1、Json转化为实体类

* 安装Gson

在build.gradle文件中的dependencie标签内添加

compile 'com.google.code.gson:gson:2.7'

* 使用

try {  
 JSONObject jsonObject=new JSONObject(json);  
 JSONArray jsonArray=jsonObject.getJSONArray("data");  
 List<UserInfoModel> userInfoModelList=new ArrayList<UserInfoModel>();  
 //这两句代码必须的，为的是初始化出来gson这个对象，才能拿来用  
 Type type1=new TypeToken<List<UserInfoModel>>(){}.getType();  
 userInfoModelList=new Gson().fromJson(jsonArray.toString(),type1);  
 System.out.println("----1212-----"+userInfoModelList.size());  
} catch (JSONException e) {  
 e.printStackTrace();  
}

2、设置连接超时

HttpClient httpClient=new DefaultHttpClient();  
HttpParams params = null;  
params = httpClient.getParams();  
//若3秒未连接上，则强制断开  
HttpConnectionParams.setConnectionTimeout(params, 3000);  
//若连接上3秒之后没有接收到数据，则强制断开  
HttpConnectionParams.setSoTimeout(params, 3000);

3、设置Activity停留几秒后再跳转

private void getHomeActivity(final Context context) {  
 Timer timer=new Timer();  
 TimerTask task=new TimerTask(){  
 public void run(){

Intent intent = new Intent(InitActivity.this, MainActivity.class)；

startActivity(intent);  
 }  
 };  
 timer.schedule(task, 2000);  
}

4、自动登录

* 用户信息管理类

package ouc.edu.com.function;  
import android.content.Context;  
import android.content.SharedPreferences;  
import android.text.TextUtils;  
import ouc.edu.com.model.UserInfoModel;  
/\*\*  
 \* Created by JHC on 2017/11/23.  
 \* 自动登录  
 \*/  
public class AutoMaticLogin {  
 private static AutoMaticLogin instance;  
 private AutoMaticLogin() {  
 }  
 public static AutoMaticLogin getInstance() {  
 if (instance == null) {  
 instance = new AutoMaticLogin();  
 }  
 return instance;  
 }  
 /\*\*  
 \* 保存自动登录的用户信息  
 \*/  
 public void saveUserInfo(Context context, String username, String password) {  
 SharedPreferences sp = context.getSharedPreferences("userInfo", Context.MODE\_PRIVATE);//Context.MODE\_PRIVATE表示SharePrefences的数据只有自己应用程序能访问。  
 SharedPreferences.Editor editor = sp.edit();  
 editor.putString("USER\_NAME", username);  
 editor.putString("PASSWORD", password);  
 editor.commit();  
 }  
 /\*\*  
 \* 获取用户信息model  
 \*  
 \* @param context  
 \* @param  
 \* @param  
 \*/  
 public UserInfoModel getUserInfo(Context context) {  
 SharedPreferences sp = context.getSharedPreferences("userInfo", Context.MODE\_PRIVATE);  
 UserInfoModel userInfo = new UserInfoModel();  
 userInfo.setLgname(sp.getString("USER\_NAME", ""));  
 userInfo.setLgpwd(sp.getString("PASSWORD", ""));  
 return userInfo;  
 }  
 /\*\*  
 \* userInfo中是否有数据  
 \*/  
 public boolean hasUserInfo(Context context) {  
 UserInfoModel userInfo = getUserInfo(context);  
 if (userInfo != null) {  
 if ((!TextUtils.isEmpty(userInfo.getLgname())) && (!TextUtils.isEmpty(userInfo.getLgpwd()))) {//有数据  
 return true;  
 } else {  
 return false;  
 }  
 }  
 return false;  
 }  
  
}

* 用户信息实体类

package ouc.edu.com.model;  
/\*\*  
 \* Created by JHC on 2017/11/22.  
 \* 用户信息Model  
 \*/  
public class UserInfoModel {  
 /\*\*  
 \*  
 \*/  
 private static final long serialVersionUID = 5790767399049546805L;  
 private int id; // id  
 private String lgname; // 登录名  
 private String lgpwd; // 登录密码  
 private String truename; // 真实姓名  
 private String role; // 角色  
 private String phone; // 电话  
 private String unit; // 单位  
 public int getId() {  
 return id;  
 }  
 public void setId(int id) {  
 this.id = id;  
 }  
 public String getLgname() {  
 return lgname;  
 }  
 public void setLgname(String lgname) {  
 this.lgname = lgname;  
 }  
 public String getLgpwd() {  
 return lgpwd;  
 }  
 public void setLgpwd(String lgpwd) {  
 this.lgpwd = lgpwd;  
 }  
 public String getTruename() {  
 return truename;  
 }  
 public void setTruename(String truename) {  
 this.truename = truename;  
 }  
 public String getRole() {  
 return role;  
 }  
 public void setRole(String role) {  
 this.role = role;  
 }  
 public String getPhone() {  
 return phone;  
 }  
 public void setPhone(String phone) {  
 this.phone = phone;  
 }  
 public String getUnit() {  
 return unit;  
 }  
 public void setUnit(String unit) {  
 this.unit = unit;  
 }  
}

* 登陆成功后写入

String userName = editText\_username.getText().toString();  
String userPwd = editText\_password.getText().toString();  
AutoMaticLogin.getInstance().saveUserInfo(LoginActivity.this, userName, userPwd);

* 判断是否可以自动登录

if (AutoMaticLogin.getInstance().hasUserInfo(context))//自动登录判断，SharePrefences中有数据，则跳转到主页，没数据则跳转到登录页  
{  
   
} else {  
   
}

5、退出自动登录

* 添加清除SharedPreferences函数

//清除SharedPreferences

public void clearSharedPreferences(Context context){

SharedPreferences sp = context.getSharedPreferences("userInfo", Context.MODE\_PRIVATE);

sp.edit().clear().commit();

}

* 调用清除数据AutoMaticLogin.getInstance().clearSharedPreferences(getContext());

System.exit(0);

6、开机自启动

* 新建一个接收类，并实现BroadcastRecevicer，重写Onreceive方法

Intent intent =new Intent(context,MyReceive.class);

intent.setFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK);

context.startActivity(intent);

* 在Manifest.xml注册此类，并添加

<intent-filter> <!—标明应用程序的入口程序，此标签在那个Activity下，应用程序就会先启动哪个Activity-->  
 <action android:name="android.intent.action.BOOT\_COMPLETED" />  
 <category android:name="android.intent.category.HOME" />  
</intent-filter>

注：视频链接：http://study.163.com/course/courseLearn.htm?courseId=270014#/learn/video?lessonId=377177&courseId=270014

7、Handler控制PorgressBar

* 启动Handler

handler.post(runnable);

* 创建Handler

Handler handler=new Handler(){

//接收消息

Public void handlerMessage(Message msg){

Pb.setProgess(msg.arg1);//其中pb为ProgresBar的对象

handler.post(runnable);

if(msg.arg1==100){

handler.removeCallbacks(runnable);

}

}

}

Runnable runnable=new Runnable(){

Int i=0;

Public void run(){

I+=10;

//Message是Android系统所提供的的消息对象

Message message=handler.obtainMessage();

//message.arg1是系统提供的，并且会节省系统资源消耗

message.arg1=i;

Try{

Thread.sleep(1000); //由于线程执行太快，故而可使线程睡眠1秒

}catach(InterruptedException e){

e.printStackTrace();

}

//发送消息

handler.sendMessage(message);

}

}

8、两次返回键退出程序

// 定义一个变量，来标识是否退出

private static boolean isExit = false;

Handler mHandler = new Handler() {

@Override

public void handleMessage(Message msg) {

super.handleMessage(msg);

isExit = false;

}

};

@Overrid

public boolean onKeyDown(int keyCode, KeyEvent event) {

if (keyCode == KeyEvent.KEYCODE\_BACK) {

exit();

return false;

}

return super.onKeyDown(keyCode, event);

}

private void exit() {

if (!isExit) {

isExit = true;

Toast.makeText(getApplicationContext(), "再按一次退出程序",

Toast.LENGTH\_SHORT).show();

// 利用handler延迟发送更改状态信息

mHandler.sendEmptyMessageDelayed(0, 2000);

} else {

finish();

System.exit(0);

}

}

9、两个界面传递List数据

* 第一个界面

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

info.add(name);

info.add(website);

info.add(weibo);

Intent intent = new Intent(MainActivity.this, ResultActivity.class);

intent.putStringArrayListExtra("infoList", info);

startActivity(intent);

* 第二个界面接收

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

infoList = getIntent().getStringArrayListExtra("infoList");

10、两个界面传递List<Integer>的方法

* 第一个界面

intent.putIntegerArrayListExtra(key, list);

startActivity(intent);

* 第二个界面接收

list =(ArrayList<Integer>) getIntent().getIntegerArrayListExtra(key);

11、两个界面传递Object

有两种方式来传递Object:Serializable和Parcelable

* 使用Serializable方式

前提：Object需要实现Serializable接口

用Serializable方式传递Object的语法：bundle.putSerializable(key,object);

用Serializable方式接收Object的语法：object=(Object) getIntent().getSerializableExtra(key);

实现Serializable接口就是把对象序列化，然后再传输，和Java的常用编程没什么明显区别，而且Object不需要明显改变,推荐用这种方式。

Object实现Serializable

package com.edu.ouc.model;

import java.io.Serializable;

/\*\*

\* OBJECT实现SERIALIZABLE

\* @author bixiaopeng 2013-2-18 上午11:32:19

\*/

public class SerInfo implements Serializable {

private String name;

private String website;

private String weibo;

public SerInfo(){}

public SerInfo(String name, String website, String weibo){

this.name = name;

this.website = website;

this.weibo = weibo;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getWebsite() {

return website;

}

public void setWebsite(String website) {

this.website = website;

}

public String getWeibo() {

return weibo;

}

public void setWeibo(String weibo) {

this.weibo = weibo;

}

}

* 用Serializable方式传递Object

SerInfo serInfo = new SerInfo(name, website, weibo);

Intent intent = new Intent();

Bundle bundle = new Bundle();

bundle.putSerializable("serinfo", serInfo);

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

intent.putExtras(bundle);

startActivity(intent);

* 用Serializable方式接收Object

//获得Serializable方式传过来的值

SerInfo serInfo = (SerInfo) getIntent().getSerializableExtra("serinfo");

注：参考连接：<http://blog.csdn.net/wirelessqa/article/details/8589200>

12、控件的自定义隐藏和显示

* 可见（visible)  
  XML文件：android:visibility="visible"  
  Java代码：view.setVisibility(View.VISIBLE);
* 不可见（invisible）  
  XML文件：android:visibility="invisible"  
  Java代码：view.setVisibility(View.INVISIBLE);
* 隐藏（GONE）  
  XML文件：android:visibility="gone"  
  Java代码：view.setVisibility(View.GONE);

注：当控件visibility属性为INVISIBLE时，textview依然会占据那部分位置；而控件属性为GONE时，界面则不保留textview控件所占有的空间

13、显示弹框-确定、取消

new AlertDialog.Builder(this).setTitle("提示")//设置对话框标题  
 .setMessage("请确认是否接收任务！")//设置显示的内容  
 .setPositiveButton("确定",new DialogInterface.OnClickListener() {//添加确定按钮  
 @Override  
 public void onClick(DialogInterface dialog, int which) {//确定按钮的响应事件  
 }  
 }).setNegativeButton("返回",new DialogInterface.OnClickListener() {//添加返回按钮  
 @Override  
 public void onClick(DialogInterface dialog, int which) {//响应事件  
 // TODO Auto-generated method stub  
 }  
 }).show();//在按键响应事件中显示此对话框

