

# 物联网校园气象站

## 项目计划书

|  |  |  |
| --- | --- | --- |
| 文件状态：  　[　]草稿  　[　]正式发布  　[√]正在修改 | 文件标识： | G08-WB-Plan |
| 当前版本： | 1.0 |
| 作　　者： | 郑楠 |
| 完成日期： | 2016-04-04 |

# 版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| 1.0 | 郑楠 | 郑楠 | 2016-05-29  至  2016-05-29 | 起草代码清单 |

目录

[物联网校园气象站 1](#_Toc452288524)

[项目计划书 1](#_Toc452288525)

[版 本 历 史 2](#_Toc452288526)

[1. 引言 4](#_Toc452288527)

[1.1. 编写目的 4](#_Toc452288528)

[1.2. 背景 4](#_Toc452288529)

[1.2.1. 项目名称 4](#_Toc452288530)

[1.2.2. 项目委托单位 4](#_Toc452288531)

[1.2.3. 项目的用户 4](#_Toc452288532)

[1.2.4. 任务提出者 4](#_Toc452288533)

[1.2.5. 项目主要承担部门 5](#_Toc452288534)

[1.2.6. 项目建设背景 5](#_Toc452288535)

[1.3. 定义 5](#_Toc452288536)

[1.4. 参考资料 5](#_Toc452288537)

[2. 包结构 6](#_Toc452288538)

[2.1. display子包 6](#_Toc452288539)

[2.1.1. control 6](#_Toc452288543)

[2.1.2. intrfc 24](#_Toc452288544)

[2.1.3. model 25](#_Toc452288545)

[2.1.4. ui 26](#_Toc452288546)

[2.1.5. util 37](#_Toc452288547)

[2.1.6. DisplayHistoryActivity.java 45](#_Toc452288548)

[2.2. registandlogin 46](#_Toc452288549)

[2.2.1. control 46](#_Toc452288550)

[2.2.2. intrfc 52](#_Toc452288551)

[2.2.3. model 53](#_Toc452288552)

[2.2.4. ui 54](#_Toc452288553)

[2.2.5. util 57](#_Toc452288554)

[2.2.6. RegistAndLoginActivity.java 59](#_Toc452288555)

[2.3. MainActivity.java 60](#_Toc452288556)

# 引言

## 编写目的

为了保证项目团队按时保质地完成项目目标，便于项目团队成员更好地了解项目情况，

使项目工作开展的各个过程合理有序，因此以文件化的形式，把对于在项目生命周期内的工

作任务范围、各项工作的任务分解、项目团队组织结构、各团队成员的工作责任、团队内外

沟通协作方式、开发进度、经费预算、项目内外环境条件、风险对策等内容做出的安排以书

面的方式，作为项目团队成员以及项目干系人之间的共识与约定，项目生命周期内的所有项

目活动的行动基础，项目团队开展和检查项目工作的依据。 (邱, 冯, 胡, 郑, & 周, 2013-05-09)

## 背景

### 项目名称

项目名称：物联网气象站原型

项目代号：WeatherBase by Group 08

英文简称：WB

### 项目委托单位

浙江大学城市学院机器人与智慧工厂实验室

浙江大学城市学院Google-ARM移动科技实验室

### 项目的用户

用户范围：浙江大学城市学院

用户群体：全体学生

### 任务提出者

浙江大学城市学院Google-ARM移动科技实验室

### 项目主要承担部门

浙江大学城市学院软件工程14级G08小组

### 项目建设背景

随着科技的发展，物联网慢慢走近人们的工作与生活当中。越来越多的人们发现他们需要的是远端监控传感器等设备的运行情况，而不是花费人力物力到实际场地中，只为了单纯地获取数据。

物联网被世界公认为是继计算机、互联网与移动通信网之后的世界信息产业第三次浪潮。他是以感知为前提，实现人与人、人与物、物与物全面互联的网络。 (ewt, 2014-01-07)

在物联网概念的基础上，更有被称作物联网云平台的产品诞生，利用这些云平台，人们可以更加方便的利用物联网技术，把重心放在功能的设计而不是后台的编写上，并且即使是学生层次，也能够方便地利用云平台提供的API实现头脑中的奇思妙想。

杭州气温变化大，极端情况可能一天就走过了一个四季，所以很出门前最好能了解身边的环境变化，为此我们准备做一个小型的校园气象站，针对学生日常所关注的指数做一个详细的汇报，切实的汇报我们身边的气候，并给予有效的出门建议。本项目可能会与广播站等多平台联合，力求让更多的人了解到身边的气候变化。

## 定义

表格1术语定义表

|  |  |
| --- | --- |
| 物联网 | 物联网是新一代信息技术的重要组成部分，也是“信息化”时代的重要发展阶段。其英文名称是：“Internet of things（IoT）”。顾名思义，物联网就是物物相连的互联网。 |
| 气象站 | 实时监测温度、湿度、风速、风向、雨量、气压、紫外辐射、噪声、粉尘等多种气象参数的一整套硬件设备。 |
| 气象站应用软件 | 监听获取气象站硬件设备传递回的数据并以友好的UI界面与用户进行交互的软件。 |
| PTC-ThingWorx | ThingWorx是市场领先的IOT平台提供商，现已被PTC公司收购。它允许开发者快速地连接他们的设备，创建、删除应用以及对“物”的分析。 |
| ArduinoYun | Arduino Yun 是一款基于ATmega32U4 和Atheros AR9331 的单片机板。 Atheros AR9331 可以运行一个基于Linux 和OpenWRT 的操作系统 Linino，这给Yun带来前所未有的网络能力。并且由于使用了LINUX，可以自由的选择你需要的通信接口. |
| Android | Android是一种基于Linux的自由及开放源代码的操作系统，主要使用于移动设备，如智能手机和平板电脑，由Google公司和开放手机联盟领导及开发。 |
| 酷热指数 | 是一种综合空气温度和相对湿度来确定体感温度的指数──即真正感受到的热度。 |

## 参考资料

1. 百度百科《物联网》《Android》《气象站》
2. PTC-ThingWorx官方网站  
   <http://www.thingworx.com/>
3. Arduino中文社区 arduino YUN 中文资料

<http://www.arduino.cn/thread-4208-1-1.html>

1. 百度百科

http://baike.baidu.com/link?url=1Gn98GXBXzsanl5d31XYW8-XkeGI4st-fv9hp5clg6v0B-1Fe8fhqHKSoufMMqZrEti4Q5Bq4uVqGrnRmATxpq

# 包结构

display

registandlogin

## display子包

control

intrfc

model

ui

util



### control

#### RequestDay.java

package ubibots.zuccweatherbase.display.control;

import android.os.AsyncTask;

import android.os.Handler;

import android.os.Message;

import android.widget.Toast;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Timer;

import java.util.TimerTask;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

import ubibots.zuccweatherbase.display.model.BeanConstant;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

import ubibots.zuccweatherbase.display.ui.DayView;

import ubibots.zuccweatherbase.display.util.RequestUtil;

public class RequestDay {

private static RequestDayHandler requestDayHandler;

public RequestDay(){

requestDayHandler = new RequestDayHandler();

}

public void executeRequest(){

DayView.setDay(new BeanTabMessage(new ArrayList<>(), new ArrayList<>(), new ArrayList<>()));

Calendar dayCalendar = Calendar.getInstance();

dayCalendar.set(Calendar.SECOND, dayCalendar.get(Calendar.SECOND) - BeanConstant.delayDay / 1000 \* (RequestDayHistory.MAX - 1));

for (int i = 0; i < RequestDayHistory.MAX; i++) {

DayView.getDay().getTemperature().add(0.0);

DayView.getDay().getHumidity().add(0.0);

DayView.getDay().getDate().add("");

dayHistory(DayView.getDay(), dayCalendar, i);

dayCalendar.set(Calendar.SECOND, dayCalendar.get(Calendar.SECOND) + BeanConstant.delayDay / 1000);

}

Toast.makeText(DisplayHistoryActivity.getContext(), "正在获取数据中,请耐心等待...",

Toast.LENGTH\_LONG).show();

}

public static void dayHistory(BeanTabMessage day, Calendar calendar, int id) {

String strUrl = RequestUtil.combineUrl((Calendar) calendar.clone());

RequestDayHistory request = new RequestDayHistory(day, id, 0);

request.executeOnExecutor(AsyncTask.THREAD\_POOL\_EXECUTOR, strUrl);

}

private static Timer requestDayTimer = new Timer();

private static TimerTask requestDayTask = new TimerTask() {

@Override

public void run() {

// 需要做的事:发送消息

Message message = new Message();

message.what = 1;

requestDayHandler.sendMessage(message);

}

};

public static Timer getRequestDayTimer() {

return requestDayTimer;

}

public static TimerTask getRequestDayTask() {

return requestDayTask;

}

static class RequestDayHandler extends Handler {

public void handleMessage(Message msg) {

if (msg.what == 1) {

Calendar calendar = Calendar.getInstance();

dayStep(DayView.getDay(),calendar);

}

super.handleMessage(msg);

