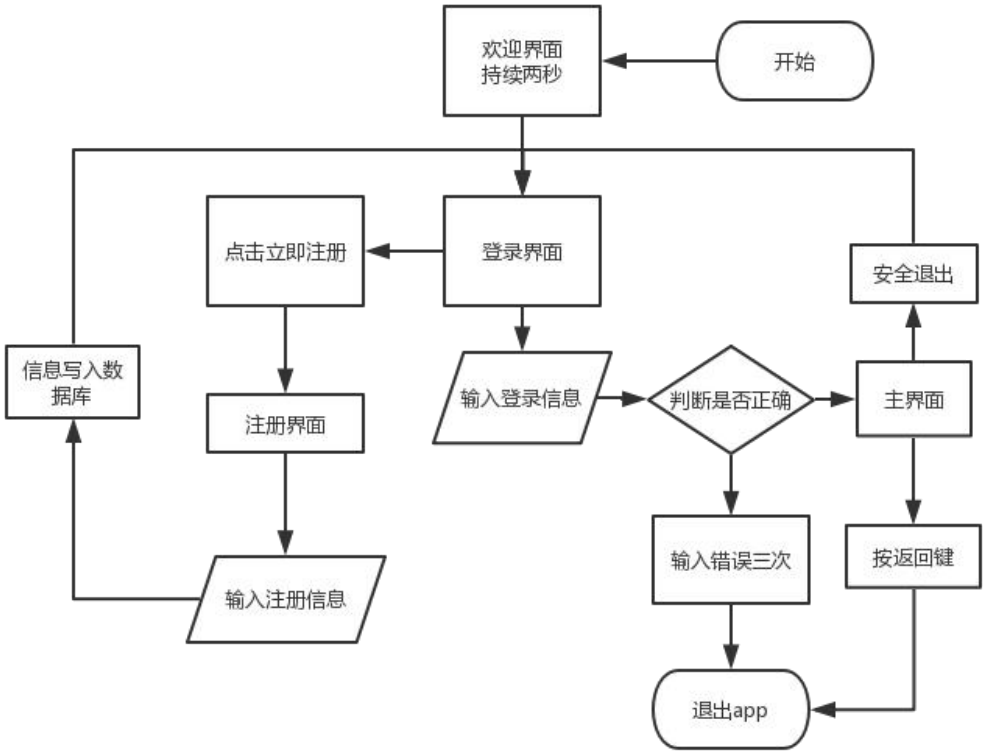
```
package com.example.zdm.simpletim.dao;

import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

import com.example.zdm.simpletim.db.DatabaseHelper;
import com.example.zdm.simpletim.model.Person;

import java.util.ArrayList;

/**
 * User: ZDM
 * DateTime: 2019/10/14
 * Description: 联系人数据访问对象
 */
public class PersonDAO {

    private static ArrayList<Person> personList = null;   // 保存联系人数据

    // 获取所有联系人
    public static ArrayList<Person> getPersonList() {
        if(null == personList) {

            // 给 PersonDAO 类进行加锁，则每个 PersonDAO 类都进行同步
            synchronized (PersonDAO.class) {
                if(null==personList){
                    personList=new ArrayList<Person>();
                }
            }

            // 把数据从数据库中拿出来(Android 查询数据是通过 Cursor 类来实现)
            DatabaseHelper databaseHelper = new DatabaseHelper();   // 创建数
```

```
据库
            SQLiteDatabase db = databaseHelper.getConnect();         // 获得连接
            Cursor cursor = db.query("tb_person", null, null, null, null, null,
null);    // 使用 SQLiteDatabase.query()方法时，就会得到 Cursor 对象， Cursor 所指向
的就是每一条数据

            // 一行行读数据
            while(cursor.moveToNext()) {
                 // 返回指定列的名称
                int namenum=cursor.getColumnIndex("name");
                 int usernamenum=cursor.getColumnIndex("username");
                 int passwordnum=cursor.getColumnIndex("password");
                 // 根据名称取数据
                String name=cursor.getString(namenum);
                 String username=cursor.getString(usernamenum);
                 String password=cursor.getString(passwordnum);
                 // 把联系人放入列表
                Person person=new Person(name,username,password);
                 personList.add(person);
                 //cursor.moveToNext();
             }
        }

        return personList;
    }

    // 根据用户名查找用户真实姓名
    public static String findNameByUsername(String username){
        if(null == personList){
            getPersonList();
        }
        for ( int i = 0; i < personList.size(); i++ ) {
            Person person = personList.get(i);
            if(username.equals(person.getUsername())){
                return person.getName();
            }
        }
        return "";
    }

    // 查找用户是否存在
    public boolean checkUsername(String username){
        if(null == personList){
            getPersonList();
        }
        for ( int i = 0; i < personList.size(); i++ ) {
            Person book = personList.get(i);
            if(username.equals(book.getUsername())){
                return true;
            }
        }
        return false;
    }

    // 插入联系人
    public boolean insert(Person person){

        if(checkUsername(person.getUsername())){  //如果用户名存在则直接返回失败
            return false;
```

```
        }
        try {
            personList.add(person);
            DatabaseHelper conn = new DatabaseHelper();
            SQLiteDatabase db = conn.getConnect();
            String sql="insert into tb_person(name,username,password)
values('"

+person.getName()+"','"+person.getUsername()+"','"+person.getPassword()+"')";
            db.execSQL(sql);
            db.close();
            return true;
        }catch ( Exception e ){
            return false;
        }

    }

    // 检查登录
    public boolean chechLogin(String username,String password){
        if(null==personList){
            getPersonList();
        }
        for ( int i = 0; i < personList.size(); i++ ) {
            Person book=personList.get(i);
            if(username.equals(book.getUsername()) &&
password.equals(book.getPassword())){
                return true;
            }
        }
        return false;
    }
}

/**
 * Cursor 是每行的集合。
 * 使用 moveToFirst() 定位第一行。
 * 你必须知道每一列的名称。
 * 你必须知道每一列的数据类型。
 * Cursor 是一个随机的数据源。
 * 所有的数据都是通过下标取得
 */
```