14、显示复选框弹框

AlertDialog.Builder builder = new AlertDialog.Builder(TaskInfoActivity.this);  
builder.setTitle("爱好");  
final String[] hobbies = {"篮球", "足球", "网球", "斯诺克"};  
// 设置一个单项选择下拉框  
/\*\*  
 \* 第一个参数指定我们要显示的一组下拉多选框的数据集合  
 \* 第二个参数代表哪几个选项被选择，如果是null，则表示一个都不选择，如果希望指定哪一个多选选项框被选择，  
 \* 需要传递一个boolean[]数组进去，其长度要和第一个参数的长度相同，例如 {true, false, false, true};  
 \* 第三个参数给每一个多选项绑定一个监听器  
 \*/  
builder.setMultiChoiceItems(hobbies, null, new DialogInterface.OnMultiChoiceClickListener()  
{  
 StringBuffer sb = new StringBuffer(100);  
 @Override  
 public void onClick(DialogInterface dialog, int which, boolean isChecked)  
 {  
 if(isChecked)  
 {  
 sb.append(hobbies[which] + ", ");  
 }  
 Toast.makeText(TaskInfoActivity.this, "爱好为：" + sb.toString(), Toast.LENGTH\_SHORT).show();  
 }  
});  
builder.setPositiveButton("确定", new DialogInterface.OnClickListener()  
{  
 @Override  
 public void onClick(DialogInterface dialog, int which)  
 {  
 }  
});  
builder.setNegativeButton("取消", new DialogInterface.OnClickListener()  
{  
 @Override  
 public void onClick(DialogInterface dialog, int which)  
 {  
 }  
});  
builder.show();

### 15、解决设置Android 23.0以上版本对SD卡的读写权限无效的问题

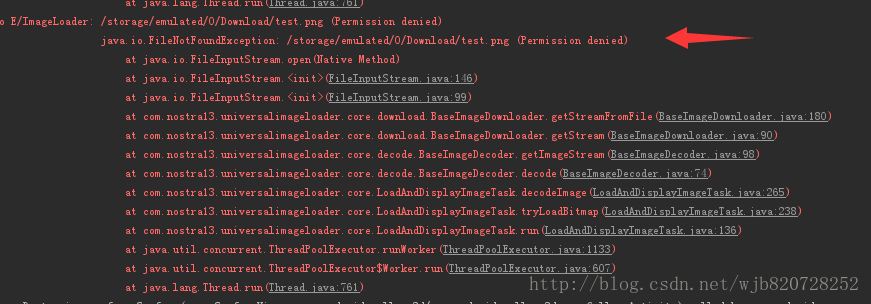
* 问题描述

对Android的SD卡进行读取权限设置时：

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"/>

结果报错：



* 修正方法：新建一个读写文件操作权限公共类

public class PermisionUtils {

// Storage Permissions

private static final int REQUEST\_EXTERNAL\_STORAGE = 1;

private static String[] PERMISSIONS\_STORAGE = {

Manifest.permission.READ\_EXTERNAL\_STORAGE,

Manifest.permission.WRITE\_EXTERNAL\_STORAGE};

/\*\*

\* Checks if the app has permission to write to device storage

\* If the app does not has permission then the user will be prompted to

\* grant permissions

\*

\* @param activity

\*/

public static void verifyStoragePermissions(Activity activity) {

// Check if we have write permission

int permission = ActivityCompat.checkSelfPermission(activity,

Manifest.permission.WRITE\_EXTERNAL\_STORAGE);

if (permission != PackageManager.PERMISSION\_GRANTED) {

// We don't have permission so prompt the user

ActivityCompat.requestPermissions(activity, PERMISSIONS\_STORAGE,

REQUEST\_EXTERNAL\_STORAGE);

}

}

}

* 在需要授权的地方调用：

PermisionUtils.verifyStoragePermissions(this);

16、解决Android N7以上版本打开摄像头失败

* 问题描述：

为了提高私有文件的安全性，在targetSdk版本为N或者以后版本的app中，其私有目录将会限制访问。这可以防止私有文件元数据的泄露，比如文件大小或者是文件是否存在。但这给开发者带来了很多不利的影响：

文件的owner不能放宽文件权限，如果你使用MODE\_WORLD\_READABLE或者 MODE\_WORLD\_WRITEABLE操作文件，将会触发SecurityException。

当你跨package域传递file://的URI时，接收者得到的将是一个无权访问的路径，因此，这将会触发FileUriExposedException。对于这类操作，官方推荐的方式是使用FileProvider，当然你也可以使用ContentProvider。这里只看文字理解起来可能有点小困难，所以我将以调用系统拍照为例说明下：在targetSdk为Android N之前的系统版本时，你可以使用如下方法调用系统相机拍照并存入指定路径中。

Intent intent = new Intent(MediaStore.ACTION\_IMAGE\_CAPTURE);

Uri uri = Uri.fromFile(sdcardTempFile);

intent.putExtra(MediaStore.EXTRA\_OUTPUT, uri);

然而在执行到这段代码时app就crash了，crash便是FileUriExposedException。

* 解决办法：

方法一：   
把代码修改下，使用ContentProvider方式传递uri，这样在Android N上便可以正常运行了。

Intent intent = new Intent(MediaStore.ACTION\_IMAGE\_CAPTURE);

ContentValues contentValues = new ContentValues(1);

contentValues.put(MediaStore.Images.Media.DATA, sdcardTempFile.getAbsolutePath());

Uri uri = context.getContentResolver().insert(MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI,contentValues);

方法二：

通过FileProvider解决，实例下载：<https://github.com/honjane/fileProviderDemo>

<http://blog.csdn.net/honjane/article/details/52057132>

17、okhttputils实现上传图片

* 服务器端

@ResponseBody

@RequestMapping(value = "/upload", method = RequestMethod.POST)

public void upload(HttpServletRequest request, HttpServletResponse response) throws Exception {

response.setHeader("Access-Control-Allow-Origin", "\*"); // 解决跨域请求

LOG.info("-----进入后台----upload接口---开始---");

//获取工厂类实例

DiskFileItemFactory factory = new DiskFileItemFactory();

//创建解析器类实例

ServletFileUpload fileUpload = new ServletFileUpload(factory);

//fileUpload通过该对象来限制文件的大小

//设置当文件的大小为50KB

//fileUpload.setFileSizeMax(1024\*50);

//设置多个文件的总大小为300mb

fileUpload.setSizeMax(1024\*1024\*300);

try {

//解析request

List<FileItem> fileItems = fileUpload.parseRequest(request);

//遍历fileItems，读取表单的信息

for (FileItem fileItem : fileItems) {

//判断当前表单项是否是一个普通表单项

if(fileItem.isFormField()){

//获取属性名

String fieldName = fileItem.getFieldName();

//获取属性值

String value = fileItem.getString("utf-8");

System.out.println(fieldName+" = "+value);

}else{

//如果是文件表单项

//获取文件的大小

long size = fileItem.getSize();

//判断size是否为0

if(size==0){

continue;

}

//获取文件的类型

String contentType = fileItem.getContentType();

//获取文件的名字

String name = fileItem.getName();

//判断name中是否包含有路径信息

if(name.contains("\\")){

//如果包含则截取字符串

name = name.substring(name.lastIndexOf("\\")+1);

}

//生成一个UUID，作为文件名的前缀

String prefix = UUID.randomUUID().toString().replace("-", "");

name = prefix+"\_"+name;

//获取表单项的属性名

String fieldName = fileItem.getFieldName();

System.out.println("文件的大小: "+size);

System.out.println("文件的类型: "+contentType);

System.out.println("文件的名字: "+name);

System.out.println("表单项name属性名: "+fieldName);

//获取ServletContext对象

ServletContext context = request.getServletContext();

//获取项目的真实路径

String path = context.getRealPath("/upload");

//判断路径是否存在

File file = new File(path);

if(!file.exists()){

//如果不存在该路径，则创建一个路径

file.mkdirs();

}

//将文件写入到磁盘中

fileItem.write(new File(path+"/"+name));

}

}

PrintWriter out=response.getWriter();

out.println("upload success!!!");//输出的内容要放在body中

//设置3秒钟跳转

//response.setHeader("refresh", "3;../test/index.jsp");

}catch(FileSizeLimitExceededException e){

//一但捕获到该异常，则说明单个文件大小超过限制。

//设置一个错误消息

request.setAttribute("msg", "单个文件大小请不要超过50KB");

//转发到index.jsp

request.getRequestDispatcher("test/testUploadFile.jsp").forward(request, response);

}catch(SizeLimitExceededException e){

//一但捕获到该异常，则说明单个文件大小超过限制。

//设置一个错误消息

request.setAttribute("msg", "所有文件大小请不要超过300mb");

//转发到index.jsp

request.getRequestDispatcher("test/testUploadFile.jsp").forward(request, response);

}

catch (FileUploadException e) {

e.printStackTrace();

} catch (Exception e) {

e.printStackTrace();

}

LOG.info("--后台----upload成功------");

}

* 客户端

引入依赖：compile 'com.zhy:okhttputils:2.6.2'

public String upoadFile(File file, String requestURL){  
 String result="";  
 Map<String, String> params = new HashMap<String, String>();  
 params.put("username", "ouccs-jhc");//上传的用户名,没有不用传参数  
 params.put("password", "123456");//上传的密码,没有不用传参数  
 String url = requestURL ;  
 OkHttpUtils.post()//  
 .addFile("mFile", "server\_afu.png", file)//可以上传多个文件  
 /\* .addFile("mFile", "server\_test.txt", file2)//\*/  
 .url(url)  
 .params(params)//  
 .build()//  
 .execute(new UpLoadSummaryActivity.MyStringCallback());  
 return result;  
}

UpLoadFileToServer upLoadFileToServer=new UpLoadFileToServer();  
upLoadFileToServer.upoadFile(new File(imagePaths.get(0)),"http://10.0.2.2:8080/TaskTrackingService/upload.do?");

//okhttp-utils的回调类  
public static class MyStringCallback extends StringCallback {  
 @Override  
 public void onBefore(Request request, int id) {  
 //setTitle("loading...");  
 }  
 @Override  
 public void onAfter(int id) {  
 // setTitle("Sample-okHttp");  
 }  
 @Override  
 public void onError(Call call, Exception e, int id) {  
 e.printStackTrace();  
 //tv\_result.setText("onError:" + e.getMessage());  
 }  
 @Override  
 public void onResponse(String response, int id) {  
 //Log.e(TAG, "onResponse：complete");  
 //tv\_result.setText("onResponse:" + response);  
 switch (id) {  
 case 100://http请求的响应码  
 //Toast.makeText(OKHttpActivity.this, "http", Toast.LENGTH\_SHORT).show();  
 break;  
 case 101://http请求的响应码  
 // Toast.makeText(OKHttpActivity.this, "https", Toast.LENGTH\_SHORT).show();  
 break;  
 }  
 }  
 @Override  
 public void inProgress(float progress, long total, int id) {  
 //Log.e(TAG, "inProgress:" + progress);  
 /\*mProgressBar.setProgress((int) (100 \* progress));\*/  
 }  
}

18、ScrollView中用ListView的事件冲突

* 问题描述：

屏幕要足够大，外面用的是ScrollView，里面有一个评论框，用的是ListView，遇到的问题是

1、listView中数据显示不全

2、listView的滚动和ScrollView的滚动冲突

* 解决办法：

1、给ListView设置固定高度 android:layout\_height="220dp"

2、给ScrollView 设置 android:fillViewport="true"

     在listView的onTouch里面拦截ScrollView的事件，使得scrollView不可以获得点击事件，这样只有listView获得点击事件！

 commentlv.setOnTouchListener(new OnTouchListener() {

 @Override public boolean onTouch(View arg0, MotionEvent arg1) {

 // TODO Auto-generated method stub

 sv.requestDisallowInterceptTouchEvent(true);

return false;

    }

 });

19、ScrollView布局初始化显示的时候使页面置顶

* 在XML中将ScrollView布局中添加如下两行即可。

  android:focusableInTouchMode="true"

  android:focusable="true"