}

}

public static void dayStep(BeanTabMessage day, Calendar calendar) {

String strUrl = RequestUtil.combineUrl((Calendar) calendar.clone());

RequestDayStep request = new RequestDayStep(day, 0);

request.executeOnExecutor(AsyncTask.THREAD\_POOL\_EXECUTOR, strUrl);

}

}

#### RequestDayHistory.java

package ubibots.zuccweatherbase.display.control;

import android.os.AsyncTask;

import android.view.View;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Locale;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import ubibots.zuccweatherbase.display.model.BeanConstant;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

import ubibots.zuccweatherbase.display.ui.DayView;

import ubibots.zuccweatherbase.display.util.RequestUtil;

public class RequestDayHistory extends AsyncTask<String, Integer, String> {

public final static int MAX = 48;

private BeanTabMessage day;

private int id;

private String strURL;

private int time;

public RequestDayHistory(BeanTabMessage day, int id, int time) {

this.day = day;

this.id = id;

this.time = time;

}

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected String doInBackground(String... params) {

//System.out.println("Url: " + params[0]);

URL url;

try {

url = new URL(params[0]);

strURL = params[0];

HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();

urlConn.setDoInput(true); //允许输入流，即允许下载

urlConn.setDoOutput(true); //允许输出流，即允许上传

urlConn.setUseCaches(false); //不使用缓冲

urlConn.setRequestMethod("POST"); //使用get请求

InputStreamReader in = new InputStreamReader(urlConn.getInputStream());

BufferedReader bufferedReader = new BufferedReader(in);

String result = "";

String readLine;

while ((readLine = bufferedReader.readLine()) != null) {

result += readLine;

}

in.close();

urlConn.disconnect();

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(String result) {

if (result != null && time < BeanConstant.MAXTIME) {

Pattern pattern = Pattern.compile("<TD>(.\*?)</TD>");

Matcher matcher = pattern.matcher(result);

ArrayList<String> tmp = new ArrayList<>();

while (matcher.find()) {

tmp.add(matcher.group(1));

}

if (tmp.size() >= 3) {

String dateString = tmp.get(0);

double temp = Double.valueOf(tmp.get(1));

double humi = Double.valueOf(tmp.get(2));

//丢包重发

if (dateString.length() != 24 || temp <= 0 || humi <= 0) {

reconnect(strURL, day, id);

return;

}

dateString = dateString.substring(0, 10) + " " + dateString.substring(11, 23);

Calendar calendar = RequestUtil.dateToCalender(dateString, "yyyy-MM-dd HH:mm:ss.SSS");

calendar.set(Calendar.HOUR\_OF\_DAY, calendar.get(Calendar.HOUR\_OF\_DAY) + 8);

SimpleDateFormat sdf = new SimpleDateFormat("dd HH:mm", Locale.getDefault());

dateString = sdf.format(calendar.getTime());

day.getDate().set(id, dateString);

day.getTemperature().set(id, temp);

day.getHumidity().set(id, humi);

day.count++;

//历史数据收集完毕

if (day.count == MAX) {

//刷新界面

RequestUtil.reflashLineView(DayView.getDayBeanLineView(), day, "日 时:分");

RequestDay.getRequestDayTimer().schedule(RequestDay.getRequestDayTask(), BeanConstant.delayDay, BeanConstant.delayDay);

DayView.getDayProgressBar().setVisibility(View.GONE);

}

DayView.getDayProgressBar().setProgress(100 \* day.count / MAX);

System.out.println("Time: " + day.getDate().get(id) + " " + "Temperature: " + day.getTemperature().get(id) + " " + "Humidity: " + day.getHumidity().get(id) + " " + "Num: " + id + " " + "Count: " + day.count + " " + "Time: " + time);

} else {//丢包重发

reconnect(strURL, day, id);

}

} else {

RequestUtil.connectFailed();

}

}

//该方法运行在UI线程当中,并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPreExecute() {

}

public void reconnect(String strURL, BeanTabMessage day, int id) {

RequestHourHistory another = new RequestHourHistory(day, id, time + 1);

System.out.println("time: " + time);

System.out.println(strURL);

another.execute(strURL);

}

}

#### RequestDayStep.java

package ubibots.zuccweatherbase.display.control;

import android.os.AsyncTask;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Locale;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import ubibots.zuccweatherbase.display.model.BeanConstant;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

import ubibots.zuccweatherbase.display.ui.DayView;

import ubibots.zuccweatherbase.display.util.RequestUtil;

public class RequestDayStep extends AsyncTask<String, Integer, String> {

public final static int MAX = 48;

private BeanTabMessage day;

private String strURL;

private int time;

public RequestDayStep(BeanTabMessage day, int time) {

this.day = day;

this.time = time;

}

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected String doInBackground(String... params) {

//System.out.println("Url: " + params[0]);

URL url;

try {

url = new URL(params[0]);

strURL = params[0];

HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();

urlConn.setDoInput(true); //允许输入流，即允许下载

urlConn.setDoOutput(true); //允许输出流，即允许上传

urlConn.setUseCaches(false); //不使用缓冲

urlConn.setRequestMethod("POST"); //使用get请求

InputStreamReader in = new InputStreamReader(urlConn.getInputStream());

BufferedReader bufferedReader = new BufferedReader(in);

String result = "";

String readLine;

while ((readLine = bufferedReader.readLine()) != null) {

result += readLine;

}

in.close();

urlConn.disconnect();

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(String result) {

if (result != null && time < BeanConstant.MAXTIME) {

Pattern pattern = Pattern.compile("<TD>(.\*?)</TD>");

Matcher matcher = pattern.matcher(result);

ArrayList<String> tmp = new ArrayList<>();

while (matcher.find()) {

tmp.add(matcher.group(1));

}

if (tmp.size() >= 3) {

String dateString = tmp.get(0);

double temp = Double.valueOf(tmp.get(1));

double humi = Double.valueOf(tmp.get(2));

//丢包重发

if (dateString.length() != 24 || temp <= 0 || humi <= 0) {

reconnect(strURL, day);

return;

}

dateString = dateString.substring(0, 10) + " " + dateString.substring(11, 23);

Calendar calendar = RequestUtil.dateToCalender(dateString,"yyyy-MM-dd HH:mm:ss.SSS");

calendar.set(Calendar.HOUR\_OF\_DAY, calendar.get(Calendar.HOUR\_OF\_DAY) + 8);

SimpleDateFormat sdf = new SimpleDateFormat("dd HH:mm", Locale.getDefault());

dateString = sdf.format(calendar.getTime());

day.getDate().remove(0);

day.getDate().add(dateString);

day.getTemperature().remove(0);

day.getTemperature().add(temp);

day.getHumidity().remove(0);

day.getHumidity().add(humi);

//刷新界面

RequestUtil.reflashLineView(DayView.getDayBeanLineView(), day, "日 时:分");

System.out.println("Time: " + day.getDate().get(MAX - 1) + " " + "Temperature: " + day.getTemperature().get(MAX - 1) + " " + "Humidity: " + day.getHumidity().get(MAX - 1) + " " + "Time: " + time);

} else {//丢包重发

reconnect(strURL, day);

}

} else {

RequestUtil.connectFailed();

}

}

//该方法运行在UI线程当中,并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPreExecute() {

}

public void reconnect(String strURL, BeanTabMessage day) {

RequestDayStep another = new RequestDayStep(day, time + 1);

System.out.println("time: " + time);

System.out.println(strURL);

another.execute(strURL);

}

}

#### RequestHour.java

package ubibots.zuccweatherbase.display.control;

import android.os.AsyncTask;

import android.os.Handler;

import android.os.Message;

import android.widget.Toast;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Timer;

import java.util.TimerTask;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

import ubibots.zuccweatherbase.display.model.BeanConstant;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

import ubibots.zuccweatherbase.display.ui.HourView;

import ubibots.zuccweatherbase.display.util.RequestUtil;

public class RequestHour {

private static RequestHourHandler requestHourHandler;

public RequestHour(){

requestHourHandler = new RequestHourHandler();

}

public void executeRequest(){

HourView.setHour(new BeanTabMessage(new ArrayList<>(), new ArrayList<>(), new ArrayList<>()));

Calendar hourCalendar = Calendar.getInstance();

hourCalendar.set(Calendar.SECOND, hourCalendar.get(Calendar.SECOND) - BeanConstant.delayHour / 1000 \* (RequestHourHistory.MAX - 1));

for (int i = 0; i < RequestHourHistory.MAX; i++) {

HourView.getHour().getTemperature().add(0.0);

HourView.getHour().getHumidity().add(0.0);

HourView.getHour().getDate().add("");

hourHistory(HourView.getHour(), hourCalendar, i);

hourCalendar.set(Calendar.SECOND, hourCalendar.get(Calendar.SECOND) + BeanConstant.delayHour / 1000);

}

Toast.makeText(DisplayHistoryActivity.getContext(), "正在获取数据中,请耐心等待...",

Toast.LENGTH\_LONG).show();

}

public static void hourHistory(BeanTabMessage hour, Calendar calendar, int id) {

String strUrl = RequestUtil.combineUrl((Calendar) calendar.clone());