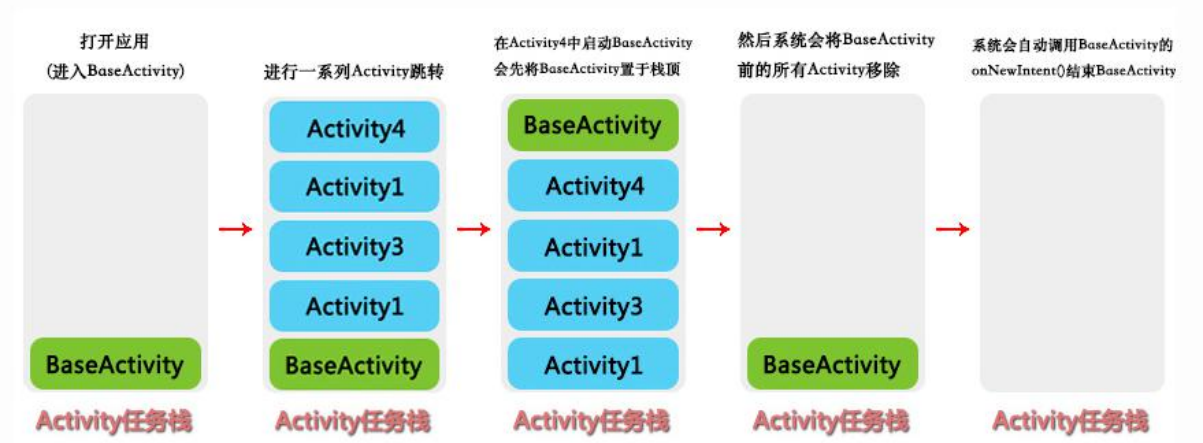
六、软件流程

①工作流程



②安全设置处理

把 WelcomeActivity 模式设置成 simgleTask（即图中 BaseActivity），然后重载 onNewIntent()



代码实现：

第一步：设置 WelcomeActivity 的启动模式为 singleTask

```
android:launchMode="singleTask"
```

第二步：重写 WelcomeActivity 的 onNewIntent()方法

```
//声明一个静态常量，用作退出 BaseActivity 的 Tag
public static final String EXIST = "exist";
@Override
protected void onNewIntent(Intent intent) {
```

```
    super.onNewIntent(intent);
    if (intent != null) {//判断其他 Activity 启动本 Activity 时传递来的 intent 是否为空
        //获取 intent 中对应 Tag 的布尔值
         boolean isExist = intent.getBooleanExtra(EXIST, false);
        //如果为真则退出本 Activity
         if (isExist) {
            this.finish();
        }
    }
}
```

