# 手机安全卫士项目

## 01\_程序功能简介：

**手机防盗:**

根据sim的变更,判断手机是否被盗,根据协议发送手机中sim卡的信息,和当前手机的位置(经纬度),给安全号码.

快速进入手机防盗

可以远程发送指令,锁屏,工厂重置手机

md5加密 防止卸载

**通讯卫士:**

来电归属地显示,来电黑名单/短信黑名单管理. 电话短信备份.

**软件管理:**

系统软件和本地软件,显示软件的详细信息,启动,删除应用程序. 获取服务器上软件的 评分信息. 程序锁.

**任务管理:**

管理系统任务 显示系统内存信息,杀死,批量杀死进程,widget

**上网管理:**

显示出每个程序访问的流量. 漫游管理.

**手机杀毒**:根据包名和程序的证书识别病毒,查杀

**系统优化:** 清理手机缓存

优化电池管理

**高级工具:**

自动ip拨号

手机号码归属地查询

更改归属地位置

常用号码

**设置中心:**

开机是否自动启动

服务启动是否在状态栏显示

设置归属地显示在屏幕的位置

**关于软件**: html webview

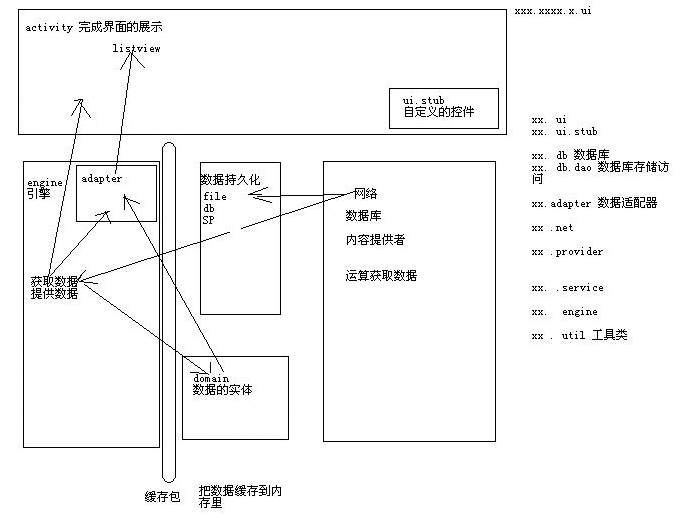
bug管理,mantis使用,log管理

打包,混淆加密.发布, 三方广告条的添加

## 02\_splash界面ui：

### 通用包结构简介：

**一般程序的包结构如下：**



**利用程序的功能（业务）逻辑进行代码控件的划分：**



### splash界面功能简介：

完成一个产品logo显示.

1. 后台完成数据库初始化的操作

2. 联网访问服务器,获取服务器最新信息

3. 不同的时间显示出来不同logo 判断当前系统时间, 素材一般从服务器上下载下来.

4. 判断时间 根据时间显示不同的logo

### 相关知识点：

**布局设置背景图片android:background**

控件命名规则:Tv\_类名\_name

**控件在布局中，控制自身位置的方式:**

android:layout\_marginTop等……是一种不怎么靠谱的方式，是否应该在程序中动态地根据当前屏幕的size动态修改。是否可以再XML文件中获取当前屏幕的size？

**取消标题栏的方法：(在setcontentview之前)**

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

android.view.Window中定义的常量

**完成Activity的全屏显示，（setcontentView之后）**

getWindow().setFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN,WindowManager.LayoutParams.FLAG\_FULLSCREEN);

Ps: 也可以通过主题设置窗体全屏显示

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

**获取当前应用程序的版本号**

在AndroidManifest文件中，定义有

android:versionCode="1"

android:versionName="1.0"

如何在程序中得到这个信息呢？

PackageInfo类：

Overall information about the contents of a package. This corresponds to all of the information collected from AndroidManifest.xml.