RequestHourHistory request = new RequestHourHistory(hour, id, 0);

request.executeOnExecutor(AsyncTask.THREAD\_POOL\_EXECUTOR, strUrl);

}

private static Timer requestHourTimer = new Timer();

private static TimerTask requestHourTask = new TimerTask() {

@Override

public void run() {

// 需要做的事:发送消息

Message message = new Message();

message.what = 1;

requestHourHandler.sendMessage(message);

}

};

public static Timer getRequestHourTimer() {

return requestHourTimer;

}

public static TimerTask getRequestHourTask() {

return requestHourTask;

}

static class RequestHourHandler extends Handler {

public void handleMessage(Message msg) {

if (msg.what == 1) {

Calendar calendar = Calendar.getInstance();

hourStep(HourView.getHour(),calendar);

}

super.handleMessage(msg);

}

}

public static void hourStep(BeanTabMessage hour, Calendar calendar) {

String strUrl=RequestUtil.combineUrl((Calendar) calendar.clone());

RequestHourStep request = new RequestHourStep(hour, 0);

request.executeOnExecutor(AsyncTask.THREAD\_POOL\_EXECUTOR, strUrl);

}

}

#### RequestHourHistory.java

package ubibots.zuccweatherbase.display.control;

import android.os.AsyncTask;

import android.view.View;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Locale;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import ubibots.zuccweatherbase.display.model.BeanConstant;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

import ubibots.zuccweatherbase.display.ui.HourView;

import ubibots.zuccweatherbase.display.util.RequestUtil;

public class RequestHourHistory extends AsyncTask<String, Integer, String> {

public final static int MAX = 120;

private BeanTabMessage hour;

private int id;

private String strURL;

private int time;

public RequestHourHistory(BeanTabMessage hour, int id, int time) {

this.hour = hour;

this.id = id;

this.time = time;

}

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected String doInBackground(String... params) {

//System.out.println("Url: " + params[0]);

URL url;

try {

url = new URL(params[0]);

strURL = params[0];

HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();

urlConn.setDoInput(true); //允许输入流，即允许下载

urlConn.setDoOutput(true); //允许输出流，即允许上传

urlConn.setUseCaches(false); //不使用缓冲

urlConn.setRequestMethod("POST"); //使用get请求

InputStreamReader in = new InputStreamReader(urlConn.getInputStream());

BufferedReader bufferedReader = new BufferedReader(in);

String result = "";

String readLine;

while ((readLine = bufferedReader.readLine()) != null) {

result += readLine;

}

in.close();

urlConn.disconnect();

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(String result) {

if (result != null && time < BeanConstant.MAXTIME) {

Pattern pattern = Pattern.compile("<TD>(.\*?)</TD>");

Matcher matcher = pattern.matcher(result);

ArrayList<String> tmp = new ArrayList<>();

while (matcher.find()) {

tmp.add(matcher.group(1));

}

if (tmp.size() >= 3) {

String dateString = tmp.get(0);

double temp = Double.valueOf(tmp.get(1));

double humi = Double.valueOf(tmp.get(2));

//丢包重发

if (dateString.length() != 24 || temp < 0 || humi < 0) {

reconnect(strURL, hour, id);

return;

}

dateString = dateString.substring(0, 10) + " " + dateString.substring(11, 23);

Calendar calendar = RequestUtil.dateToCalender(dateString,"yyyy-MM-dd HH:mm:ss.SSS");

calendar.set(Calendar.HOUR\_OF\_DAY, calendar.get(Calendar.HOUR\_OF\_DAY) + 8);

SimpleDateFormat sdf = new SimpleDateFormat("HH:mm:ss", Locale.getDefault());

dateString = sdf.format(calendar.getTime());

hour.getDate().set(id, dateString);

hour.getTemperature().set(id, temp);

hour.getHumidity().set(id, humi);

hour.count++;

//历史数据收集完毕

if (hour.count == MAX) {

//刷新界面

RequestUtil.reflashLineView(HourView.getHourBeanLineView(), hour, "时:分:秒");

RequestHour.getRequestHourTimer().schedule(RequestHour.getRequestHourTask(), BeanConstant.delayHour, BeanConstant.delayHour);

HourView.getHourProgressBar().setVisibility(View.GONE);

new RequestDay().executeRequest();

}

HourView.getHourProgressBar().setProgress(100 \* hour.count / MAX);

System.out.println("Time: " + hour.getDate().get(id) + " " + "Temperature: " + hour.getTemperature().get(id) + " " + "Humidity: " + hour.getHumidity().get(id) + " " + "Num: " + id + " " + "Count: " + hour.count + " " + "Time: " + time);

} else {//丢包重发

reconnect(strURL, hour, id);

}

} else {

RequestUtil.connectFailed();

}

}

//该方法运行在UI线程当中,并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPreExecute() {

}

public void reconnect(String strURL, BeanTabMessage hour, int id) {

RequestHourHistory another = new RequestHourHistory(hour, id, time + 1);

System.out.println("time: " + time);

System.out.println(strURL);

another.execute(strURL);

}

}

#### RequestHourStep.java

package ubibots.zuccweatherbase.display.control;

import android.os.AsyncTask;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Locale;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import ubibots.zuccweatherbase.display.model.BeanConstant;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

import ubibots.zuccweatherbase.display.ui.HourView;

import ubibots.zuccweatherbase.display.util.RequestUtil;

public class RequestHourStep extends AsyncTask<String, Integer, String> {

public final static int MAX = 120;

private BeanTabMessage hour;

private String strURL;

private int time;

public RequestHourStep(BeanTabMessage hour, int time) {

this.hour = hour;

this.time = time;

}

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected String doInBackground(String... params) {

//System.out.println("Url: " + params[0]);

URL url;

try {

url = new URL(params[0]);

strURL = params[0];

HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();

urlConn.setDoInput(true); //允许输入流，即允许下载

urlConn.setDoOutput(true); //允许输出流，即允许上传

urlConn.setUseCaches(false); //不使用缓冲

urlConn.setRequestMethod("POST"); //使用get请求

InputStreamReader in = new InputStreamReader(urlConn.getInputStream());

BufferedReader bufferedReader = new BufferedReader(in);

String result = "";

String readLine;

while ((readLine = bufferedReader.readLine()) != null) {

result += readLine;

}

in.close();

urlConn.disconnect();

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(String result) {

if (result != null && time < BeanConstant.MAXTIME) {

Pattern pattern = Pattern.compile("<TD>(.\*?)</TD>");

Matcher matcher = pattern.matcher(result);

ArrayList<String> tmp = new ArrayList<>();

while (matcher.find()) {

tmp.add(matcher.group(1));

}

if (tmp.size() >= 3) {

String dateString = tmp.get(0);

double temp = Double.valueOf(tmp.get(1));

double humi = Double.valueOf(tmp.get(2));

//丢包重发

if (dateString.length() != 24 || temp < 0 || humi < 0) {

reconnect(strURL, hour);

return;

}

dateString = dateString.substring(0, 10) + " " + dateString.substring(11, 23);

Calendar calendar = RequestUtil.dateToCalender(dateString,"yyyy-MM-dd HH:mm:ss.SSS");

calendar.set(Calendar.HOUR\_OF\_DAY, calendar.get(Calendar.HOUR\_OF\_DAY) + 8);

SimpleDateFormat sdf = new SimpleDateFormat("HH:mm:ss", Locale.getDefault());

dateString = sdf.format(calendar.getTime());

hour.getDate().remove(0);

hour.getDate().add(dateString);

hour.getTemperature().remove(0);

hour.getTemperature().add(temp);

hour.getHumidity().remove(0);

hour.getHumidity().add(humi);

//刷新界面

RequestUtil.reflashLineView(HourView.getHourBeanLineView(), hour, "时:分:秒");

System.out.println("Time: " + hour.getDate().get(MAX - 1) + " " + "Temperature: " + hour.getTemperature().get(MAX - 1) + " " + "Humidity: " + hour.getHumidity().get(MAX - 1) + " " + "Time: " + time);

} else {//丢包重发

reconnect(strURL, hour);

}

} else {

RequestUtil.connectFailed();

}

}

//该方法运行在UI线程当中,并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPreExecute() {

}

public void reconnect(String strURL, BeanTabMessage hour) {

RequestHourStep another = new RequestHourStep(hour, time + 1);

System.out.println("time: " + time);

System.out.println(strURL);

another.execute(strURL);

}

}

#### UpdateRecommand.java

package ubibots.zuccweatherbase.display.control;

import android.os.AsyncTask;

import android.widget.ArrayAdapter;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.concurrent.ExecutionException;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

import ubibots.zuccweatherbase.display.intrfc.IUpdateRecommand;

import ubibots.zuccweatherbase.display.ui.HourView;

import ubibots.zuccweatherbase.display.ui.RecommandView;

import ubibots.zuccweatherbase.registandlogin.util.DBUtil;

public class UpdateRecommand implements IUpdateRecommand{

@Override

public void compareData() {

try {

float currentTemperature = HourView.getHour().getTemperature().get(HourView.getHour().getTemperature().size()-1).floatValue();

float currentHumidity = HourView.getHour().getHumidity().get(HourView.getHour().getHumidity().size()-1).floatValue();