* 在Android程序中添加如下代码，注意如果scrollview中嵌套有listview，gridview之类的控件，请将下列代码放在listview初始化之后的位置。

//设置scrollview初始化后滑动到顶部，必须在gridview填充数据之后，否则无法实现预期效果

scv\_first.smoothScrollTo(0,20);

scv\_first.setFocusable(true);

20、去掉标题栏

* 方法一：在代码里实现

this.requestWindowFeature(Window.FEATURE\_NO\_TITLE);//去掉标题栏

备注：放在在setContentView()前面

* 方法二：在清单文件中实现

<application android:icon="@drawable/icon"

       android:label="@string/app\_name"

         android:theme="@android:style/Theme.NoTitleBar">

备注：这样用可以将整个应用设置成无标题栏，如果只需要在一个Activity设置成一个无标题栏的形式，只要把上面的第三行代码写到某一个Activity里面就可以了

* 方法三：在style.xml中定义，之后引用或。

在style.xml中定义

<?xml version="1.0" encoding="UTF-8" ?>

<resources>

    <style name="notitle">

        <item name="android:windowNoTitle">true</item>

    </style>

</resources>

然后面manifest.xml中引用就可以了，这种方法稍麻烦了些。

<application android:icon="@drawable/icon"

         android:label="@string/app\_name"

         android:theme="@style/notitle">

备注：其实可以看得出来，第二种方法和第三种方法实质是一样的，只不过第二种方法调用的是系统定义好的style.xml文件，而第三种方法则是在自己的应用里定义style.xml，然后再自己再调用，其实道理是一样的，第三种方法做起来更有成就感。

21、使用glide加载图片（客户端+服务器端）

* 加载glide依赖

compile 'com.github.bumptech.glide:glide:3.7.0'

* 增加布局文件

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:background="#000"

>

<GridView

android:id="@+id/grid"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:numColumns="auto\_fit"

android:columnWidth="80dp"

android:stretchMode="columnWidth"

></GridView>

</LinearLayout>

GridView属性的含义：

android:numColumns="auto\_fit" --------列数设置为自动

android:columnWidth="90dp"，----------每列的宽度，也就是Item的宽度

android:stretchMode="columnWidth"------缩放与列宽大小同步

android:verticalSpacing="10dp"----------垂直边距

android:horizontalSpacing="10dp"-------水平边距

* 新建image显示文件item\_image.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:background="@color/white"  
 android:orientation="vertical">  
 <ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="100dp"  
 android:layout\_height="100dp"  
 android:src="@mipmap/ic\_launcher"  
 android:layout\_gravity="center\_horizontal" />  
</LinearLayout>

* 新建文件适配器

package com.edu.ouc.adapter;  
import android.content.Context;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.BaseAdapter;  
import android.widget.CheckBox;  
import android.widget.ImageView;  
import com.bumptech.glide.Glide;  
import com.edu.ouc.tasktracking.R;  
import java.util.ArrayList;  
import java.util.List;  
/\*\*  
 \* Created by JHC on 2017/12/7.  
 \*/  
public class ImageAdapter extends BaseAdapter {  
 private Context context;  
 private List<String> urls = new ArrayList<>();  
 public ImageAdapter(Context context,List<String> urls) {  
 this.context = context;  
 this.urls = urls;  
 }  
 public void setUrls(List<String> urls) {  
 this.urls = urls;  
 }  
 @Override  
 public int getCount() {  
 return urls.size();  
 }  
 @Override  
 public String getItem(int i) {  
 return urls.get(i);  
 }  
 @Override  
 public long getItemId(int i) {  
 return i;  
 }  
 @Override  
 public View getView(int i, View view, ViewGroup viewGroup) {  
 ViewHolder vh = null;  
 if(view == null){  
 view = LayoutInflater.from(context).inflate(R.layout.item\_image,null);  
 vh = new ViewHolder();  
 vh.imageView = (ImageView) view.findViewById(R.id.imageView);  
 vh.checkBox = (CheckBox) view.findViewById(R.id.checkbox);  
 view.setTag(vh);  
 }  
 vh = (ViewHolder) view.getTag();  
 if(urls!=null && urls.size()>0){  
 Glide.with(context).load("http://10.0.2.2:8080/TaskTrackingService/showImg.do?filePath="+urls.get(i)).centerCrop().into(vh.imageView);  
 }  
 return view;  
 }  
 class ViewHolder{  
 ImageView imageView;  
 CheckBox checkBox;  
 }  
}

* 在Activity中加载适配器

private ImageAdapter imageAdapter;

private GridView gridView\_showImg;

gridView\_showImg=(GridView)findViewById(R.id.grid);

imageAdapter = new ImageAdapter(getApplicationContext(),filePathList);  
gridView\_showImg.setAdapter(imageAdapter);

* 服务器端代码

@RequestMapping(value = "/showImg", method = RequestMethod.GET)

public void show(String filePath, HttpServletResponse response) throws IOException {

LOG.info("-----进入后台----show接口--" + filePath);

File file = getFile("D:/taskteacking/"+filePath);

response.setDateHeader("Expires", System.currentTimeMillis() + 1000 \* 60 \* 60 \* 24);

response.setHeader("Cache-Control", "max-age=60");

OutputStream os = response.getOutputStream();

FileInputStream is = null;

try {

is = new FileInputStream(file);

IOUtils.copy(is, os);

} catch (FileNotFoundException e) {

response.setStatus(404);

return;

} finally {

if (null != is) {

is.close();

}

if (null != os) {

os.flush();

os.close();

}

}

}

protected File getFile(String path) {

File file = new File(path);

return file;

}

22、点击图片放大

* 新建布局文件activity\_bigimage.xml

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/activity\_image"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
 <ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:id="@+id/photoView"/>  
</RelativeLayout>

* 点击图片事件

view.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View v) {  
 //点击进入详情界面  
 Intent intent = new Intent(context, BigImageActivity.class);  
 Bundle bundle = new Bundle();  
 bundle.putSerializable("URL", "http://10.0.2.2:8080/TaskTrackingService/showImg.do?filePath="+urls.get(i));  
 intent.putExtras(bundle);  
 context.startActivity(intent);  
 }  
});

* 新建activity文件 BigImageActivity

package com.edu.ouc.activity;  
import android.app.Activity;  
import android.net.Uri;  
import android.os.Bundle;  
import android.support.annotation.Nullable;  
import android.widget.ImageView;  
import android.widget.Toast;  
import com.bumptech.glide.Glide;  
import com.edu.ouc.tasktracking.R;  
import uk.co.senab.photoview.PhotoView;  
/\*\*  
 \* Created by JHC on 2017/12/8.  
 \*/  
public class BigImageActivity extends Activity {  
 private PhotoView photoView;  
 @Override  
 protected void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_bigimage);  
 photoView = (PhotoView) findViewById(R.id.photoView);  
 String url = getIntent().getStringExtra("URL");  
 photoView.setScaleType(ImageView.ScaleType.FIT\_START);  
 try {  
 Glide.with(this).load(url).centerCrop().into(photoView);  
 }catch (Exception e){  
 Toast.makeText(getApplicationContext(), "图片加载失败。。。", Toast.LENGTH\_SHORT).show();  
 }  
 }  
}

23、Glide下载图片

* 添加依赖

compile 'com.github.bumptech.glide:glide:3.5.2' //加载图片

* Glide保存图片函数

//Glide保存图片  
public void savePicture(final String fileName, String url){  
 Glide.with(this).load(url).asBitmap().toBytes().into(new SimpleTarget<byte[]>() {  
 @Override  
 public void onResourceReady(byte[] bytes, GlideAnimation<? super byte[]> glideAnimation) {  
 try {  
 savaFileToSD(fileName,bytes);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 });  
}  
//往SD卡写入文件的方法  
public void savaFileToSD(String filename, byte[] bytes) throws Exception {  
 //如果手机已插入sd卡,且app具有读写sd卡的权限  
 if (Environment.getExternalStorageState().equals(Environment.MEDIA\_MOUNTED)) {  
 String filePath = Environment.getExternalStorageDirectory().getCanonicalPath()+"/Pictures";  
 File dir1 = new File(filePath);  
 if (!dir1.exists()){  
 dir1.mkdirs();  
 }  
 filename = filePath+ "/" + filename;  
 imgePath=filename;  
 //这里就不要用openFileOutput了,那个是往手机内存中写数据的  
 FileOutputStream output = new FileOutputStream(filename);  
 output.write(bytes);  
 //将bytes写入到输出流中  
 output.close();  
 //关闭输出流  
 Toast.makeText(this, "图片已成功保存到"+filePath, Toast.LENGTH\_SHORT).show();  
 //发送广播，告知其重新加载SD卡  
 MediaScannerConnection.scanFile(this, new String[]{imgePath}, null,null);  
 } else Toast.makeText(this, "SD卡不存在或者不可读写", Toast.LENGTH\_SHORT).show();  
}

* 调用

SimpleDateFormat simpleDateFormat=new SimpleDateFormat("yyyyMMddHHmmss");  
String currentDate=simpleDateFormat.format(new Date());  
savePicture(currentDate+".jpg",url);

24、长按屏幕弹框并做出响应

private RelativeLayout relativeLayout\_activity\_image; //用于长按弹出对话框

relativeLayout\_activity\_image=(RelativeLayout)findViewById(R.id.activity\_image);  
relativeLayout\_activity\_image.setOnCreateContextMenuListener(new View.OnCreateContextMenuListener() {  
 @Override  
 public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {  
 menu.add(0, 0, 0, "下载");  
 menu.add(0, 1, 0, "取消");  
 menu.add(0, 2, 0, "你猜");  
 }  
});

//点击下载图片之后的响应方法  
@Override  
public boolean onContextItemSelected(MenuItem aItem) {  
 ContextMenu.ContextMenuInfo menuInfo = (ContextMenu.ContextMenuInfo) aItem.getMenuInfo();  
 /\* Switch on the ID of the item, to get what the user selected. \*/  
 switch (aItem.getItemId()) {  
 case 0:  
 Toast.makeText(getApplicationContext(), "点击了下载", Toast.LENGTH\_SHORT).show();  
 break;  
 case 1:  
 Toast.makeText(getApplicationContext(), "点击了取消", Toast.LENGTH\_SHORT).show();  
 break;  
 case 2:  
 Toast.makeText(getApplicationContext(), "点击了你猜", Toast.LENGTH\_SHORT).show();  
 break;  
 default:  
 break;  
 }  
 return false;  
}

25、自动生成验证码

* 布局文件

<LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="15dp"  
 android:layout\_marginRight="15dp"  
 android:layout\_marginTop="20dp"  
 android:orientation="horizontal" >  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="40dp">  
 <EditText  
 android:id="@+id/et\_Codes"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:maxLines="1"  
 android:background="@null"  
 android:textColor="#ff3b3b3b"  
 android:textSize="20.0sp"  
 android:hint="请输入右侧验证码" />  
 </LinearLayout>  
 <ImageView  
 android:id="@+id/iv\_showCode"  
 android:layout\_width="150dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_height="match\_parent" />  
</LinearLayout>