需要通过PackageManager得到PackageInfo的实例：

PackageManager manager = getPackageManager();

PackageInfo info = manager.getPackageInfo(getPackageName(), 0);

然后，info.versionName 得到需要的版本信息

透明动画效果

方法：

AlphaAnimation aa = new AlphaAnimation(0.0f, 1.0f);

aa.setDuration(2000);

ll\_splash\_main.startAnimation(aa);

所有的View子类都能实现，包括各种布局ViewGroup

**[Open Declaration](eclipse-open:%E2%98%82=mobilesafe/G:\/adt-bundle-windows-x86_64-20140321\/adt-bundle-windows-x86_64-20140321\/sdk\/platforms\/android-19\/android.jar%3candroid.view(View.class%E2%98%83View~startAnimation~Landroid.view.animation.Animation;)void** [**android**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid)**.**[**view**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.view)**.**[**View**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.view(View.class%E2%98%83View)**.startAnimation(**[**Animation**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.view(View.class%E2%98%83View~startAnimation~Landroid.view.animation.Animation;%E2%98%82android.view.animation.Animation) **animation)**

## 03\_获取更新的服务器配置,显示更新对话框：

**项目开发流程：**

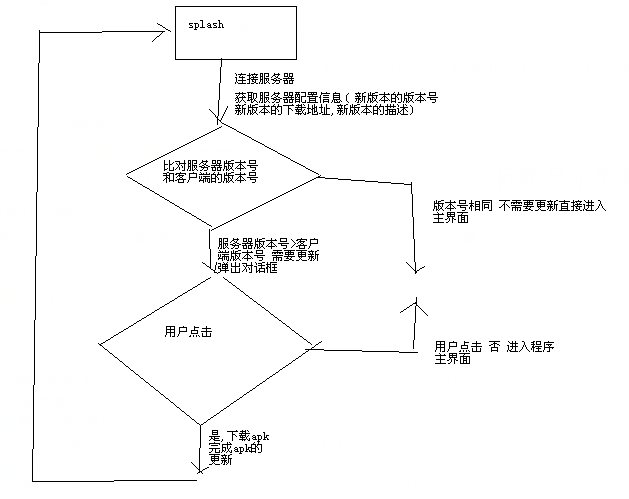
1.明确项目的需求:

2.美工人员根据需求构图.

3.开发

项目经理 控制项目开发进度.

### 更新apk的流程：



### 相关知识点

**服务器端配置update信息**

1. 配置好Tomcat的环境变量，CATALINA\_HOME，C:\Program Files\Apache Software Foundation\Tomcat 7.0
2. C:\Program Files\Apache Software Foundation\Tomcat 7.0\webapps\ROOT文件夹下，新建update.xml文件，定义各种更新信息。注意编码为utf-8
3. <http://localhost:8088/update.xml> 测试一下，就知道OK否。

**解析xml文件的方法：**

XmlPullParser parser = Xml.newPullParser();

UpdataInfo info = new UpdataInfo();

**parser.setInput(is, "utf-8");**

*//定位到XML文件的开头*

int type = parser.geteventtype();

while (type != xmlpullparser.end\_document) {

switch (type) {

case XmlPullParser.START\_TAG:

if("version".equals(parser.getName())){

String version = parser.nextText();

info.setVersion(version);

}else if("description".equals(parser.getName())){

String description = parser.nextText();

info.setDescription(description);

}else if("apkurl".equals(parser.getName())){

String apkurl = parser.nextText();

info.setApkurl(apkurl);

}

break;

}

// next

type = parser.next();

}

return info;

}

**Android中Junit测试的方法：**

AndroidManifest.xml中添加

1. <application下：

<uses-library android:name="android.test.runner"

1. <manifest下：

<instrumentation

android:name="android.test.InstrumentationTestRunner"

android:targetPackage="cn.itcast.mobilesafe" />

1. 创建测试子包

创建测试类继承AndroidTestCase类

1. 编程，断言asserEquas……

**网络操作注意事项：**

Android中不允许在主线程中进行网络操作，所以所有涉及网络的操作都必须new Thread执行,或者使用异步任务；同时，android中所有对界面的更新操作都必须在主线程中进行，所以涉及子线程中操作主线程，必须定义一个Handler，并重写她的handlermessenage

**AlertDialog的使用问题：**

AlertDialog.Builder builder = new Builder(Aty\_Entry.this);

设置完所有的参数，监听器之后，最后才使用

builder.create().show();

不要在两个按钮的事件处理中执行耗时任务，这样按下按钮之后就会死机……

**注意数值的除法中……如果需要精度必须有一个运算方是Float……**

**其他异常：**

09-25 05:13:25.866: E/AndroidRuntime(8724): java.lang.RuntimeException: Can't create handler inside thread that has not called Looper.prepare()

在new Thread中使用Toast方法会出现这样的情况。因为子线程不能直接操作主界面，要通过Handler。

通过使用Looper.prepare()方法也可以在子线程中更新主UI。

## 04\_自动更新apk：

### 安装应用签名问题：

在做程序自动安装更新的时候 ,必须保证程序的签名和包名是相同.