List<String> ret = new ExecuteCompare().execute(currentTemperature,currentHumidity).get();

ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(DisplayHistoryActivity.getActivity(), android.R.layout.simple\_expandable\_list\_item\_1, ret);

RecommandView.getRecommand().setAdapter(arrayAdapter);

} catch (InterruptedException | ExecutionException e) {

e.printStackTrace();

}

}

private class ExecuteCompare extends AsyncTask<Float, Integer, List<String>> {

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected List<String> doInBackground(Float... params) {

List<String> ret = new ArrayList<>();

Connection conn = null;

try{

conn = DBUtil.getConnection();

String sql = "select activityname from beanrecommandactivity " +

"where uptemperature > ? and downtemperature < ?" +

"and uphumidity > ? and downhumidity < ?";

PreparedStatement pst = conn.prepareStatement(sql);

pst.setFloat(1,params[0]);

pst.setFloat(2,params[0]);

pst.setFloat(3,params[1]);

pst.setFloat(4,params[1]);

ResultSet rs = pst.executeQuery();

while(rs.next()){

ret.add(rs.getString(1));

}

if(ret.size()==0){

ret.add("抱歉，暂时没有建议的活动");

}

}catch (SQLException e) {

e.printStackTrace();

} finally {

if (conn != null) {

try {

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

return ret;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(List<String> result) {

}

}

}

### intrfc

#### IUpdateRecommand.java

package ubibots.zuccweatherbase.display.intrfc;

public interface IUpdateRecommand {

public void compareData();

}

### model

#### BeanConstant.java

package ubibots.zuccweatherbase.display.model;

public class BeanConstant {

public final static int MAXTIME = 3;

public final static float UPTEMP = (float)19.9;

public final static float DOWNTEMP = 14;

public final static float UPHUMI = 55;

public final static float DOWNHUMI = 45;

public final static int delayHour = 30000;

public final static int delayDay = 1800000;

}

#### BeanLineView.java

package ubibots.zuccweatherbase.display.model;

import lecho.lib.hellocharts.view.LineChartView;

public class BeanLineView {

LineChartView temperatureView;

LineChartView humidityView;

public BeanLineView(LineChartView temperatureView, LineChartView humidityView) {

this.temperatureView = temperatureView;

this.humidityView = humidityView;

}

public LineChartView getTemperatureView() {

return temperatureView;

}

public LineChartView getHumidityView() {

return humidityView;

}

}

#### BeanTabMessage.java

package ubibots.zuccweatherbase.display.model;

import java.util.ArrayList;

public class BeanTabMessage {

public int count = 0;

private ArrayList<Double> temperature;

private ArrayList<Double> humidity;

private ArrayList<String> date;

public BeanTabMessage(ArrayList<Double> temperature, ArrayList<Double> humidity, ArrayList<String> date) {

this.temperature = temperature;

this.humidity = humidity;

this.date = date;

}

public ArrayList<Double> getTemperature() {

return temperature;

}

public ArrayList<Double> getHumidity() {

return humidity;

}

public ArrayList<String> getDate() {

return date;

}

}

### ui

#### DayView.java

package ubibots.zuccweatherbase.display.ui;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.support.v4.view.PagerAdapter;

import android.support.v4.view.ViewPager;

import android.text.Spannable;

import android.text.SpannableString;

import android.text.style.ImageSpan;

import android.view.View;

import android.view.ViewGroup;

import android.widget.LinearLayout;

import android.widget.ProgressBar;

import android.widget.TextView;

import java.util.ArrayList;

import java.util.List;

import lecho.lib.hellocharts.gesture.ContainerScrollType;

import lecho.lib.hellocharts.gesture.ZoomType;

import lecho.lib.hellocharts.view.LineChartView;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

import ubibots.zuccweatherbase.display.model.BeanLineView;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

public class DayView {

private static BeanLineView dayBeanLineView;

private static BeanTabMessage day;

private static List<View> dayViewList;

private static TextView[] dayDots;

private static int dayCurrentIndex;

private static ViewPager dayViewPager;

private static ProgressBar dayProgressBar;

public static BeanLineView getDayBeanLineView() {

return dayBeanLineView;

}

public static ViewPager getDayViewPager() {

return dayViewPager;

}

public static ProgressBar getDayProgressBar() {

return dayProgressBar;

}

public static BeanTabMessage getDay() {

return day;

}

public static void setDay(BeanTabMessage day) {

DayView.day = day;

}

public DayView() {

dayViewPager = (ViewPager) DisplayHistoryActivity.getActivity().findViewById(R.id.dayView);

dayViewList = new ArrayList<>();

View view1 = View.inflate(DisplayHistoryActivity.getContext(), R.layout.temperatureday, null);

LineChartView temperatureDayView = (LineChartView) view1.findViewById(R.id.temperatureday);

temperatureDayView.setInteractive(false);

temperatureDayView.setZoomType(ZoomType.HORIZONTAL);

temperatureDayView.setContainerScrollEnabled(true, ContainerScrollType.HORIZONTAL);

temperatureDayView.setVisibility(View.VISIBLE);

View view2 = View.inflate(DisplayHistoryActivity.getContext(), R.layout.humidityday, null);

LineChartView humidityDayView = (LineChartView) view2.findViewById(R.id.humidityday);

humidityDayView.setInteractive(false);

humidityDayView.setZoomType(ZoomType.HORIZONTAL);

humidityDayView.setContainerScrollEnabled(true, ContainerScrollType.HORIZONTAL);

humidityDayView.setVisibility(View.VISIBLE);

dayViewList.add(view1);

dayViewList.add(view2);

dayBeanLineView = new BeanLineView(temperatureDayView, humidityDayView);

initDayDots();

PagerAdapter dayPagerAdapter = new PagerAdapter() {

//官方建议这么写

@Override

public boolean isViewFromObject(View arg0, Object arg1) {

return arg0 == arg1;

}

//返回一共有多少个界面

@Override

public int getCount() {

return dayViewList.size();

}

//实例化一个item

@Override

public Object instantiateItem(ViewGroup container, int position) {

container.addView(dayViewList.get(position));

return dayViewList.get(position);

}

//销毁一个item

@Override

public void destroyItem(ViewGroup container, int position, Object object) {

container.removeView(dayViewList.get(position));

}

};

dayViewPager.setAdapter(dayPagerAdapter);

dayViewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {

@Override

public void onPageSelected(int arg0) {

setDayDots(arg0);

}

@Override

public void onPageScrolled(int arg0, float arg1, int arg2) {

}

@Override

public void onPageScrollStateChanged(int arg0) {

}

});

dayProgressBar = (ProgressBar)DisplayHistoryActivity.getActivity().findViewById(R.id.dayProgressBar);

}

/\*\*

\* 初始化底部的点

\*/

private void initDayDots() {

LinearLayout dayPointLayout = (LinearLayout) DisplayHistoryActivity.getActivity().findViewById(R.id.point\_layout1);

dayDots = new TextView[dayViewList.size()];

for (int i = 0; i < dayViewList.size(); i++) {

dayDots[i] = (TextView) dayPointLayout.getChildAt(i);

setTextDrawable(dayDots[i], R.drawable.dian, i);

}

dayCurrentIndex = 0;

setTextDrawable(dayDots[dayCurrentIndex], R.drawable.dian\_down, dayCurrentIndex);

}

/\*\*

\* 当滚动的时候更换点的背景图

\*/

private void setDayDots(int position) {

if (position < 0 || position > dayViewList.size() - 1

|| dayCurrentIndex == position) {

return;

}

setTextDrawable(dayDots[position], R.drawable.dian\_down, position);

setTextDrawable(dayDots[dayCurrentIndex], R.drawable.dian, dayCurrentIndex);

dayCurrentIndex = position;

}

private void setTextDrawable(TextView tv, int id, int index) {

Bitmap b = BitmapFactory.decodeResource(DisplayHistoryActivity.getActivity().getResources(), id);

ImageSpan imgSpan = new ImageSpan(DisplayHistoryActivity.getContext(), b);

SpannableString spanString = new SpannableString("icon");

spanString.setSpan(imgSpan, 0, 4, Spannable.SPAN\_EXCLUSIVE\_EXCLUSIVE);

tv.setText(spanString);

if (index == 0) {

tv.append("温度");

} else {

tv.append("湿度");

}

}

}

#### DisplayView.java

package ubibots.zuccweatherbase.display.ui;

import ubibots.zuccweatherbase.display.control.RequestHour;

public class DisplayView {

public DisplayView() {

new HourView();

new RequestHour().executeRequest();

new DayView();

new RecommandView();

new ListTab();