* 验证码管理类

package com.edu.ouc.function;  
import android.graphics.Bitmap;  
import android.graphics.Canvas;  
import android.graphics.Color;  
import android.graphics.Paint;  
import java.util.Random;  
/\*\*  
 \* Created by JHC on 2017/12/9.  
 \* 验证码生成函数  
 \*/  
public class CodeProduce {  
 //随机数数组  
 private static final char[] CHARS = {  
 '2', '3', '4', '5', '6', '7', '8', '9',  
 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'j', 'k', 'm',  
 'n', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z',  
 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',  
 'N', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'  
 };  
  
 private static CodeProduce bmpCode;  
  
 public static CodeProduce getInstance() {  
 if(bmpCode == null)  
 bmpCode = new CodeProduce();  
 return bmpCode;  
 }  
 //default settings  
 //验证码默认随机数的个数  
 private static final int DEFAULT\_CODE\_LENGTH = 4;  
 //默认字体大小  
 private static final int DEFAULT\_FONT\_SIZE = 25;  
 //默认线条的条数  
 private static final int DEFAULT\_LINE\_NUMBER = 5;  
 //padding值  
 private static final int BASE\_PADDING\_LEFT = 10, RANGE\_PADDING\_LEFT = 15, BASE\_PADDING\_TOP = 15, RANGE\_PADDING\_TOP = 20;  
 //验证码的默认宽高  
 private static final int DEFAULT\_WIDTH = 100, DEFAULT\_HEIGHT = 40;  
 //settings decided by the layout xml  
 //canvas width and height  
 private int width = DEFAULT\_WIDTH, height = DEFAULT\_HEIGHT;  
 //random word space and pading\_top  
 private int base\_padding\_left = BASE\_PADDING\_LEFT, range\_padding\_left = RANGE\_PADDING\_LEFT,  
 base\_padding\_top = BASE\_PADDING\_TOP, range\_padding\_top = RANGE\_PADDING\_TOP;  
 //number of chars, lines; font size  
 private int codeLength = DEFAULT\_CODE\_LENGTH, line\_number = DEFAULT\_LINE\_NUMBER, font\_size = DEFAULT\_FONT\_SIZE;  
 //variables  
 private String code;  
 private int padding\_left, padding\_top;  
 private Random random = new Random();  
 //验证码图片  
 public Bitmap createBitmap() {  
 padding\_left = 0;  
 Bitmap bp = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB\_8888);  
 Canvas c = new Canvas(bp);  
 code = createCode();  
 c.drawColor(Color.WHITE);  
 Paint paint = new Paint();  
 paint.setAntiAlias(true);  
 paint.setTextSize(font\_size);  
 //画验证码  
 for (int i = 0; i < code.length(); i++) {  
 randomTextStyle(paint);  
 randomPadding();  
 c.drawText(code.charAt(i) + "", padding\_left, padding\_top, paint);  
 }  
 //画线条  
 for (int i = 0; i < line\_number; i++) {  
 drawLine(c, paint);  
 }  
 c.save( Canvas.ALL\_SAVE\_FLAG );//保存  
 c.restore();//  
 return bp;  
 }  
 public String getCode() {  
 return code;  
 }  
 //生成验证码  
 private String createCode() {  
 StringBuilder buffer = new StringBuilder();  
 for (int i = 0; i < codeLength; i++) {  
 buffer.append(CHARS[random.nextInt(CHARS.length)]);  
 }  
 return buffer.toString();  
 }  
 //画干扰线  
 private void drawLine(Canvas canvas, Paint paint) {  
 int color = randomColor();  
 int startX = random.nextInt(width);  
 int startY = random.nextInt(height);  
 int stopX = random.nextInt(width);  
 int stopY = random.nextInt(height);  
 paint.setStrokeWidth(1);  
 paint.setColor(color);  
 canvas.drawLine(startX, startY, stopX, stopY, paint);  
 }  
 //生成随机颜色  
 private int randomColor() {  
 return randomColor(1);  
 }  
 private int randomColor(int rate) {  
 int red = random.nextInt(256) / rate;  
 int green = random.nextInt(256) / rate;  
 int blue = random.nextInt(256) / rate;  
 return Color.rgb(red, green, blue);  
 }  
 //随机生成文字样式，颜色，粗细，倾斜度  
 private void randomTextStyle(Paint paint) {  
 int color = randomColor();  
 paint.setColor(color);  
 paint.setFakeBoldText(random.nextBoolean()); //true为粗体，false为非粗体  
 float skewX = random.nextInt(11) / 10;  
 skewX = random.nextBoolean() ? skewX : -skewX;  
 paint.setTextSkewX(skewX); //float类型参数，负数表示右斜，整数左斜  
 //paint.setUnderlineText(true); //true为下划线，false为非下划线  
 //paint.setStrikeThruText(true); //true为删除线，false为非删除线  
 }  
 //随机生成padding值  
 private void randomPadding() {  
 padding\_left += base\_padding\_left + random.nextInt(range\_padding\_left);  
 padding\_top = base\_padding\_top + random.nextInt(range\_padding\_top);  
 }  
}

* 调用

iv\_showCode = (ImageView) findViewById(R.id.iv\_showCode);  
//将验证码用图片的形式显示出来  
iv\_showCode.setImageBitmap(CodeProduce.getInstance().createBitmap());  
realCode = CodeProduce.getInstance().getCode().toLowerCase();  
iv\_showCode.setOnClickListener(this);

//点击函数

case R.id.iv\_showCode: iv\_showCode.setImageBitmap(CodeProduce.getInstance().createBitmap());  
 realCode = CodeProduce.getInstance().getCode().toLowerCase();  
 break;

26、时间倒计时

* 时间倒计时管理类

package com.edu.ouc.function;  
import android.os.Handler;  
import android.os.Message;  
import android.os.SystemClock;  
/\*\*  
 \* Created by JHC on 2017/12/9.  
 \* 时间倒计时  
 \*/  
public abstract class CountDownTimer {  
 /\*\*  
 \* Millis since epoch when alarm should stop.  
 \*/  
 private final long mMillisInFuture;  
 /\*\*  
 \* The interval in millis that the user receives callbacks  
 \*/  
 private final long mCountdownInterval;  
 private long mStopTimeInFuture;  
 /\*\*  
 \* @param millisInFuture The number of millis in the future from the call  
 \* to {@link #start()} until the countdown is done and {@link #onFinish()}  
 \* is called.  
 \* @param countDownInterval The interval along the way to receive  
 \* {@link #onTick(long)} callbacks.  
 \*/  
 public CountDownTimer(long millisInFuture, long countDownInterval) {  
 mMillisInFuture = millisInFuture;  
 mCountdownInterval = countDownInterval;  
 }  
 /\*\*  
 \* Cancel the countdown.  
 \*/  
 public final void cancel() {  
 mHandler.removeMessages(MSG);  
 }  
 /\*\*  
 \* Start the countdown.  
 \*/  
 public synchronized final CountDownTimer start() {  
 if (mMillisInFuture <= 0) {  
 onFinish();  
 return this;  
 }  
 mStopTimeInFuture = SystemClock.elapsedRealtime() + mMillisInFuture;  
 mHandler.sendMessage(mHandler.obtainMessage(MSG));  
 return this;  
 }  
 /\*\*  
 \* Callback fired on regular interval.  
 \* @param millisUntilFinished The amount of time until finished.  
 \*/  
 public abstract void onTick(long millisUntilFinished);  
 /\*\*  
 \* Callback fired when the time is up.  
 \*/  
 public abstract void onFinish();  
 private static final int MSG = 1;  
 // handles counting down  
 private Handler mHandler = new Handler() {  
 @Override  
 public void handleMessage(Message msg) {  
 synchronized (CountDownTimer.this) {  
 final long millisLeft = mStopTimeInFuture - SystemClock.elapsedRealtime();  
 if (millisLeft <= 0) {  
 onFinish();  
 } else if (millisLeft < mCountdownInterval) {  
 // no tick, just delay until done  
 sendMessageDelayed(obtainMessage(MSG), millisLeft);  
 } else {  
 long lastTickStart = SystemClock.elapsedRealtime();  
 onTick(millisLeft);  
 // take into account user's onTick taking time to execute  
 long delay = lastTickStart + mCountdownInterval - SystemClock.elapsedRealtime();  
 // special case: user's onTick took more than interval to  
 // complete, skip to next interval  
 while (delay < 0) delay += mCountdownInterval;  
 sendMessageDelayed(obtainMessage(MSG), delay);  
 }  
 }  
 }  
 };  
}

* 调用

package com.edu.ouc.activity;  
import android.app.Activity;  
import android.os.Bundle;  
import android.support.annotation.Nullable;  
import android.widget.Button;  
import android.widget.EditText;  
import com.edu.ouc.function.CountDownTimer;  
import com.edu.ouc.tasktracking.R;  
/\*\*  
 \* Created by JHC on 2017/12/9.  
 \* 输入邮箱验证码  
 \*/  
public class EmailCodeActivity extends Activity {  
 private MyCountDownTimer mc;  
 private EditText editText\_emailCode;  
 private Button button\_email\_ok;  
 @Override  
 protected void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_email); editText\_emailCode=(EditText)findViewById(R.id.edt\_email\_code); button\_email\_ok=(Button)findViewById(R.id.btn\_email\_ok);  
 mc = new MyCountDownTimer(30000, 1000);  
 mc.start();  
 countDown(); //倒计时  
 }  
 private void countDown(){  
 }  
 /\*\*  
 \* 继承 CountDownTimer 防范  
 \*  
 \* 重写 父类的方法 onTick() 、 onFinish()  
 \*/  
 class MyCountDownTimer extends CountDownTimer {  
 /\*\*  
 \*  
 \* @param millisInFuture  
 \* 表示以毫秒为单位 倒计时的总数  
 \*  
 \* 例如 millisInFuture=1000 表示1秒  
 \*  
 \* @param countDownInterval  
 \* 表示 间隔 多少微秒 调用一次 onTick 方法  
 \*  
 \* 例如: countDownInterval =1000 ; 表示每1000毫秒调用一次onTick()  
 \*  
 \*/  
 public MyCountDownTimer(long millisInFuture, long countDownInterval) {  
 super(millisInFuture, countDownInterval);  
 }  
 @Override  
 public void onFinish() {  
 button\_email\_ok.setText("done");  
 }  
 @Override  
 public void onTick(long millisUntilFinished) {  
 button\_email\_ok.setText("倒计时(" + millisUntilFinished / 1000 + ")...");  
 }  
 }  
}

27、发送邮件

* 导入jar包

activation.jar、addition.jar、mail.jar

* 邮箱管理类

private static String myEmailAccount = "jhchuang@126.com";  
private static String myEmailPassword = "\*\*\*\*\*\*\*\*\*";  
// 发件人邮箱的 SMTP 服务器地址, 必须准确, 不同邮件服务器地址不同, 一般(只是一般, 绝非绝对)格式为: smtp.xxx.com  
// 网易163邮箱的 SMTP 服务器地址为: smtp.163.com  
private static String myEmailSMTPHost = "smtp.126.com";