C:\Documents and Settings\zehua\.android\ **debug.keystore**

**debug的签名,开发时候的签名**

如果签名不一致：就不能更新了……

Re-installation failed due to different application signatures.

Please execute 'adb uninstall cn.itcast.mobilesafe' in a shell.

Eclipse安装Android应用会以debug keystore的方式安装

### 需求：

进入应用程序后立刻读取服务器数据（03讲），返回相应的数据，判断是否存在新版本。

需要创建一个执行下载任务的Java类，新开一个线程执行下载任务，进度条。

服务器端，需要在root目录下，放置新版本的apk文件

**进度条的疑问：**

为什么可以在子线程中直接去更新ProgressBar？应该是只有在主线程或者handler中才能操作界面组件的吗？

点击开来ProgressBar的源代码，发现其实还是通过一个Handler实现的。

所以在子线程中也可以很方便的操作进度条。

想想看还有什么其他控件是这样的呢？

**下载完成后安装：**

private void install(File file) {

Intent intent = new Intent();

intent.setAction(Intent.ACTION\_VIEW);

intent.setDataAndType(Uri.fromFile(file),

"application/vnd.android.package-archive");

finish();

startActivity(intent);

}

关键方法：

[String](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content(Intent.class%E2%98%83Intent%5EACTION_VIEW%E2%98%82java.lang.String)[android](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid)**.**[content](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content)**.**[Intent](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content(Intent.class%E2%98%83Intent)**.ACTION\_VIEW = "android.intent.action.VIEW"**

Activity Action: Display the data to the user. This is the most common action performed on data -- it is the generic action you can use on a piece of data to get the most reasonable thing to occur. For example, when used on a contacts entry it will view the entry; when used on a mailto: URI it will bring up a compose window filled with the information supplied by the URI; when used with a tel: URI it will invoke the dialer.

用于显示用户的数据。

比较通用，会根据用户的数据类型打开相应的Activity。

比如 tel:13400010001打开拨号程序，http://www.g.cn则会打开浏览器等。

[**Intent**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content(Intent.class%E2%98%83Intent~setDataAndType~Landroid.net.Uri;~Ljava.lang.String;%E2%98%82android.content.Intent)[**android**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid)**.**[**content**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content)**.**[**Intent**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content(Intent.class%E2%98%83Intent)**.setDataAndType(**[**Uri**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content(Intent.class%E2%98%83Intent~setDataAndType~Landroid.net.Uri;~Ljava.lang.String;%E2%98%82android.net.Uri) **data,** [**String**](eclipse-javadoc:%E2%98%82=mobilesafe/G:%5C/adt-bundle-windows-x86_64-20140321%5C/adt-bundle-windows-x86_64-20140321%5C/sdk%5C/platforms%5C/android-19%5C/android.jar%3Candroid.content(Intent.class%E2%98%83Intent~setDataAndType~Landroid.net.Uri;~Ljava.lang.String;%E2%98%82java.lang.String) **type)**

(Usually optional) Set the data for the intent along with an explicit MIME data type. This method should very rarely be used -- it allows you to override the MIME type that would ordinarily be inferred from the data with your own type given here.