}

}

#### HourView.java

package ubibots.zuccweatherbase.display.ui;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.support.v4.view.PagerAdapter;

import android.support.v4.view.ViewPager;

import android.text.Spannable;

import android.text.SpannableString;

import android.text.style.ImageSpan;

import android.view.View;

import android.view.ViewGroup;

import android.widget.LinearLayout;

import android.widget.ProgressBar;

import android.widget.TextView;

import java.util.ArrayList;

import java.util.List;

import lecho.lib.hellocharts.gesture.ContainerScrollType;

import lecho.lib.hellocharts.gesture.ZoomType;

import lecho.lib.hellocharts.view.LineChartView;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

import ubibots.zuccweatherbase.display.model.BeanLineView;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

public class HourView {

private static BeanLineView hourBeanLineView;

private static BeanTabMessage hour;

private static List<View> hourViewList;

private static TextView[] hourDots;

private static int hourCurrentIndex;

private static ViewPager hourViewPager;

private static ProgressBar hourProgressBar;

public static BeanLineView getHourBeanLineView() {

return hourBeanLineView;

}

public static ViewPager getHourViewPager() {

return hourViewPager;

}

public static ProgressBar getHourProgressBar() {

return hourProgressBar;

}

public static BeanTabMessage getHour() {

return hour;

}

public static void setHour(BeanTabMessage hour) {

HourView.hour = hour;

}

public HourView() {

hourViewPager = (ViewPager) DisplayHistoryActivity.getActivity().findViewById(R.id.hourView);

hourViewList = new ArrayList<>();

View view1 = View.inflate(DisplayHistoryActivity.getContext(), R.layout.temperaturehour, null);

LineChartView temperatureHourView = (LineChartView) view1.findViewById(R.id.temperaturehour);

temperatureHourView.setInteractive(false);

temperatureHourView.setZoomType(ZoomType.HORIZONTAL);

temperatureHourView.setContainerScrollEnabled(true, ContainerScrollType.HORIZONTAL);

temperatureHourView.setVisibility(View.VISIBLE);

View view2 = View.inflate(DisplayHistoryActivity.getContext(), R.layout.humidityhour, null);

LineChartView humidityHourView = (LineChartView) view2.findViewById(R.id.humidityhour);

humidityHourView.setInteractive(false);

humidityHourView.setZoomType(ZoomType.HORIZONTAL);

humidityHourView.setContainerScrollEnabled(true, ContainerScrollType.HORIZONTAL);

humidityHourView.setVisibility(View.VISIBLE);

hourViewList.add(view1);

hourViewList.add(view2);

hourBeanLineView = new BeanLineView(temperatureHourView, humidityHourView);

initHourDots();

PagerAdapter hourPagerAdapter = new PagerAdapter() {

//官方建议这么写

@Override

public boolean isViewFromObject(View arg0, Object arg1) {

return arg0 == arg1;

}

//返回一共有多少个界面

@Override

public int getCount() {

return hourViewList.size();

}

//实例化一个item

@Override

public Object instantiateItem(ViewGroup container, int position) {

container.addView(hourViewList.get(position));

return hourViewList.get(position);

}

//销毁一个item

@Override

public void destroyItem(ViewGroup container, int position, Object object) {

container.removeView(hourViewList.get(position));

}

};

hourViewPager.setAdapter(hourPagerAdapter);

hourViewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {

@Override

public void onPageSelected(int arg0) {

setHourDots(arg0);

}

@Override

public void onPageScrolled(int arg0, float arg1, int arg2) {

}

@Override

public void onPageScrollStateChanged(int arg0) {

}

});

hourProgressBar = (ProgressBar)DisplayHistoryActivity.getActivity().findViewById(R.id.hourProgressBar);

}

/\*\*

\* 初始化底部的点

\*/

private void initHourDots() {

LinearLayout hourPointLayout = (LinearLayout) DisplayHistoryActivity.getActivity().findViewById(R.id.point\_layout1);

hourDots = new TextView[hourViewList.size()];

for (int i = 0; i < hourViewList.size(); i++) {

hourDots[i] = (TextView) hourPointLayout.getChildAt(i);

setTextDrawable(hourDots[i], R.drawable.dian, i);

}

hourCurrentIndex = 0;

setTextDrawable(hourDots[hourCurrentIndex], R.drawable.dian\_down, hourCurrentIndex);

}

/\*\*

\* 当滚动的时候更换点的背景图

\*/

private void setHourDots(int position) {

if (position < 0 || position > hourViewList.size() - 1

|| hourCurrentIndex == position) {

return;

}

setTextDrawable(hourDots[position], R.drawable.dian\_down, position);

setTextDrawable(hourDots[hourCurrentIndex], R.drawable.dian, hourCurrentIndex);

hourCurrentIndex = position;

}

private void setTextDrawable(TextView tv, int id, int index) {

Bitmap b = BitmapFactory.decodeResource(DisplayHistoryActivity.getActivity().getResources(), id);

ImageSpan imgSpan = new ImageSpan(DisplayHistoryActivity.getContext(), b);

SpannableString spanString = new SpannableString("icon");

spanString.setSpan(imgSpan, 0, 4, Spannable.SPAN\_EXCLUSIVE\_EXCLUSIVE);

tv.setText(spanString);

if (index == 0) {

tv.append("温度");

} else {

tv.append("湿度");

}

}

}

#### ListTab.java

package ubibots.zuccweatherbase.display.ui;

import android.graphics.Color;

import android.view.View;

import android.widget.ArrayAdapter;

import android.widget.ListView;

import java.util.ArrayList;

import java.util.List;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

public class ListTab {

private int currentTab;

private ListView listView;

public int getCurrentTab() {

return currentTab;

}

public ListView getListView() {

return listView;

}

public ListTab() {

listView = (ListView) DisplayHistoryActivity.getActivity().findViewById(R.id.listview);

listView.setBackgroundColor(Color.GRAY);

listView.setCacheColorHint(0);

final List<String> data = new ArrayList<>();

data.add("每时");

data.add("每日");

data.add("活动推荐");

ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(DisplayHistoryActivity.getActivity(), android.R.layout.simple\_expandable\_list\_item\_1, data);

listView.setAdapter(arrayAdapter);

listView.setOnItemClickListener((parent, view, position, id) -> {

String click = data.get(position);

switch (click) {

case "每时":

hourVisible();

currentTab = 0;

break;

case "每日":

dayVisible();

currentTab = 1;

break;

case "活动推荐":

break;

}

});

hourVisible();

currentTab = 0;

}

private void hourVisible() {

if (HourView.getHourViewPager() != null) {

HourView.getHourViewPager().setVisibility(View.VISIBLE);

dayInvisible();

if (HourView.getHourProgressBar().getVisibility() != View.GONE) {

HourView.getHourProgressBar().setVisibility(View.VISIBLE);

}

}

}

private void hourInvisible() {

if (HourView.getHourViewPager() != null) {

HourView.getHourViewPager().setVisibility(View.INVISIBLE);

if (HourView.getHourProgressBar().getVisibility() != View.GONE) {

HourView.getHourProgressBar().setVisibility(View.INVISIBLE);

}

}

}

private void dayVisible() {

if (DayView.getDayViewPager() != null) {

DayView.getDayViewPager().setVisibility(View.VISIBLE);

hourInvisible();

if (DayView.getDayProgressBar().getVisibility() != View.GONE) {

DayView.getDayProgressBar().setVisibility(View.VISIBLE);

}

}

}

private void dayInvisible() {

if (DayView.getDayViewPager() != null) {

DayView.getDayViewPager().setVisibility(View.INVISIBLE);

if (DayView.getDayProgressBar().getVisibility() != View.GONE) {

DayView.getDayProgressBar().setVisibility(View.INVISIBLE);

}

}

}

}

#### RecommandView.java

package ubibots.zuccweatherbase.display.ui;

import android.widget.ListView;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

public class RecommandView {

private static ListView recommand;

public static ListView getRecommand() {

return recommand;

}

public RecommandView(){

recommand = (ListView) DisplayHistoryActivity.getActivity().findViewById(R.id.recommandList);

recommand.setClickable(false);

}

}

### util

#### RequestUtil.java

package ubibots.zuccweatherbase.display.util;

import android.graphics.Color;

import android.widget.Toast;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.HashMap;

import java.util.List;

import java.util.Locale;

import java.util.Map;

import lecho.lib.hellocharts.model.Axis;

import lecho.lib.hellocharts.model.AxisValue;

import lecho.lib.hellocharts.model.Line;

import lecho.lib.hellocharts.model.LineChartData;

import lecho.lib.hellocharts.model.PointValue;

import ubibots.zuccweatherbase.display.model.BeanConstant;

import ubibots.zuccweatherbase.display.model.BeanLineView;

import ubibots.zuccweatherbase.display.model.BeanTabMessage;

import ubibots.zuccweatherbase.registandlogin.RegistAndLoginActivity;

public class RequestUtil {

public static String UTCDateFormat(Calendar calendar) {

String UTCDate;

SimpleDateFormat sdf;

Calendar tmp = (Calendar) calendar.clone();

tmp.set(Calendar.HOUR\_OF\_DAY, tmp.get(Calendar.HOUR\_OF\_DAY) - 8);

sdf = new SimpleDateFormat("yyyy-MM-dd", Locale.getDefault());

UTCDate = sdf.format(tmp.getTime()) + "T";

sdf = new SimpleDateFormat("HH:mm:ss", Locale.getDefault());