public void sendEmail(){  
 // 这个类主要是设置邮件  
 new Thread(new Runnable() {  
 @Override  
 public void run() {  
 String receiveMailAccount = "614277037@qq.com";  
 SendMail(myEmailAccount, myEmailPassword, myEmailSMTPHost, receiveMailAccount);  
 }  
 }).start();}  
 public static void SendMail(String myEmailAccount, String myEmailPassword, String myEmailSMTPHost,  
 String receiveMailAccount) {  
 // TODO Auto-generated method stub  
 // 1. 创建参数配置, 用于连接邮件服务器的参数配置  
 Properties props = new Properties(); // 参数配置  
 props.setProperty("mail.transport.protocol", "smtp"); // 使用的协议（JavaMail规范要求）  
 props.setProperty("mail.smtp.host", myEmailSMTPHost); // 发件人的邮箱的 SMTP  
 props.setProperty("mail.smtp.auth", "true"); // 需要请求认证  
 // PS: 某些邮箱服务器要求 SMTP 连接需要使用 SSL 安全认证 (为了提高安全性, 邮箱支持SSL连接, 也可以自己开启),  
 // 如果无法连接邮件服务器, 仔细查看控制台打印的 log, 如果有有类似 “连接失败, 要求 SSL 安全连接” 等错误,  
 // 打开下面 /\* ... \*/ 之间的注释代码, 开启 SSL 安全连接。  
/\*  
 \* // SMTP 服务器的端口 (非 SSL 连接的端口一般默认为 25, 可以不添加, 如果开启了 SSL 连接, //  
 \* 需要改为对应邮箱的 SMTP 服务器的端口, 具体可查看对应邮箱服务的帮助, // QQ邮箱的SMTP(SLL)端口为465或587,  
 \* 其他邮箱自行去查看) final String smtpPort = "465";  
 \* props.setProperty("mail.smtp.port", smtpPort);  
 \* props.setProperty("mail.smtp.socketFactory.class",  
 \* "javax.net.ssl.SSLSocketFactory");  
 \* props.setProperty("mail.smtp.socketFactory.fallback", "false");  
 \* props.setProperty("mail.smtp.socketFactory.port", smtpPort);  
 \*/  
 // 2. 根据配置创建会话对象, 用于和邮件服务器交互  
 Session session = Session.getDefaultInstance(props);  
 session.setDebug(true); // 设置为debug模式, 可以查看详细的发送 log  
 // 3. 创建一封邮件  
 try {  
 // 1. 创建一封邮件  
 MimeMessage message = new MimeMessage(session);  
 // 2. From: 发件人（昵称有广告嫌疑，避免被邮件服务器误认为是滥发广告以至返回失败，请修改昵称）  
 message.setFrom(new InternetAddress(myEmailAccount, "潍坊孚日集团有限公司", "UTF-8"));  
 // 3. To: 收件人（可以增加多个收件人、抄送、密送）  
 message.setRecipient(MimeMessage.RecipientType.TO, new InternetAddress(receiveMailAccount, "某用户", "UTF-8"));  
 // 4. Subject: 邮件主题（标题有广告嫌疑，避免被邮件服务器误认为是滥发广告以至返回失败，请修改标题）  
 message.setSubject("任务跟踪-找回密码", "UTF-8");  
 // 5. Content:  
 // 邮件正文（可以使用html标签）（内容有广告嫌疑，避免被邮件服务器误认为是滥发广告以至返回失败，请修改发送内容）  
 message.setContent("尊敬的用户您好, 本邮件来自\*\*\*系统自动发送的邮件。<br>"+"验证码为：", "text/html;charset=UTF-8");  
 // 6. 设置发件时间  
 message.setSentDate(new Date());  
 // 7. 保存设置  
 message.saveChanges();  
 // 4. 根据 Session 获取邮件传输对象  
 Transport transport = session.getTransport();  
 // 5. 使用 邮箱账号 和 密码 连接邮件服务器, 这里认证的邮箱必须与 message 中的发件人邮箱一致, 否则报错  
 //  
 // PS\_01: 成败的判断关键在此一句, 如果连接服务器失败, 都会在控制台输出相应失败原因的 log,  
 // 仔细查看失败原因, 有些邮箱服务器会返回错误码或查看错误类型的链接, 根据给出的错误  
 // 类型到对应邮件服务器的帮助网站上查看具体失败原因。  
 //  
 // PS\_02: 连接失败的原因通常为以下几点, 仔细检查代码:  
 // (1) 邮箱没有开启 SMTP 服务;  
 // (2) 邮箱密码错误, 例如某些邮箱开启了独立密码;  
 // (3) 邮箱服务器要求必须要使用 SSL 安全连接;  
 // (4) 请求过于频繁或其他原因, 被邮件服务器拒绝服务;  
 // (5) 如果以上几点都确定无误, 到邮件服务器网站查找帮助。  
 //  
 // PS\_03: 仔细看log, 认真看log, 看懂log, 错误原因都在log已说明。  
 transport.connect(myEmailAccount, myEmailPassword);  
 // 6. 发送邮件, 发到所有的收件地址, message.getAllRecipients()  
 // 获取到的是在创建邮件对象时添加的所有收件人, 抄送人, 密送人  
 transport.sendMessage(message, message.getAllRecipients());  
 // 7. 关闭连接  
 transport.close();  
 } catch (Exception e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 }  
 }

28、随机生成6位数字

String.valueOf((int) ((Math.random() \* 9 + 1) \* 100000))

29、新闻列表

* 主页面

<**RelativeLayout** xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="com.example.administrator.wechat.fragment.DiscoveryFragment">  
 <!-- TODO: Update blank fragment layout -->  
 <ListView  
 android:id="@+id/lv\_task\_ok"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" />  
</RelativeLayout>

* 布局文件：

ListView listview\_fragmeng\_task.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_margin="10dp">  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="60dp"  
 android:layout\_margin="10dp"  
 android:orientation="horizontal"  
 android:gravity="center">  
 <TextView  
 android:id="@+id/tv\_tis"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:singleLine="true"  
 android:text="[已完成]"  
 android:textSize="20dp" />  
 </LinearLayout>  
 <LinearLayout  
 android:layout\_marginLeft="25dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="60dp"  
 android:layout\_margin="10dp"  
 android:orientation="vertical"  
 >  
 <TextView  
 android:id="@+id/tv\_title"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:singleLine="true"  
 android:text="11111"  
 android:textSize="18dp"/>  
 <TextView  
 android:id="@+id/tv\_info"  
 android:layout\_marginTop="10dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="11111"  
 android:textSize="14dp"/>  
 </LinearLayout>  
 </LinearLayout>

* 自定义适配器

package com.edu.ouc.adapter;  
import android.content.Context;  
import android.support.v7.widget.RecyclerView;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.BaseAdapter;  
import android.widget.ImageView;  
import android.widget.TextView;  
import android.widget.Toast;  
import com.edu.ouc.model.TaskShowModel;  
import com.edu.ouc.tasktracking.R;  
import java.util.ArrayList;  
import java.util.List;  
/\*\*  
 \* Created by JHC on 2017/11/23.  
 \* 这里主要是为ListView写一个填充数据的适配器,通过这个适配器来确定ListView上每个Item显示的数据.  
 \*/  
public class TaskBaseAdapter extends BaseAdapter {  
 private List<TaskShowModel> datas = new ArrayList<TaskShowModel>();//新闻列表集合  
 private Context context;  
 public TaskBaseAdapter(Context context, List<TaskShowModel> datas) {  
 this.datas = datas;  
 this.context = context;  
 }  
 @Override  
 public int getCount() {  
 return datas.size(); //返回列表的长度  
 }  
 @Override  
 public TaskShowModel getItem(int position) {  
 return datas.get(position); //通过列表的位置 获得集合中的对象  
 }  
 @Override  
 public long getItemId(int position) { // 获得集合的Item的位  
 return position;  
 }  
 @Override  
 public View getView(final int position, View convertView, ViewGroup parent) {  
 ViewHolder vh;  
 if(convertView==null){  
 vh = new ViewHolder();  
 convertView = LayoutInflater.from(context).inflate(R.layout.listview\_fragment\_task,null);  
 vh.tvTitle = (TextView) convertView.findViewById(R.id.tv\_title);  
 vh.tvDate = (TextView) convertView.findViewById(R.id.tv\_info);  
 convertView.setTag(vh);  
 }else {  
 vh = (ViewHolder) convertView.getTag();  
 }  
 convertView.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Toast.makeText(context, "点击了"+datas.get(position).getTasktitle(),Toast.LENGTH\_SHORT).show();  
 /\* Intent intent = new Intent(context, Main2Activity.class);  
 context.startActivity(intent);\*/  
 }  
 });  
 /\*vh.img.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Toast.makeText(context,"点击图片",Toast.LENGTH\_SHORT).show();  
 }  
 });\*/  
 TaskShowModel taskShowModel = datas.get(position);  
 vh.tvTitle.setText(taskShowModel.getTasktitle());  
 vh.tvDate.setText(taskShowModel.getStartDate());  
 return convertView;  
 }  
 protected class ViewHolder {  
 private TextView tvTitle;  
 private TextView tvDate;  
 }  
}

* framment

package com.edu.ouc.fragment;  
import android.app.Activity;  
import android.content.Context;  
import android.os.Bundle;  
import android.support.annotation.Nullable;  
import android.support.v4.app.Fragment;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ListView;  
import com.edu.ouc.adapter.TaskBaseAdapter;  
import com.edu.ouc.model.TaskShowModel;  
import com.edu.ouc.tasktracking.R;  
import java.util.ArrayList;  
import java.util.List;  
/\*\*  
 \* A simple {@link Fragment} subclass.  
 \*/  
public class TaskFragment extends Fragment {  
 private Activity activity;  
 private ListView listView;  
 private List<TaskShowModel> datas; //新闻集合对象  
 private TaskBaseAdapter adapter; //自定义适配器对象  
 public TaskFragment() {  
 // Required empty public constructor  
 }  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 // Inflate the layout for this fragment  
 return inflater.inflate(R.layout.fragment\_task, container, false);  
 }  
 @Override  
 public void onActivityCreated(@Nullable Bundle savedInstanceState) {  
 super.onActivityCreated(savedInstanceState);  
 listView=(ListView)getView().findViewById(R.id.lv\_task\_ok);  
 datas=new ArrayList<TaskShowModel>();  
 getDatas();  
 /\*\*  
 \* 实例化Adapter对象(注意:必须要写在在getDatas() 方法后面,不然datas中没有数据)  
 \*/  
 adapter = new TaskBaseAdapter(getActivity(), datas);  
 //将适配器变量的内容加载到List里(也就是把那一堆新闻都放了进去)  
 listView.setAdapter(adapter);  
 }  
 /\*\*  
 \* 通过接口获取新闻列表的方法  
 \* @param url  
 \*/  
 public void getDatas(){  
 for (int i=0;i<20;i++){  
 TaskShowModel taskShowModel=new TaskShowModel();  
 taskShowModel.setTasktitle("文章标题"+i);  
 taskShowModel.setStartDate("时间："+i);  
 datas.add(taskShowModel);  
 }  
 }  
}

30、打开和关闭按钮



* layout\_activity.xml:

<Switch

        android:id="@+id/activity\_setServe\_switch\_open"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:textOff=""

        android:layout\_marginTop="100dp"

        android:layout\_marginLeft="100dp"

        android:textOn=""

        android:thumb="@drawable/switch\_thumb"

        android:track="@drawable/switch\_track" />

* switch\_thumb.xml

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

    android:shape="oval">

    <size

        android:width="30dp"

        android:height="30dp">

    </size>

    <solid

        android:color="@android:color/white">

    </solid>

</shape>

* switch\_track.xml

<?xml version="1.0" encoding="utf-8"?>

<selector xmlns:android="http://schemas.android.com/apk/res/android">

    <item

        android:state\_checked="true"

        android:drawable="@drawable/switch\_track\_on"/>

    <item

        android:state\_checked="false"

        android:drawable="@drawable/switch\_track\_off"/>

</selector>

* switch\_track\_off.xml

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

    android:shape="rectangle">

    <solid

        android:color="#e4e4e4">

    </solid>

    <corners

        android:radius="30dp">

    </corners>

</shape>

* switch\_track\_on.xml

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

    android:shape="rectangle">

    <solid

        android:color="#00ABEC">

    </solid>

    <corners

        android:radius="32dp">

    </corners>

</shape>

* 设置监听事件

((Switch)findViewById(R.id.activity\_setServe\_switch\_open)).

               setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {

           @Override

           public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {

               if (isChecked) {

                   Toast.makeText(MainActivity.this,"打开",Toast.LENGTH\_LONG).show();

               } else {

                   Toast.makeText(MainActivity.this,"关闭",Toast.LENGTH\_LONG).show();

               }

           }

       });

31、从右边弹出或滑出选择菜单或窗口

* activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

    xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="fill\_parent"

    android:layout\_height="wrap\_content"

    android:gravity="center\_horizontal"

    android:orientation="vertical"

  >

  <TextView

      android:layout\_width="match\_parent"

      android:layout\_height="wrap\_content"

      android:id="@+id/text"

      android:text="按我彈窗口"/>

</RelativeLayout>

* alert\_dialog.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout\_width="fill\_parent"

    android:layout\_height="wrap\_content" >

    <LinearLayout

        android:layout\_width="match\_parent"

        android:layout\_height="match\_parent"

        android:orientation="horizontal" >

        <LinearLayout

            android:layout\_width="0dp"

            android:layout\_height="match\_parent"

            android:layout\_weight="1"