Type是data的mime类型。

打开tomcat下的conf文件夹，打开web.xml，可以找到

<mime-mapping>

<extension>apk</extension>

<mime-type>application/vnd.android.package-archive</mime-type>

</mime-mapping>

## 05\_上下文的区别

两种上下文

03-30 04:01:48.632: ERROR/AndroidRuntime(4841):

android.view.WindowManager$BadTokenException: Unable to add window -- token null is not for an application

applicationcontext 生命周期 进程存在 上下文就存在

activity.this activity 在任务栈 上下文存在

对话框就只能使用Avtivity的上下文了。

## 06\_程序主界面

GridView的使用方法，添加间隔等美化方式。

继承baseadapter的方式

## 07\_手机防盗自定义对话框

### 相关知识点：

**在一个service或者广播中启动一个Activity**

必须要加上这一行代码：

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

//指定要激活的activity在自己的任务栈里面运行

03-30 07:28:38.542: ERROR/AndroidRuntime(7028): java.lang.RuntimeException:

Unable to start receiver cn.itcast.mobilesafe.receiver.CallPhoneReceiver:

android.util.AndroidRuntimeException: Calling startActivity()

from outside of an Activity context requires the FLAG\_ACTIVITY\_NEW\_TASK flag.

Is this really what you want?

## 08\_手机防盗界面对话框2：

### 加载视图

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

//dialog.setContentView(R.layout.first\_entry\_dialog);

View view = View.inflate(this, R.layout.first\_entry\_dialog, null);

et\_pwd = (EditText) view.findViewById(R.id.et\_first\_entry\_pwd);

et\_pwd\_confirm = (EditText) view.findViewById(R.id.et\_first\_entry\_pwd\_confirm);

选择inflate方法的原因是，这个方法返回了view，可以方便地获取到layout中定义的各个控件。

### 加载一个自定义View的对话框：

先利用layoutinflater获得相应的xml布局文件view

然后新建对话框，

setView(view)方法设置当前对话框的自定义布局为view

create()并show();

### 对存储的参数的MD5加密

MD5加密：不可逆的加密方式。

SharedPreference存储的参数，很容易被导出被别人观察到。记得最后要commit()最后的修改才能生效，这个和数据库一样啊。

使用Md5加密

**MessageDigest** digest = MessageDigest.getInstance("MD5");

byte[] bytes = digest.**digest**(pwd.getBytes());

java.security.MessageDigest

Uses a one-way hash function to turn an arbitrary number of bytes into a fixed-length byte sequence. The original arbitrary-length sequence is the message, and the fixed-length byte sequence is the digest or message digest.

弱密码的MD5可以反向查询……

## 09\_一些布局和显示的细节

反编译安卓，可以看到许多资源文件，xml文件，如何反编译？

Xml ->定义一个背景颜色 shape (参考api文件)

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

android:shape="rectangle" >

<corners android:radius="2dip" >

</corners>

<gradient android:startColor="#ff2c2d2d"

android:centerColor="#ff262626"

android:endColor="#ff515151"

/>

</shape>

### <selector />

选择器的使用

某种状态下可以动态调整外观

如何创建？

在dreawable文件夹中创建。

如何使用？

在控件的background属性中设置

LayoutmarginLeft=100，控件距离左边100

Paddingleft=100，控件里面的内容距离左边100

## 10\_手机防盗界面01

**需求:**

判断用户是否已经进行过设置向导,

如果用户已经设置过了程序,不在提示用户进入设置向导

如果用户没有设置过手机防盗, 提示用户进入设置向导界面

**使用@android下的图片资源好处:**

1.减少程序apk的体积

2.提高程序获取图片的速度.

**Style的使用**

如何创建？在Values文件夹下

<style name="view\_divide\_line\_style">

<item name="android:layout\_width">fill\_parent</item>

<item name="android:layout\_height">1dip</item>

<item name="android:layout\_marginTop">5dip</item>

<item name="android:background">@drawable/devide\_line</item>

</style>

如何使用？

需要使用的控件指定style属性

作用？

就是抽象出共同的属性，方便修改

## 11\_手机防盗设置界面02&03

**设置activity切换时候的动画效果**

overridePendingTransition(R.anim.tran\_enter, R.anim.tran\_exit); 方法

如何定义动画效果？

位移动画：

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

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

android:fromXDelta=*"-100%p"*

android:toXDelta=*"0"*

android:duration=*"200"*>

</translate>

**注意：**

当在应用中需要重复使用某个固定字符串，务必使用静态常量命名，不然后期有你受的啊……血的教训啊朋友……

同样的，在XML中的style也可以抽象出来

## 13\_手机防盗设置完成

需求：

完成手机防盗的逻辑，手机开机后，侦测开机完成事件，然后判断当前Sim卡是否和存储的Sim卡序列号一致，若不一致，向目标电话号码发送短信，报告当前位置，当前Sim卡。（是否需要把这条短信影隐藏呢？）