UTCDate += sdf.format(tmp.getTime()) + ".";

sdf = new SimpleDateFormat("SSS", Locale.getDefault());

UTCDate += sdf.format(tmp.getTime()) + "Z";

return UTCDate;

}

public static void connectFailed() {

Toast.makeText(RegistAndLoginActivity.getContext(), "连接失败，请检查网络环境并重启本程序...",

Toast.LENGTH\_SHORT).show();

}

public static void reflashLineView(BeanLineView lineView, BeanTabMessage tab, String xName) {

List<Line> temperatureLineList = new ArrayList<>();

List<PointValue> temperatureValuesList;

Line temperatureLine;

List<AxisValue> temperatureAxisValue = new ArrayList<>();

final int LOWLINE = 0;

final int MIDLINE = 1;

final int UPLINE = 2;

int state = -1;

temperatureValuesList = new ArrayList<>();

temperatureLine = new Line(temperatureValuesList).setColor(Color.BLACK).setCubic(false);

temperatureLine.setHasPoints(false);

temperatureLine.setHasLines(false);

temperatureLine = null;

float maxTemperature = -2333;

float minTemperature = 2333;

for (int i = 0; i < tab.getTemperature().size(); i++) {

float tmpTemperature = tab.getTemperature().get(i).floatValue();

maxTemperature = Math.max(tmpTemperature, maxTemperature);

minTemperature = Math.min(tmpTemperature, minTemperature);

if (tmpTemperature < BeanConstant.DOWNTEMP && state != LOWLINE) {

float tmp = -1;

if (temperatureLine != null) {

temperatureLineList.add(temperatureLine);

tmp = tab.getTemperature().get(i - 1).floatValue();

}

temperatureValuesList = new ArrayList<>();

temperatureLine = new Line(temperatureValuesList).setColor(Color.BLUE).setCubic(false);

temperatureLine.setHasPoints(false);

if (tmp != -1) {

temperatureValuesList.add(new PointValue(i - 1, tmp));

}

temperatureValuesList.add(new PointValue(i, tmpTemperature));

state = LOWLINE;

} else if (tmpTemperature >= BeanConstant.DOWNTEMP && tmpTemperature <= BeanConstant.UPTEMP && state != MIDLINE) {

float tmp = -1;

if (temperatureLine != null) {

temperatureLineList.add(temperatureLine);

tmp = tab.getTemperature().get(i - 1).floatValue();

}

temperatureValuesList = new ArrayList<>();

temperatureLine = new Line(temperatureValuesList).setColor(Color.GREEN).setCubic(false);

temperatureLine.setHasPoints(false);

if (tmp != -1) {

temperatureValuesList.add(new PointValue(i - 1, tmp));

}

temperatureValuesList.add(new PointValue(i, tmpTemperature));

state = MIDLINE;

} else if (tmpTemperature > BeanConstant.UPTEMP && state != UPLINE) {

float tmp = -1;

if (temperatureLine != null) {

temperatureLineList.add(temperatureLine);

tmp = tab.getTemperature().get(i - 1).floatValue();

}

temperatureValuesList = new ArrayList<>();

temperatureLine = new Line(temperatureValuesList).setColor(Color.RED).setCubic(false);

temperatureLine.setHasPoints(false);

if (tmp != -1) {

temperatureValuesList.add(new PointValue(i - 1, tmp));

}

temperatureValuesList.add(new PointValue(i, tmpTemperature));

state = UPLINE;

} else {

temperatureValuesList.add(new PointValue(i, tmpTemperature));

}

temperatureAxisValue.add(new AxisValue(i).setLabel(tab.getDate().get(i)));

}

temperatureLineList.add(temperatureLine);

temperatureValuesList = new ArrayList<>();

temperatureLine = new Line(temperatureValuesList).setColor(Color.BLACK).setCubic(false);

temperatureLine.setHasPoints(false);

temperatureLine.setHasLines(false);

float mm = maxTemperature - minTemperature;

temperatureValuesList.add(new PointValue(0, maxTemperature + mm + 1));

temperatureValuesList.add(new PointValue(1, minTemperature - mm - 1));

temperatureLineList.add(temperatureLine);

LineChartData temperatureData = new LineChartData();

temperatureData.setLines(temperatureLineList);

//坐标轴

Axis axisX = new Axis();//X轴

axisX.setHasLines(true);

axisX.setHasTiltedLabels(true);

axisX.setTextColor(Color.WHITE);

axisX.setName(xName);

axisX.setMaxLabelChars(6);

axisX.setValues(temperatureAxisValue);

temperatureData.setAxisXBottom(axisX);

Axis axisY1 = new Axis();//Y1轴

axisY1.setHasLines(true);

axisY1.setTextColor(Color.WHITE);

axisY1.setName("摄氏度/℃");

axisY1.setMaxLabelChars(4);

temperatureData.setAxisYLeft(axisY1);

Axis axisY2 = new Axis();//Y2轴

axisY2.setHasLines(true);

axisY2.setTextColor(Color.WHITE);

axisY2.setName("摄氏度/℃");

axisY2.setMaxLabelChars(4);

temperatureData.setAxisYRight(axisY2);

lineView.getTemperatureView().setLineChartData(temperatureData);

List<Line> humidityLineList = new ArrayList<>();

List<PointValue> humidityValuesList;

Line humidityLine;

List<AxisValue> humidityAxisValue = new ArrayList<>();

state = -1;

humidityValuesList = new ArrayList<>();

humidityLine = new Line(humidityValuesList).setColor(Color.BLACK).setCubic(false);

humidityLine.setHasPoints(false);

humidityLine.setHasLines(false);

humidityLine = null;

float maxHumidity = -2333;

float minHumidity = 2333;

for (int i = 0; i < tab.getHumidity().size(); i++) {

float tmpHumidity = tab.getHumidity().get(i).floatValue();

maxHumidity = Math.max(tmpHumidity, maxHumidity);

minHumidity = Math.min(tmpHumidity, minHumidity);

if (tmpHumidity < BeanConstant.DOWNHUMI && state != LOWLINE) {

float tmp = -1;

if (humidityLine != null) {

humidityLineList.add(humidityLine);

tmp = tab.getHumidity().get(i - 1).floatValue();

}

humidityValuesList = new ArrayList<>();

humidityLine = new Line(humidityValuesList).setColor(Color.BLUE).setCubic(false);

humidityLine.setHasPoints(false);

if (tmp != -1) {

humidityValuesList.add(new PointValue(i - 1, tmp));

}

humidityValuesList.add(new PointValue(i, tmpHumidity));

state = LOWLINE;

} else if (tmpHumidity >= BeanConstant.DOWNHUMI && tmpHumidity <= BeanConstant.UPHUMI && state != MIDLINE) {

float tmp = -1;

if (humidityLine != null) {

humidityLineList.add(humidityLine);

tmp = tab.getHumidity().get(i - 1).floatValue();

}

humidityValuesList = new ArrayList<>();

humidityLine = new Line(humidityValuesList).setColor(Color.GREEN).setCubic(false);

humidityLine.setHasPoints(false);

if (tmp != -1) {

humidityValuesList.add(new PointValue(i - 1, tmp));

}

humidityValuesList.add(new PointValue(i, tmpHumidity));

state = MIDLINE;

} else if (tmpHumidity > BeanConstant.UPHUMI && state != UPLINE) {

float tmp = -1;

if (humidityLine != null) {

humidityLineList.add(humidityLine);

tmp = tab.getHumidity().get(i - 1).floatValue();

}

humidityValuesList = new ArrayList<>();

humidityLine = new Line(humidityValuesList).setColor(Color.RED).setCubic(false);

humidityLine.setHasPoints(false);

if (tmp != -1) {

humidityValuesList.add(new PointValue(i - 1, tmp));

}

humidityValuesList.add(new PointValue(i, tmpHumidity));

state = UPLINE;

} else {

humidityValuesList.add(new PointValue(i, tmpHumidity));

}

humidityAxisValue.add(new AxisValue(i).setLabel(tab.getDate().get(i)));

}

humidityLineList.add(humidityLine);

humidityValuesList = new ArrayList<>();

humidityLine = new Line(humidityValuesList).setColor(Color.BLACK).setCubic(false);

humidityLine.setHasPoints(false);

humidityLine.setHasLines(false);

mm = maxHumidity - minHumidity;

humidityValuesList.add(new PointValue(0, maxHumidity + mm + 1));

humidityValuesList.add(new PointValue(1, minHumidity - mm - 1));

humidityLineList.add(humidityLine);

LineChartData humidityData = new LineChartData();

humidityData.setLines(humidityLineList);

//坐标轴

axisX = new Axis();//X轴

axisX.setHasLines(true);

axisX.setHasTiltedLabels(true);

axisX.setTextColor(Color.WHITE);

axisX.setName(xName);

axisX.setMaxLabelChars(6);

axisX.setValues(humidityAxisValue);

humidityData.setAxisXBottom(axisX);

axisY1 = new Axis();//Y1轴

axisY1.setHasLines(true);

axisY1.setTextColor(Color.WHITE);

axisY1.setName("摄氏度/℃");

axisY1.setMaxLabelChars(4);

humidityData.setAxisYLeft(axisY1);

axisY2 = new Axis();//Y2轴

axisY2.setHasLines(true);

axisY2.setTextColor(Color.WHITE);

axisY2.setName("摄氏度/℃");

axisY2.setMaxLabelChars(4);

humidityData.setAxisYRight(axisY2);

lineView.getHumidityView().setLineChartData(humidityData);