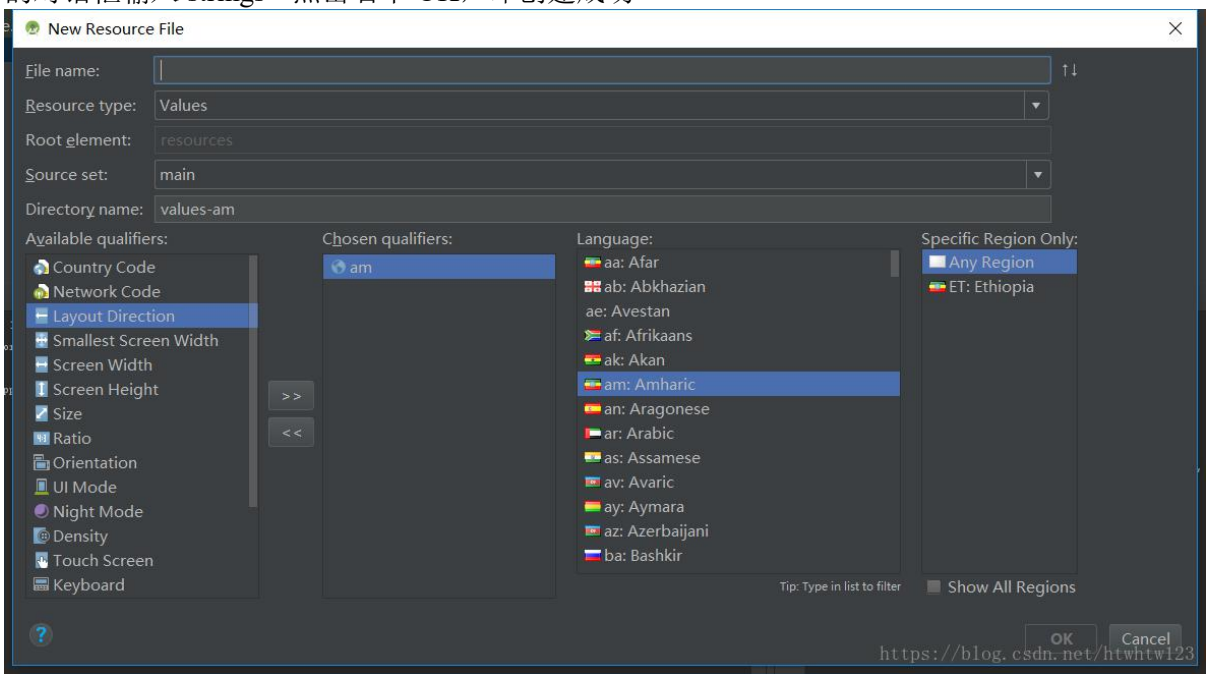
第三步：在需要做出退出应用动作的 LoginActivity 中启动 WelcomeActivity

```
if(登录错误次数大于三次){
    Intent intent = new Intent(LoginActivity.this,BaseActivity.class);
    //传递退出所有 Activity 的 Tag 对应的布尔值为 true
    intent.putExtra(WelcomeActivity.EXIST, true);
    //启动 BaseActivity
    startActivity(intent);
}
```

③国际化处理

方法：使用 Android Studio3.0.1 的工具

左侧文件资源树->android 视图或 Project 视图都可以->new ->Android resource file->点击弹出框左下框中的 Local->点击弹出框中部“>>”符号->直接选目标语言了（如下图）->最上面的对话框输入 strings->点击右下 OK，即创建成功



七、难点（或遇到的问题）和解决方案

这次设计的难点主要在于如何与数据库进行交互以及如何做到安全设置直接退出app，以及主界面的布局。

1.登陆注册布局：布局上，要让布局适配升起的软键盘
解决方式：使用滚动视图 ScrollView

2.主界面：实现侧滑抽屉布局以及侧边栏。
解决方案：本人参考了以下的教程
侧滑界面：https://blog.csdn.net/dzjin1234/article/details/79008269
字母导航栏：https://blog.csdn.net/qq_30379689/article/details/52684312
pinyin4j 包：https://blog.csdn.net/sky_limitless/article/details/79443540

3.链接数据库：SQLite 的使用
解决方案：参考网上的 demo 并在自己的代码中实践
SQLite 的使用：https://www.jianshu.com/p/7d098ef952d7

4.安全设置：输入三次错误后直接退出 app
解决方案：把最初的 WelcomeActivity 设置成 simgleTask，重写 onNewIntent()
参考：https://www.cnblogs.com/caobotao/p/5127645.html

5.app 国际化：适配中英双语
解决方案：新建两个 string.xml，分别设置不同的国家代码
参考：https://www.cnblogs.com/caobotao/p/5127645.html

八、不足之处

1.无法修改密码和用户信息

2.密码存储使用的是明文，不安全

3.注册号码并没有进行限制，应该使用正则匹配进行相应的限制

4.登录界面最下方的两个按钮会被弹起的软件盘抬起来，可进行相应的改善

5.在主界面按退出应该直接退出 app，而不是退回到登录界面

九、今后的设想

这次的模拟 TIM 的登陆注册 demo 的开发，让我学习到如何使用 TextInputLayout 组件进行登陆注册界面的设计与排布，同时也知道了如何使用选择器 selector 来改变按下按钮的样式与正常按钮的样式。最重要的，在这次开发中，我由于不想单纯的把账户信息存储在一个本地文件中，我尝试着使用了 SQLite 这个 Android 自带的轻型数据库，在这样一个小小的挑战自我中，我巩固加深了之前不太熟练的软件框架设计，设计出了一个从 UI 到代码逻辑层到数据库处理的较完整的 demo。同时，我为了美观，也尝试着跟着网络的教程模仿设计了 Android 最基本的带有侧滑抽屉界面的主页面，这也让我初次体会到了设计一个看着简洁美观的主页面的难度。

在这次 demo 的编写中我虽然成功的挑战了自己，但同时，我也明白这一个 demo 并不能算自己一个人写出来的，我借鉴学习了网上许多前辈的思路，才最终完成了这个 demo。下一步我的目标将是熟读并好好理解这个简单的 demo，并且继续在这个 demo 上加入基本功能，把我的简单的登陆注册 demo 变成为更加类似 TIM 社交客户端的复杂 demo，同时，在下一次编写复杂 demo 的时候，尽量做到自己一个人完成所有的功能逻辑模块的设计与开发。

在这次 android studio 的学习以及代码的编写中我也感觉自己的能力又有了一些提升，不过未来的路也还很长，我还需要更努力地学习！