            >

        </LinearLayout>

        <LinearLayout

            android:id="@+id/pop\_layout"

            android:layout\_width="0dp"

            android:layout\_height="match\_parent"

            android:layout\_weight="1"

            android:orientation="vertical"

            android:background="#ffffff" >

        </LinearLayout>

    </LinearLayout>

</RelativeLayout>

* SelectPicPopupWindow.java

package com.lxm.selectpicpopupwindow;

import java.util.ArrayList;

import java.util.List;

import com.lxm.selectpicpopupwindow.been.Items;

import android.app.Activity;

import android.os.Bundle;

import android.view.MotionEvent;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.LinearLayout;

import android.widget.ListView;

import android.widget.Toast;

public class SelectPicPopupWindow extends Activity implements OnClickListener {

      @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.alert\_dialog);

    }

    // 实现onTouchEvent触屏函数但点击屏幕时销毁本Activity

    @Override

    public boolean onTouchEvent(MotionEvent event) {

        finish();

        return true;

    }

    @Override

    public void onClick(View v) {

        // TODO Auto-generated method stub

    }

}

* MainActivity.java

package com.lxm.selectpicpopupwindow;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.ListView;

import android.widget.TextView;

public class MainActivity extends Activity {

    @Override

    public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        TextView tv = (TextView) this.findViewById(R.id.text);

        //把文字控件添加监听，点击弹出自定义窗口

        tv.setOnClickListener(new OnClickListener() {

            public void onClick(View v) {

                startActivity(new Intent(MainActivity.this,SelectPicPopupWindow.class));

            }

        });

    }

}

* AndroidManifest.xml

<activity android:name=".SelectPicPopupWindow" android:theme="@style/MyDialogStyleBottom" />

* styles.xml

<style name="AnimBottom" parent="@android:style/Animation">

       <item name="android:windowEnterAnimation">@anim/push\_bottom\_in</item>

       <item name="android:windowExitAnimation">@anim/push\_buttom\_out</item>

   </style>

   <style name="MyDialogStyleBottom" parent="android:Theme.Holo.DialogWhenLarge">

       <item name="android:windowAnimationStyle">@style/AnimBottom</item>

       <item name="android:windowFrame">@null</item>

<!-- 边框 -->

     <!--   <item name="android:windowIsFloating">true</item> -->

<!-- 是否浮现在activity之上 -->

 <!-- 半透明 -->

       <item name="android:windowIsTranslucent">true</item>

 <!-- 无标题 -->

       <item name="android:windowNoTitle">true</item>

<!-- 背景透明 -->

       <item name="android:windowBackground">@android:color/transparent</item>

<!-- 是否允许对话框的背景变暗 -->

       <item name="android:backgroundDimEnabled">true</item>

   </style>

* push\_bottom\_in.xml（res/anim文件夹下）

<?xml version="1.0" encoding="utf-8"?>

<!-- 右左滑入式 -->

<set xmlns:android="http://schemas.android.com/apk/res/android" >

    <translate

        android:duration="200"

        android:fromXDelta="100%p"

        android:toXDelta="0"

     />

</set>

* push\_buttom\_out.xml

<?xml version="1.0" encoding="utf-8"?>

<!-- 左右滑出式 -->

<set xmlns:android="http://schemas.android.com/apk/res/android" >

    <translate

        android:duration="200"

        android:fromXDelta="0"

        android:toXDelta="100%p"

     />

</set>

32、从底部弹出的Dialog

* activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:fitsSystemWindows="true"

tools:context="com.example.dialogdemo.MainActivity">

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerInParent="true"

android:onClick="show"

android:text="显示弹框"

/>

</RelativeLayout>

* activity\_mapsetting.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:orientation="vertical"

android:background="@drawable/background"

android:layout\_height="match\_parent">

<TextView

android:id="@+id/takePhoto"

android:layout\_width="match\_parent"

android:layout\_height="45dp"

android:layout\_margin="2dp"

android:gravity="center"

android:text="拍照"

android:textColor="#0000ff"

android:textSize="18sp"

android:textStyle="bold" />

<View

android:layout\_width="match\_parent"

android:layout\_height="1px"

android:background="#9e9e9e"

/>

<TextView

android:id="@+id/choosePhoto"

android:layout\_width="match\_parent"

android:layout\_height="45dp"

android:layout\_margin="2dp"

android:gravity="center"

android:text="从相册选择"

android:textColor="#0000ff"

android:textSize="18sp"

android:textStyle="bold" />

</LinearLayout>

* drawable\_radius.xml

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android">

<solid android:color="#ffffff"/>

<corners android:radius="5dp"/>

</shape>

MainActivity.java

public class MainActivity extends AppCompatActivity implements View.OnClickListener{

private View inflate;

private TextView choosePhoto;

private TextView takePhoto;

private Dialog dialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

}

public void show(View view){

dialog = new Dialog(this,R.style.ActionSheetDialogStyle);

//填充对话框的布局

inflate = LayoutInflater.from(this).inflate(R.layout.dialog\_layout, null);

//初始化控件

choosePhoto = (TextView) inflate.findViewById(R.id.choosePhoto);

takePhoto = (TextView) inflate.findViewById(R.id.takePhoto);

choosePhoto.setOnClickListener(this);

takePhoto.setOnClickListener(this);

//将布局设置给Dialog

dialog.setContentView(inflate);

//获取当前Activity所在的窗体

Window dialogWindow = dialog.getWindow();

//设置Dialog从窗体底部弹出

dialogWindow.setGravity( Gravity.BOTTOM);

//获得窗体的属性

WindowManager.LayoutParams lp = dialogWindow.getAttributes();

lp.y = 20;//设置Dialog距离底部的距离

// 将属性设置给窗体

dialogWindow.setAttributes(lp);

dialog.show();//显示对话框

}

@Override

public void onClick(View view) {

switch (view.getId()){

case R.id.takePhoto:

Toast.makeText(this,"点击了拍照",Toast.LENGTH\_SHORT).show();

break;

case R.id.choosePhoto:

Toast.makeText(this,"点击了从相册选择",Toast.LENGTH\_SHORT).show();

break;

}

dialog.dismiss();

}

* Style.xml

<style name="ActionSheetDialogStyle" parent="@android:style/Theme.Dialog">  
 <!-- 背景透明 -->  
 <item name="android:windowBackground">@android:color/transparent</item>  
 <item name="android:windowContentOverlay">@null</item>  
 <!-- 浮于Activity之上 -->  
 <item name="android:windowIsFloating">true</item>  
 <!-- 边框 -->  
 <item name="android:windowFrame">@null</item>  
 <!-- Dialog以外的区域模糊效果 -->  
 <item name="android:backgroundDimEnabled">true</item>  
 <!-- 无标题 -->  
 <item name="android:windowNoTitle">true</item>  
 <!-- 半透明 -->  
 <item name="android:windowIsTranslucent">true</item>  
 <!-- Dialog进入及退出动画 -->  
 <item name="android:windowAnimationStyle">@style/ActionSheetDialogAnimation</item>  
</style>  
<!-- ActionSheet进出动画 -->  
<style name="ActionSheetDialogAnimation" parent="@android:style/Animation.Dialog">  
 <item name="android:windowEnterAnimation">@anim/anim\_push\_bottom\_in</item>  
 <item name="android:windowExitAnimation">@anim/anim\_push\_buttom\_out</item>  
</style>

* anim\_push\_bottom\_in.xml

<?xml version="1.0" encoding="utf-8"?>  
<!-- 右左滑入式 -->  
<set xmlns:android="http://schemas.android.com/apk/res/android" >  
 <translate  
 android:duration="200"  
 android:fromYDelta="100%"  
 android:toYDelta="0"  
 />  
</set>

* anim\_push\_buttom\_out.xml

<?xml version="1.0" encoding="utf-8"?>  
<!-- 左右滑出式 -->  
<set xmlns:android="http://schemas.android.com/apk/res/android" >  
 <translate  
 android:duration="200"  
 android:fromYDelta="0"  
 android:toYDelta="100%"  
 />  
</set>

33、获取屏幕宽度、高度

Display display= getWindow().getWindowManager().getDefaultDisplay();

DisplayMetrics dm=new DisplayMetrics();

display.getMetrics(dm);

int mWidth=dm.widthPixels;

int mHeight=dm.heightPixels;

34、录音并保存至本地



* 添加权限

<uses-permission android:name="android.permission.RECORD\_AUDIO" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

* RecordButton 类，我们的自定义控件，重新复写了onTouchEvent方法

package com.example.recordtest;

import android.annotation.SuppressLint;

import android.app.Dialog;

import android.content.Context;

import android.os.Handler;

import android.os.Message;

import android.util.AttributeSet;

import android.view.Gravity;

import android.view.LayoutInflater;

import android.view.MotionEvent;

import android.view.View;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.TextView;

import android.widget.Toast;

public class RecordButton extends Button {

private static final int MIN\_RECORD\_TIME = 1; // 最短录音时间，单位秒

private static final int RECORD\_OFF = 0; // 不在录音

private static final int RECORD\_ON = 1; // 正在录音

private Dialog mRecordDialog;

private RecordStrategy mAudioRecorder;

private Thread mRecordThread;

private RecordListener listener;

private int recordState = 0; // 录音状态

private float recodeTime = 0.0f; // 录音时长，如果录音时间太短则录音失败

private double voiceValue = 0.0; // 录音的音量值

private boolean isCanceled = false; // 是否取消录音

private float downY;

private TextView dialogTextView;

private ImageView dialogImg;

private Context mContext;

public RecordButton(Context context) {

super(context);

// TODO Auto-generated constructor stub

init(context);

}

public RecordButton(Context context, AttributeSet attrs, int defStyle) {

super(context, attrs, defStyle);

// TODO Auto-generated constructor stub

init(context);

}

public RecordButton(Context context, AttributeSet attrs) {

super(context, attrs);

// TODO Auto-generated constructor stub

init(context);

}

private void init(Context context) {

mContext = context;

this.setText("按住 说话");

}

public void setAudioRecord(RecordStrategy record) {

this.mAudioRecorder = record;

}

public void setRecordListener(RecordListener listener) {

this.listener = listener;

}

// 录音时显示Dialog

private void showVoiceDialog(int flag) {

if (mRecordDialog == null) {

mRecordDialog = new Dialog(mContext, R.style.Dialogstyle);

mRecordDialog.setContentView(R.layout.dialog\_record);

dialogImg = (ImageView) mRecordDialog

.findViewById(R.id.record\_dialog\_img);

dialogTextView = (TextView) mRecordDialog

.findViewById(R.id.record\_dialog\_txt);

}

switch (flag) {

case 1:

dialogImg.setImageResource(R.drawable.record\_cancel);

dialogTextView.setText("松开手指可取消录音");

this.setText("松开手指 取消录音");

break;

default:

dialogImg.setImageResource(R.drawable.record\_animate\_01);

dialogTextView.setText("向上滑动可取消录音");

this.setText("松开手指 完成录音");

break;

}

dialogTextView.setTextSize(14);

mRecordDialog.show();

}

// 录音时间太短时Toast显示

private void showWarnToast(String toastText) {

Toast toast = new Toast(mContext);

View warnView = LayoutInflater.from(mContext).inflate(

R.layout.toast\_warn, null);

toast.setView(warnView);

toast.setGravity(Gravity.CENTER, 0, 0);// 起点位置为中间

toast.show();

}

// 开启录音计时线程

private void callRecordTimeThread() {

mRecordThread = new Thread(recordThread);

mRecordThread.start();

}

// 录音Dialog图片随录音音量大小切换

private void setDialogImage() {

if (voiceValue < 600.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_01);

} else if (voiceValue > 600.0 && voiceValue < 1000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_02);