}

public static String combineUrl(Calendar calendar) {

String ipAddress = "zucc.cloud.thingworx.com:80";

String appKey = "deaf648e-e691-4e9e-88a9-1a80b21145c3";

String things = "DHT21Thing";

String service;

String strUrl;

String startDate;

String endDate;

Calendar tmp = (Calendar) calendar.clone();

endDate = RequestUtil.UTCDateFormat(tmp);

tmp.set(Calendar.YEAR, tmp.get(Calendar.YEAR) - 1);

startDate = RequestUtil.UTCDateFormat(tmp);

service = "QueryPropertyHistory";

strUrl = "http://"

+ ipAddress

+ "/Thingworx"

+ "/Things/" + things

+ "/Services/" + service + "?";

Map<String, String> params;

params = new HashMap<>();

params.put("method", "post");

params.put("appKey", appKey);

params.put("startDate", startDate);

params.put("endDate", endDate);

params.put("maxItems", "1");

params.put("oldestFirst", "false");

strUrl = RequestUtil.addParameter(strUrl, params);

return strUrl;

}

public static String addParameter(String path, Map<String, String> params) {

String URL = path;

if (params != null && URL.length() != 0) {

for (Map.Entry<String, String> entry : params.entrySet()) {

try {

URL += entry.getKey() + "=" + entry.getValue();

} catch (Exception ex) {

ex.printStackTrace();

}

URL += "&";

}

URL = URL.substring(0, URL.length() - 1);

}

return URL;

}

public static Calendar dateToCalender(String string, String format){

Calendar calendar = null;

try {

SimpleDateFormat sdf = new SimpleDateFormat(format, Locale.getDefault());

calendar = Calendar.getInstance();

calendar.setTime(sdf.parse(string));

}catch (Exception ex){

ex.printStackTrace();

}

return calendar;

}

}

### DisplayHistoryActivity.java

package ubibots.zuccweatherbase.display;

import android.app.Activity;

import android.content.Context;

import android.os.Bundle;

import android.view.WindowManager;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.display.ui.DisplayView;

public class DisplayHistoryActivity extends Activity {

private static Context context;

private static DisplayHistoryActivity activity;

public static Context getContext() {

return context;

}

public static DisplayHistoryActivity getActivity() {

return activity;

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.display\_main);

context = this;

activity = this;

getWindow().addFlags(WindowManager.LayoutParams.FLAG\_KEEP\_SCREEN\_ON);

new DisplayView();

}

}

## registandlogin

### control

#### UserManager.java

package ubibots.zuccweatherbase.registandlogin.control;

import android.content.Intent;

import android.os.AsyncTask;

import android.widget.Toast;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.concurrent.ExecutionException;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import ubibots.zuccweatherbase.display.DisplayHistoryActivity;

import ubibots.zuccweatherbase.registandlogin.RegistAndLoginActivity;

import ubibots.zuccweatherbase.registandlogin.intrfc.IUserManager;

import ubibots.zuccweatherbase.registandlogin.ui.FrmLogin;

import ubibots.zuccweatherbase.registandlogin.util.DBUtil;

import ubibots.zuccweatherbase.registandlogin.util.RegistAndLoginUtil;

public class UserManager implements IUserManager {

@Override

public String login(String userName, String userPassword) {

String ret = null;

try {

ret = new ExecuteLogin().execute(userName, userPassword).get();

if(ret.equals("欢迎使用")) {

if (RegistAndLoginActivity.getRegistAndLoginActivity() != null) {

/\* 新建一个Intent对象 \*/

Intent intent = new Intent();

/\* 指定intent要启动的类 \*/

intent.setClass(RegistAndLoginActivity.getRegistAndLoginActivity(), DisplayHistoryActivity.class);

/\* 启动一个新的Activity \*/

RegistAndLoginActivity.getRegistAndLoginActivity().startActivity(intent);

/\* 关闭当前的Activity \*/

RegistAndLoginActivity.getRegistAndLoginActivity().finish();

}

}

} catch (InterruptedException | ExecutionException e) {

e.printStackTrace();

}

return ret;

}

public class ExecuteLogin extends AsyncTask<String, Integer, String> {

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected String doInBackground(String... params) {

Connection conn = null;

String ret = "";

try {

if ("".equals(params[0])) {

ret = "用户名不能为空";

return ret;

}

conn = DBUtil.getConnection();

String sql = "select userpassword from beanuser where username " +

"= ?";

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, params[0]);

ResultSet rs = pst.executeQuery();

if (!rs.next()) {

ret = "用户名不存在";

return ret;

}

if ("".equals(params[1])) {

ret = "密码不能为空";

return ret;

}

if (!RegistAndLoginUtil.MD5Encrypt(params[1]).equals(rs.getString(1))) {

ret = "密码不正确";

return ret;

}

ret = "欢迎使用";

} catch (SQLException e) {

e.printStackTrace();

ret = "网络连接中断";

} finally {

if (conn != null) {

try {

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

return ret;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(String result) {

if (RegistAndLoginActivity.getContext() != null) {

Toast.makeText(RegistAndLoginActivity.getContext(), result, Toast.LENGTH\_SHORT).show();

}

}

}

@Override

public String regist(String userName, String userMail, String userPassword, String userPassword2) {

String ret = null;

try {

ret = new ExecuteRegist().execute(userName, userMail, userPassword, userPassword2).get();

} catch (InterruptedException | ExecutionException e) {

e.printStackTrace();

}

return ret;

}

public class ExecuteRegist extends AsyncTask<String, Integer, String> {

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected String doInBackground(String... params) {

String ret = "";

Connection conn = null;

try {

if ("".equals(params[0])) {

ret = "用户名不能为空";

return ret;

}

conn = DBUtil.getConnection();

String sql = "select \* from beanuser where username = ?";

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, params[0]);

ResultSet rs = pst.executeQuery();

if (rs.next()) {

ret = "用户名已存在";

return ret;

}

conn.close();

if ("".equals(params[1])) {

ret = "邮箱不能为空";

return ret;

}

String check = "^([a-z0-9A-Z]+[-|\\.]?)+[a-z0-9A-Z]@([a-z0-9A-Z]+(-[a-z0-9A-Z]+)?\\.)+[a-zA-Z]{2,}$";

Pattern regex = Pattern.compile(check);

Matcher matcher = regex.matcher(params[1]);

if (!matcher.matches()) {

ret = "邮箱格式不对";

return ret;

}

conn = DBUtil.getConnection();

sql = "select \* from beanuser where usermail = ?";

pst = conn.prepareStatement(sql);

pst.setString(1, params[1]);

rs = pst.executeQuery();

if (rs.next()) {

ret = "邮箱已存在";

return ret;

}

conn.close();

if ("".equals(params[2])) {

ret = "密码不能为空";

} else if ("".equals(params[3])) {

ret = "确认密码不能为空";

} else if (!params[2].equals(params[3])) {

ret = "密码和确认密码不一致";

}

if (!"".equals(ret)) {

return ret;

}

conn = DBUtil.getConnection();

sql = "insert into beanuser(username, usermail, userpassword) values(?,?,?)";

pst = conn.prepareStatement(sql);

pst.setString(1, params[0]);

pst.setString(2, params[1]);

pst.setString(3, RegistAndLoginUtil.MD5Encrypt(params[2]));

pst.executeUpdate();

ret = "欢迎使用";

} catch (SQLException e) {

e.printStackTrace();

ret = "网络连接中断";

} finally {

if (conn != null) {

try {

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

return ret;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(String result) {

if (result.equals("欢迎使用")) {

if (FrmLogin.frmRegist != null) {

FrmLogin.frmRegist.setTmpUserName("");

FrmLogin.frmRegist.setTmpUserMail("");

}

}

if (FrmLogin.frmRegist != null) {

FrmLogin.frmRegist.setTmpUserPassword("");

FrmLogin.frmRegist.setTmpUserPassword2("");

}

if (RegistAndLoginActivity.getContext() != null) {

Toast.makeText(RegistAndLoginActivity.getContext(), result, Toast.LENGTH\_SHORT).show();

}

}

}

@Override

public String remove(String userName) {

String ret = null;

try {

ret = new ExecuteRemove().execute(userName).get();

} catch (InterruptedException | ExecutionException e) {

e.printStackTrace();

}

return ret;

}

public class ExecuteRemove extends AsyncTask<String, Integer, String> {

//该方法并不运行在UI线程当中，主要用于异步操作，所有在该方法中不能对UI当中的空间进行设置和修改

@Override

protected String doInBackground(String... params) {

String ret = "";

if ("".equals(params[0])) {

ret = "用户名不能为空";

return ret;

}

Connection conn = null;

try {

conn = DBUtil.getConnection();

String sql = "select \* from beanuser where username = ?";

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, params[0]);