} else if (voiceValue > 1000.0 && voiceValue < 1200.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_03);

} else if (voiceValue > 1200.0 && voiceValue < 1400.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_04);

} else if (voiceValue > 1400.0 && voiceValue < 1600.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_05);

} else if (voiceValue > 1600.0 && voiceValue < 1800.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_06);

} else if (voiceValue > 1800.0 && voiceValue < 2000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_07);

} else if (voiceValue > 2000.0 && voiceValue < 3000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_08);

} else if (voiceValue > 3000.0 && voiceValue < 4000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_09);

} else if (voiceValue > 4000.0 && voiceValue < 6000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_10);

} else if (voiceValue > 6000.0 && voiceValue < 8000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_11);

} else if (voiceValue > 8000.0 && voiceValue < 10000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_12);

} else if (voiceValue > 10000.0 && voiceValue < 12000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_13);

} else if (voiceValue > 12000.0) {

dialogImg.setImageResource(R.drawable.record\_animate\_14);

}

}

// 录音线程

private Runnable recordThread = new Runnable() {

@Override

public void run() {

recodeTime = 0.0f;

while (recordState == RECORD\_ON) {

{

try {

Thread.sleep(100);

recodeTime += 0.1;

// 获取音量，更新dialog

if (!isCanceled) {

voiceValue = mAudioRecorder.getAmplitude();

recordHandler.sendEmptyMessage(1);

}

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

};

@SuppressLint("HandlerLeak")

private Handler recordHandler = new Handler() {

@Override

public void handleMessage(Message msg) {

setDialogImage();

}

};

@Override

public boolean onTouchEvent(MotionEvent event) {

// TODO Auto-generated method stub

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN: // 按下按钮

if (recordState != RECORD\_ON) {

showVoiceDialog(0);

downY = event.getY();

if (mAudioRecorder != null) {

mAudioRecorder.ready();

recordState = RECORD\_ON;

mAudioRecorder.start();

callRecordTimeThread();

}

}

break;

case MotionEvent.ACTION\_MOVE: // 滑动手指

float moveY = event.getY();

if (downY - moveY > 50) {

isCanceled = true;

showVoiceDialog(1);

}

if (downY - moveY < 20) {

isCanceled = false;

showVoiceDialog(0);

}

break;

case MotionEvent.ACTION\_UP: // 松开手指

if (recordState == RECORD\_ON) {

recordState = RECORD\_OFF;

if (mRecordDialog.isShowing()) {

mRecordDialog.dismiss();

}

mAudioRecorder.stop();

mRecordThread.interrupt();

voiceValue = 0.0;

if (isCanceled) {

mAudioRecorder.deleteOldFile();

} else {

if (recodeTime < MIN\_RECORD\_TIME) {

showWarnToast("时间太短 录音失败");

mAudioRecorder.deleteOldFile();

} else {

if (listener != null) {

listener.recordEnd(mAudioRecorder.getFilePath());

}

}

}

isCanceled = false;

this.setText("按住 说话");

}

break;

}

return true;

}

public interface RecordListener {

public void recordEnd(String filePath);

}

}

* Dialog布局

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

android:gravity="center"

android:background="@drawable/record\_bg"

android:padding="20dp" >

<ImageView

android:id="@+id/record\_dialog\_img"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" />

<TextView

android:id="@+id/record\_dialog\_txt"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textColor="@android:color/white"

android:layout\_marginTop="5dp" />

</LinearLayout>

* 录音时间太短的Toast布局

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:background="@drawable/record\_bg"

android:padding="20dp"

android:gravity="center"

android:orientation="vertical" >

<ImageView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:src="@drawable/voice\_to\_short" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textColor="@android:color/white"

android:textSize="15sp"

android:text="时间太短 录音失败" />

</LinearLayout>

* 自定义的Dialogstyle，对话框样式

<style name="Dialogstyle">

<item name="android:windowBackground">@android:color/transparent</item>

<item name="android:windowFrame">@null</item>

<item name="android:windowNoTitle">true</item>

<item name="android:windowIsFloating">true</item>

<item name="android:windowIsTranslucent">true</item>

<item name="android:windowAnimationStyle">@android:style/Animation.Dialog</item>

<!-- 显示对话框时当前的屏幕是否变暗 -->

<item name="android:backgroundDimEnabled">false</item>

</style>

* RecordStrategy 录音策略接口

package com.example.recordtest;

/\*\*

\* RecordStrategy 录音策略接口

\* @author acer

\*/

public interface RecordStrategy {

/\*\*

\* 在这里进行录音准备工作，重置录音文件名等

\*/

public void ready();

/\*\*

\* 开始录音

\*/

public void start();

/\*\*

\* 录音结束

\*/

public void stop();

/\*\*

\* 录音失败时删除原来的旧文件

\*/

public void deleteOldFile();

/\*\*

\* 获取录音音量的大小

\* @return

\*/

public double getAmplitude();

/\*\*

\* 返回录音文件完整路径

\* @return

\*/

public String getFilePath();

}

* 个人写的一个录音实践策略

package com.example.recordtest;

import java.io.File;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.util.Date;

import android.media.MediaRecorder;

import android.os.Environment;

public class AudioRecorder implements RecordStrategy {

private MediaRecorder recorder;

private String fileName;

private String fileFolder = Environment.getExternalStorageDirectory()

.getPath() + "/TestRecord";

private boolean isRecording = false;

@Override

public void ready() {

// TODO Auto-generated method stub

File file = new File(fileFolder);

if (!file.exists()) {

file.mkdir();

}

fileName = getCurrentDate();

recorder = new MediaRecorder();

recorder.setOutputFile(fileFolder + "/" + fileName + ".amr");

recorder.setAudioSource(MediaRecorder.AudioSource.MIC);// 设置MediaRecorder的音频源为麦克风

recorder.setOutputFormat(MediaRecorder.OutputFormat.RAW\_AMR);// 设置MediaRecorder录制的音频格式

recorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR\_NB);// 设置MediaRecorder录制音频的编码为amr

}

// 以当前时间作为文件名

private String getCurrentDate() {

SimpleDateFormat formatter = new SimpleDateFormat("yyyy\_MM\_dd\_HHmmss");

Date curDate = new Date(System.currentTimeMillis());// 获取当前时间

String str = formatter.format(curDate);

return str;

}

@Override

public void start() {

// TODO Auto-generated method stub

if (!isRecording) {

try {

recorder.prepare();

recorder.start();

} catch (IllegalStateException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

isRecording = true;

}

}

@Override

public void stop() {

// TODO Auto-generated method stub

if (isRecording) {

recorder.stop();

recorder.release();

isRecording = false;

}

}

@Override

public void deleteOldFile() {

// TODO Auto-generated method stub

File file = new File(fileFolder + "/" + fileName + ".amr");

file.deleteOnExit();

}

@Override

public double getAmplitude() {

// TODO Auto-generated method stub

if (!isRecording) {

return 0;

}

return recorder.getMaxAmplitude();

}

@Override

public String getFilePath() {

// TODO Auto-generated method stub

return fileFolder + "/" + fileName + ".amr";

}

}

* MainActivity

package com.example.recordtest;

import android.os.Bundle;

import android.app.Activity;

import android.view.Menu;

public class MainActivity extends Activity {

RecordButton button;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

button = (RecordButton) findViewById(R.id.btn\_record);

button.setAudioRecord(new AudioRecorder());

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

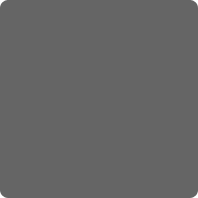
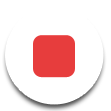
getMenuInflater().inflate(R.menu.main, menu);

return true;

}

}

* 所需资源文件



35、获取音频时间长度

//根据路径获取音频时间长度  
private String getRecordtime(String recordPath)  
{ //使用此方法可以直接在后台获取音频文件的播放时间，而不会真的播放音频  
 MediaPlayer player = new MediaPlayer(); //首先你先定义一个mediaplayer  
 try {  
 player.setDataSource(recordPath); //String是指音频文件的路径  
 player.prepare(); //这个是mediaplayer的播放准备 缓冲  
 } catch (Exception e)  
 { e.printStackTrace(); }  
 player.setOnPreparedListener(new MediaPlayer.OnPreparedListener()  
 {//监听准备  
 @Override  
 public void onPrepared(MediaPlayer player)  
 {  
 int size = player.getDuration();  
 String timelong = size / 1000 + "''";  
 }  
 });  
 double size =player.getDuration();//得到音频的时间  
 String timelong1 = (int) Math.ceil((size / 1000)) + "''";//转换为秒 单位为''  
 player.stop();//暂停播放  
 player.release();//释放资源  
 return timelong1; //返回音频时间  
}

36、通过Gradle动态配置服务器地址

* 在build.gradle中的android节点添加

*//正式环境*def API\_RELEASE\_HOST = "\"http://192.168.16.116:8080/SafeMonitorServer/\""

*//开发环境*def API\_DEV\_HOST = "\"http://192.168.16.116:8080/SafeMonitorServer/\""

* 在buildTypes 节点下的release 节点添加

buildConfigField "String", "API\_HOST", "${API\_RELEASE\_HOST}"

* 在defaultConfig节点添加

buildConfigField("String", "API\_HOST", "${API\_DEV\_HOST}")

37、基于ViewPager实现欢迎界面滑动效果

* 在acitivity\_welcome.xml中添加

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
 <android.support.v4.view.ViewPager  
 android:id="@+id/vp\_welcome"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" />  
</LinearLayout>

* 在WelcomeActivity.java中添加

package com.edu.ouc.safemonitor.activity;  
import android.app.Activity;  
import android.os.Bundle;  
import android.os.Parcelable;  
import android.support.annotation.Nullable;  
import android.support.v4.view.PagerAdapter;  
import android.support.v4.view.ViewPager;  
import android.support.v4.view.ViewPropertyAnimatorCompat;  
import android.view.View;  
import android.widget.ImageView;  
import android.widget.LinearLayout;  
import com.edu.ouc.safemonitor.R;  
import java.util.ArrayList;  
import java.util.List;  
/\*\*  
 \* Created by JHC on 2018/3/17.  
 \* 欢迎界面  
 \*/  
public class WelcomeAvtivity extends Activity implements View.OnClickListener, ViewPager.OnPageChangeListener {  
 private ViewPager viewPager\_welcome;  
 private ViewPagerAdapter viewPagerAdapter;  
 private List<View> views;  
 // 引导图片资源  
 private static final int[] pics = { R.mipmap.welocome\_one, R.mipmap.ic\_add,  
 R.mipmap.ic\_alarm\_false, R.mipmap.ic\_code };  
 @Override  
 protected void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_welcome);  
 viewPager\_welcome = (ViewPager) findViewById(R.id.vp\_welcome);  
 views = new ArrayList<View>();  
 LinearLayout.LayoutParams mParams = new LinearLayout.LayoutParams(  
 LinearLayout.LayoutParams.WRAP\_CONTENT,  
 LinearLayout.LayoutParams.WRAP\_CONTENT);  
 // 初始化引导图片列表  
 for (int i = 0; i < pics.length; i++) {  
 ImageView iv = new ImageView(this);  
 iv.setLayoutParams(mParams);  
 iv.setImageResource(pics[i]);  
 views.add(iv);  
 }  
 // 初始化Adapter  
 viewPagerAdapter = new ViewPagerAdapter(views);  
 viewPager\_welcome.setAdapter(viewPagerAdapter);  
 // 绑定回调  
 viewPager\_welcome.setOnPageChangeListener(this);  
 }  
 @Override  
 public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) {  
 }  
 @Override  
 public void onPageSelected(int position) {  
 }  
 @Override  
 public void onPageScrollStateChanged(int state) {  
 }  
 @Override  
 public void onClick(View v) {  
 }  
}  
 class ViewPagerAdapter extends PagerAdapter {  
 // 界面列表  
 private List<View> views;  
 public ViewPagerAdapter(List<View> views) {  
 this.views = views;  
 }  
 // 销毁arg1位置的界面  
 @Override  
 public void destroyItem(View arg0, int arg1, Object arg2) {  
 ((ViewPager) arg0).removeView(views.get(arg1));  
 }  
 @Override  
 public void finishUpdate(View arg0) {  
 // TODO Auto-generated method stub  
 }  
 // 获得当前界面数  
 @Override  
 public int getCount() {  
 if (views != null) {  
 return views.size();  
 }  
 return 0;  
 }  
 // 初始化arg1位置的界面  
 @Override  
 public Object instantiateItem(View arg0, int arg1) {  
 ((ViewPager) arg0).addView(views.get(arg1), 0);  
 return views.get(arg1);  
 }  
 // 判断是否由对象生成界面  
 @Override  
 public boolean isViewFromObject(View arg0, Object arg1) {  
 // TODO Auto-generated method stub  
 return (arg0 == arg1);  
 }  
 @Override  
 public void restoreState(Parcelable arg0, ClassLoader arg1) {  
 // TODO Auto-generated method stub  
 }  
 @Override  
 public Parcelable saveState() {  
 // TODO Auto-generated method stub  
 return null;  
 }  
 @Override  
 public void startUpdate(View arg0) {  
 // TODO Auto-generated method stub  
 }  
 }

38、判断应用是否第一次运行

SharedPreferences sharedPreferences = this.getSharedPreferences("share", MODE\_PRIVATE);

boolean isFirstRun = sharedPreferences.getBoolean("isFirstRun", true);

Editor editor = sharedPreferences.edit();

if (isFirstRun)

{

Log.d("debug", "第一次运行");

editor.putBoolean("isFirstRun", false);

editor.commit();

} else

{

Log.d("debug", "不是第一次运行");

}

39、**Alarm**实现后台定时任务执行

* 自定义Service

package com.edu.ouc.safemonitor.service;  
import android.app.AlarmManager;  
import android.app.PendingIntent;  
import android.app.Service;  
import android.content.Intent;  
import android.os.IBinder;  
import android.os.SystemClock;  
import android.support.annotation.Nullable;  
import android.util.Log;  
import com.edu.ouc.safemonitor.receiver.AutoUpdateReceiver;  
import java.util.Date;  
/\*\*  
 \* Created by may on 2018/3/23.  
 \*/  
public class AutoUpdateService extends Service {  
 @Nullable  
 @Override  
 public IBinder onBind(Intent intent) {  
 return null;  
 }  
 // 每次服务启动的时候调用  
 @Override  
 public int onStartCommand(Intent intent, int flags, int startId) {  
 new Thread(new Runnable() {  
 @Override  
 public void run() {  
 Log.d("AutoUpdateService", "executed at " + new Date().  
 toString());  
 }  
 }).start();  
 AlarmManager manager = (AlarmManager) getSystemService(ALARM\_SERVICE);  
 int anHour =10 \* 1000;// 定时任务  
 long triggerAtTime = SystemClock.elapsedRealtime() + anHour;  
 Intent intent2 = new Intent(this, AutoUpdateReceiver.class);  
 PendingIntent pi = PendingIntent.getBroadcast(this, 0, intent2, 0);  
 manager.set(AlarmManager.ELAPSED\_REALTIME\_WAKEUP, triggerAtTime, pi);  
 return super.onStartCommand(intent, flags, startId);  
 }  
}

注意：

ong triggerAtTime = SystemClock.elapsedRealtime() + 10 \* 1000;

manager.set(AlarmManager.ELAPSED\_REALTIME\_WAKEUP,triggerAtTime,pendingIntent);

上面的两行代码你不一定能看得明白，因为set()方法中需要传入的三个参数稍微有点复杂，下面我们就来仔细地分析一下。

第一个参数是一个整型参数，用于指定AlarmManager 的工作类型，有四种值可选，分别是ELAPSED\_REALTIME、ELAPSED\_REALTIME\_WAKEUP、RTC 和RTC\_WAKEUP。其中ELAPSED\_REALTIME 表示让定时任务的触发时间从系统开机开始算起，但不会唤醒CPU。ELAPSED\_REALTIME\_WAKEUP 同样表示让定时任务的触发时间从系统开机开始算起，但会唤醒CPU。RTC 表示让定时任务的触发时间从1970 年1月1 日0 点开始算起，但不会唤醒CPU。RTC\_WAKEUP 同样表示让定时任务的触发时间从1970 年1 月1 日0 点开始算起，但会唤醒CPU。使用SystemClock.elapsedRealtime()方法可以获取到系统开机至今所经历时间的毫秒数，使用System.currentTimeMillis()方法可以获取到1970 年1 月1 日0 点至今所经历时间的毫秒数。

然后看一下第二个参数，这个参数就好理解多了，就是定时任务触发的时间，以毫秒为单位。如果第一个参数使用的是ELAPSED\_REALTIME 或ELAPSED\_REALTIME\_WAKEUP，则这里传入开机至今的时间再加上延迟执行的时间。如果第一个参数使用的是RTC 或RTC\_WAKEUP，则这里传入1970 年1 月1 日0 点至今的时间再加上延迟执行的时间。

第三个参数是一个PendingIntent，对于它你应该已经不会陌生了吧。这里我们一般会调用getBroadcast()方法来获取一个能够执行广播的PendingIntent。这样当定时任务被触发的时候，广播接收器的onReceive()方法就可以得到执行。了解了set()方法的每个参数之后，你应该能想到，设定一个任务在10 秒钟后执行还可以写成：  
long triggerAtTime = System.currentTimeMillis() + 10 \* 1000;  
manager.set(AlarmManager.RTC\_WAKEUP, triggerAtTime, pendingIntent);

* 自定义广播接收器

package com.edu.ouc.safemonitor.receiver;  
import android.content.BroadcastReceiver;  
import android.content.Context;  
import android.content.Intent;  
import com.edu.ouc.safemonitor.service.AutoUpdateService;  
/\*\*  
 \* Created by may on 2018/3/23.  
 \* 定时任务广播接收器  
 \*/  
public class AutoUpdateReceiver extends BroadcastReceiver {  
 @Override  
 public void onReceive(Context context, Intent intent) {  
 Intent i = new Intent(context, AutoUpdateService.class);  
 context.startService(i);  
 }  
}

* 在应用启动时启动定时器

//启动service定时任务  
Intent intent =new Intent(this,AutoUpdateService.class);  
startService(intent);

* 服务和广播接收器在AndroidManifest.xml 注册

<service android:name=".service.AutoUpdateService" ></service>  
<receiver android:name=".receiver.AutoUpdateReceiver" ></receiver>

40、播放**外部存储上的音频资源文件(sdcard)**

MediaPlayer mediaPlayer = new MediaPlayer();

mediaPlayer.setDataSource("/mnt/sdcard/a3.mp3");

mediaPlayer.prepare();

mediaPlayer.start();

41、service弹框Dialog

安装常规写好 AlertDialog 功能块后，在alertObj .show()语句前加入：

alertObj.getWindow().setType(WindowManager.LayoutParams.TYPE\_SYSTEM\_ALERT);

* 添加弹框方法

private void showHostOnlineAlert(){

final AlertDialog dialog =new AlertDialog.Builder(BackgroudService.this).create();

dialog.setCanceledOnTouchOutside(false);

dialog.requestWindowFeature(Window.FEATURE\_NO\_TITLE);

dialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.TRANSPARENT));//set background was transparent

dialog.getWindow().setType(WindowManager.LayoutParams.TYPE\_SYSTEM\_ALERT);//需要添加的语句

dialog.show();

}

* 在AndroidManifest.xml中加入权限

<uses-permission android:name="android.permission.SYSTEM\_ALERT\_WINDOW"></uses-permission>

42、修改Dialog标题颜色、字体颜色

AlertDialog dialog = new AlertDialog.Builder(this)  
 .setTitle("AlerDialog")  
 .setMessage("这是一个AlertDialog")  
 .setPositiveButton("确定",null)  
 .create();  
//修改dialog的背景颜色  
dialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.WHITE));//set background was transparent  
dialog.show();

try {  
 Field mAlert = AlertDialog.class.getDeclaredField("mAlert");  
 mAlert.setAccessible(true);  
 Object mAlertController = mAlert.get(dialog);  
 //修改dialog的标题颜色  
 Field mtitle = mAlertController.getClass().getDeclaredField("mTitleView");  
 mtitle.setAccessible(true);  
 TextView mtitleView = (TextView) mtitle.get(mAlertController);  
 mtitleView.setTextColor(Color.BLACK);  
 //修改dialog的内容颜色  
 Field mMessage = mAlertController.getClass().getDeclaredField("mMessageView");  
 mMessage.setAccessible(true);  
 TextView mMessageView = (TextView) mMessage.get(mAlertController);  
 mMessageView.setTextColor(Color.BLACK);  
} catch (IllegalAccessException e) {  
 e.printStackTrace();  
} catch (NoSuchFieldException e) {  
 e.printStackTrace();  
}

43、手机振动（震动）功能实现

* 添加权限

<!--用于震动权限-->  
<uses-permission android:name="android.permission.VIBRATE" />

* 定义震动变量已调用

private Vibrator mvibrator; //创建震动服务对象

//获取手机震动服务  
mvibrator=(Vibrator)getApplication().getSystemService(Service.VIBRATOR\_SERVICE);

//设置震动周期，数组表示时间：等待+执行，单位是毫秒，下面操作代表:等待100，执行100，等待100，执行1000，  
//后面的数字如果为-1代表不重复，只执行一次，其他代表会重复，0代表从数组的第0个位置开始  
mvibrator.vibrate(new long[]{100,100,100,1000},-1);

44、Fragment实现下拉刷新（SwipeRefreshLayout）

* 添加布局

<?xml version="1.0" encoding="utf-8"?>  
<android.support.v4.widget.SwipeRefreshLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/refresh\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".ListRefreshFragment">  
 <ListView  
 android:id="@+id/lv"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
 </ListView>  
</android.support.v4.widget.SwipeRefreshLayout>

* 在Fragment中添加下拉刷新监听函数

public class AlarmFragment extends Fragment implements SwipeRefreshLayout.OnRefreshListener {  
 private List<AlarmInfoModel> datas; //报警集合对象  
 private ListView listView\_alarm; //报警list控件  
 private AlarmInfoListViewAdapter alarmInfoListViewAdapter; //自定义适配器对象  
 private SwipeRefreshLayout swipeRefreshLayout; //用于下拉刷新  
 public AlarmFragment() {  
 // Required empty public constructor  
 }  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 // Inflate the layout for this fragment  
 View view=inflater.inflate(R.layout.fragment\_alarm, container, false);  
 return view;  
 }  
 @Override  
 public void onActivityCreated(@Nullable Bundle savedInstanceState) {  
 listView\_alarm = (ListView)getActivity().findViewById(R.id.lv\_alarm);  
 swipeRefreshLayout = (SwipeRefreshLayout) getActivity().findViewById(R.id.layout\_alarm\_refresh);  
 swipeRefreshLayout.setOnRefreshListener(this);  
 swipeRefreshLayout.setColorSchemeResources(  
 android.R.color.holo\_blue\_bright,  
 android.R.color.holo\_green\_light,  
 android.R.color.holo\_orange\_light,  
 android.R.color.holo\_red\_light);

getdata();  
 super.onActivityCreated(savedInstanceState);  
 }  
 //下拉刷新执行函数  
 @Override  
 public void onRefresh() {  
 //下拉刷新再次调用获取数据函数  
 getdata();  
 swipeRefreshLayout.setRefreshing(true);  
 new Handler().postDelayed(new Runnable() {  
 public void run() {  
 swipeRefreshLayout.setRefreshing(false);  
 }  
 }, 3000);  
 }  
}