ResultSet rs = pst.executeQuery();

if (!rs.next()) {

ret = "用户名不存在";

return ret;

}

conn.close();

conn = DBUtil.getConnection();

sql = "delete from beanuser where username = ?";

pst = conn.prepareStatement(sql);

pst.setString(1, params[0]);

pst.executeUpdate();

ret = "删除成功";

} catch (SQLException e) {

e.printStackTrace();

} finally {

if (conn != null) {

try {

conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

return ret;

}

//在doInBackground方法执行结束之后在运行，并且运行在UI线程当中 可以对UI空间进行设置

@Override

protected void onPostExecute(String result) {

}

}

}

### intrfc

#### IUserManager.java

package ubibots.zuccweatherbase.registandlogin.intrfc;

public interface IUserManager {

String login(String userName, String userPassword);

String regist(String userName, String userMail, String userPassword, String userPassword2);

String remove(String userName);

}

### model

#### BeanUser.java

package ubibots.zuccweatherbase.registandlogin.model;

public class BeanUser {

private int userid;

private String userName;

private String userMail;

private String userPassword;

public int getUserid() {

return userid;

}

public void setUserid(int userid) {

this.userid = userid;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getUserMail() {

return userMail;

}

public void setUserMail(String userMail) {

this.userMail = userMail;

}

public String getUserPassword() {

return userPassword;

}

public void setUserPassword(String userPassword) {

this.userPassword = userPassword;

}

}

### ui

#### FrmLogin.java

package ubibots.zuccweatherbase.registandlogin.ui;

import android.support.v4.content.ContextCompat;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageView;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.registandlogin.RegistAndLoginActivity;

import ubibots.zuccweatherbase.registandlogin.util.RegistAndLoginUtil;

public class FrmLogin {

private EditText userName;

private EditText userPassword;

public static FrmRegist frmRegist;

public FrmLogin() {

userName = (EditText) RegistAndLoginActivity.getRegistAndLoginActivity().findViewById(R.id.edtusername);

userPassword = (EditText) RegistAndLoginActivity.getRegistAndLoginActivity().findViewById(R.id.edtuserpassword);

Button registButton = (Button) RegistAndLoginActivity.getRegistAndLoginActivity().findViewById(R.id.regist);

Button loginButton = (Button) RegistAndLoginActivity.getRegistAndLoginActivity().findViewById(R.id.login);

loginButton.setOnClickListener(v -> {

RegistAndLoginUtil.userManager.login(userName.getText().toString(), userPassword.getText().toString());

userPassword.setText("");

});

frmRegist = new FrmRegist();

registButton.setOnClickListener(v -> frmRegist.registDialog());

ImageView appLogo = (ImageView) (RegistAndLoginActivity.getRegistAndLoginActivity().findViewById(R.id.applogo));

appLogo.setImageDrawable(ContextCompat.getDrawable(RegistAndLoginActivity.getRegistAndLoginActivity(),R.drawable.logo));

}

}

#### Frmregist.java

package ubibots.zuccweatherbase.registandlogin.ui;

import android.support.v7.app.AlertDialog;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.EditText;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.registandlogin.RegistAndLoginActivity;

import ubibots.zuccweatherbase.registandlogin.control.UserManager;

public class FrmRegist {

private String tmpUserName;

private String tmpUserMail;

private String tmpUserPassword;

private String tmpUserPassword2;

public void setTmpUserName(String tmpUserName) {

this.tmpUserName = tmpUserName;

}

public void setTmpUserMail(String tmpUserMail) {

this.tmpUserMail = tmpUserMail;

}

public void setTmpUserPassword(String tmpUserPassword) {

this.tmpUserPassword = tmpUserPassword;

}

public void setTmpUserPassword2(String tmpUserPassword2) {

this.tmpUserPassword2 = tmpUserPassword2;

}

public FrmRegist(){

tmpUserName = "";

tmpUserMail = "";

tmpUserPassword = "";

tmpUserPassword2 = "";

}

public void registDialog() {

LayoutInflater inflater = RegistAndLoginActivity.getRegistAndLoginActivity().getLayoutInflater();

final View layout = inflater.inflate(R.layout.regist\_layout,

(ViewGroup) RegistAndLoginActivity.getRegistAndLoginActivity().findViewById(R.id.registdialog));

EditText userName = (EditText) layout.findViewById(R.id.edtusername);

EditText userMail = (EditText) layout.findViewById(R.id.edtusermail);

EditText userPassword = (EditText) layout.findViewById(R.id.edtuserpassword);

EditText userPassword2 = (EditText) layout.findViewById(R.id.edtuserpassword2);

if (!"".equals(tmpUserName)) {

userName.setText(tmpUserName);

}

if (!"".equals(tmpUserMail)) {

userMail.setText(tmpUserMail);

}

if (!"".equals(tmpUserPassword)) {

userPassword.setText(tmpUserPassword);

}

if (!"".equals(tmpUserPassword2)) {

userPassword2.setText(tmpUserPassword2);

}

new AlertDialog.Builder(RegistAndLoginActivity.getContext()).setTitle("Regist").setView(layout)

.setPositiveButton("确定", (dialog, which) -> {

tmpUserName = userName.getText().toString();

tmpUserMail = userMail.getText().toString();

tmpUserPassword = userPassword.getText().toString();

tmpUserPassword2 = userPassword2.getText().toString();

new UserManager().regist(tmpUserName, tmpUserMail, tmpUserPassword, tmpUserPassword2);

}).setNegativeButton("取消", null).show()

.setCanceledOnTouchOutside(false);

}

}

### util

#### BaseException.java

package ubibots.zuccweatherbase.registandlogin.util;

public class BaseException extends Exception {

public BaseException(String msg){

super(msg);

}

}

#### BusinessException.java

package ubibots.zuccweatherbase.registandlogin.util;

public class BusinessException extends BaseException {

public BusinessException(String msg){

super(msg);

}

}

#### DbException.java

package ubibots.zuccweatherbase.registandlogin.util;

public class DbException extends BaseException {

public DbException(Throwable ex){

super("Database error: "+ex.getMessage());

}

}

#### DBUtil.java

package ubibots.zuccweatherbase.registandlogin.util;

import java.sql.Connection;

public class DBUtil {

private static final String jdbcUrl="jdbc:mysql://10.66.15.150:3306/weatherbase?useUnicode=true&characterEncoding=UTF-8&useSSL=true&connectTimeout=10000";

private static final String dbUser="root";

private static final String dbPwd="zucc";

static{

try {

Class.forName("com.mysql.jdbc.Driver");

} catch(ClassNotFoundException ex) {

ex.printStackTrace();

}

}

public static Connection getConnection() throws java.sql.SQLException{

return java.sql.DriverManager.getConnection(jdbcUrl, dbUser, dbPwd);

}

}

#### RegistAndLoginUtil.java

package ubibots.zuccweatherbase.registandlogin.util;

import java.io.UnsupportedEncodingException;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import ubibots.zuccweatherbase.registandlogin.control.UserManager;

import ubibots.zuccweatherbase.registandlogin.intrfc.IUserManager;

public class RegistAndLoginUtil {

public static IUserManager userManager = new UserManager();

public static String MD5Encrypt(String initial){

String ret = null;

try {

byte[] bytesOfMessage = initial.getBytes("UTF-8");

MessageDigest md = MessageDigest.getInstance("MD5");

byte[] thedigest = md.digest(bytesOfMessage);

ret = printHexBinary(thedigest);

System.out.println(ret);

} catch (UnsupportedEncodingException | NoSuchAlgorithmException e) {

e.printStackTrace();

}

return ret;

}

private static String printHexBinary(byte[] initial){

StringBuilder hexValue = new StringBuilder();

for (byte aThedigest : initial) {

int val = ((int) aThedigest) & 0xff;

if (val < 16)

hexValue.append("0");

hexValue.append(Integer.toHexString(val));

}

return hexValue.toString();

}

}

### RegistAndLoginActivity.java

package ubibots.zuccweatherbase.registandlogin;

import android.app.Activity;

import android.content.Context;

import android.os.Bundle;

import ubibots.zuccweatherbase.R;

import ubibots.zuccweatherbase.registandlogin.ui.FrmLogin;

public class RegistAndLoginActivity extends Activity {

private static RegistAndLoginActivity registAndLoginActivity;

private static Context context;

public static RegistAndLoginActivity getRegistAndLoginActivity() {

return registAndLoginActivity;

}

public static Context getContext() {

return context;

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.login\_main);

registAndLoginActivity = this;

context = this;

new FrmLogin();

}

}

## MainActivity.java

package ubibots.zuccweatherbase;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import ubibots.zuccweatherbase.registandlogin.RegistAndLoginActivity;

public class MainActivity extends Activity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.login\_main);

/\* 新建一个Intent对象 \*/

Intent intent = new Intent();

/\* 指定intent要启动的类 \*/

intent.setClass(MainActivity.this, RegistAndLoginActivity.class);

/\* 启动一个新的Activity \*/

MainActivity.this.startActivity(intent);

/\* 关闭当前的Activity \*/

MainActivity.this.finish();

}

}