**中止广播**

有序广播 ->1.一般的有序广播 abortbroadcast() (-1000~1000)

2.指定了接受者的有序广播setResult();

相关知识点：

利用BroadCast检测开机事件，继承BroadCast

在xml中进行准确配置：

**发送短信**

android.telephony.SmsManager smsManager= android.telephony.SmsManager.getDefault();

这个包下的SmsManager同时支持CDMA和GSM

但是原来的包就不行了，注意使用。

**短信内容的获取**

Object[] pdus = (Object[]) intent.getExtras().get("pdus");

然后就可以遍历了

## 14\_通过gps获取到当前位置

如何创建一个只能创建一个实例的类？

**这就叫单例模式：**

**实现一个单例模式：**

public class A {

private static A a;

private A() {}

public static A getInstance() {

if(a == null)

a = new A();

return a;

}

}

**GPS的服务类就是要这样创建**

* 私有化构造方法
* 提供一个静态方法返回一个实例（getInstance），要使用synchronized同步代码块

如下：

public class GpsInfoService {

private static GpsInfoService gpsInfoService;

private GpsInfoService(Context context) {

this.context = context;

}

private static GpsInfoService getGpsInfoService(Context context) {

if (gpsInfoService == null) {

gpsInfoService = new GpsInfoService(context);

}

return gpsInfoService;

}

}

获取服务，必须要有上下文context，所以方法的参数要传入context

## 18\_号码归属地查询

**正则表达式语法：**

boolean java.lang.String.matches(String regularExpression)

Tests whether this string matches the given regularExpression.

**SQL语句的使用：**

占位符的使用

可以增加选择来电去电的归属地查询

**使用Service，根据来电显示来电号码归属地：**

使用服务，在用户在设置中启用这个功能后启动服务

TelephonyManager=(android.telephony.TelephonyManager) getSystemService(TELEPHONY\_SERVICE);

**注册一个监听器**

TelephonyManager.listen(MyPhoneStateListener, PhoneStateListener.LISTEN\_CALL\_STATE);

MyPhoneStateListener继承PhoneStateListener

重写方法：public void onCallStateChanged(int state, String incomingNumber)

这样就可以按照电话状态来激活相应的动作

**在服务结束的时候，要解除电话状态监听器的注册！**

TelephonyManager.listen(MyPhoneStateListener, PhoneStateListener.LISTEN\_NONE);

MyPhoneStateListener=null;

模仿Toast的源代码，设计一个方式实现来电归属地提示框悬浮在窗口上。

需要用到

WindowManager windowmanager= (WindowManager) this.getSystemService(WINDOW\_SERVICE);

以及如下的示例代码：

WindowManager.LayoutParams params = new LayoutParams();

params.height = WindowManager.LayoutParams.WRAP\_CONTENT;

params.width = WindowManager.LayoutParams.WRAP\_CONTENT;

params.flags = WindowManager.LayoutParams.FLAG\_NOT\_FOCUSABLE

| WindowManager.LayoutParams.FLAG\_NOT\_TOUCHABLE

| WindowManager.LayoutParams.FLAG\_KEEP\_SCREEN\_ON;

params.format = PixelFormat.TRANSLUCENT;

params.type = WindowManager.LayoutParams.TYPE\_TOAST;

params.setTitle("Toast");

params.gravity = Gravity.LEFT | Gravity.TOP;

TextView tv=new TextView(this);

tv.setTextSize(24);

tv.setText(address);

windowmanager.addView(tv , params);

**服务，和其他三大组件类似，运行于当前应用的主线程中。**

所以如果要在服务进行耗时操作，在服务中新开一个线程，或者使用IntentService

**Note that services, like other application objects, run in the main thread of their hosting process.** This means that, if your service is going to do any CPU intensive (such as MP3 playback) or blocking (such as networking) operations, it should spawn its own thread in which to do that work. More information on this can be found in Processes and Threads. The IntentService class is available as a standard implementation of Service that has its own thread where it schedules its work to be done.

## 20\_更改归属地显示样式和位置

点击和触摸的区别

对触摸事件的监听：

tv.setOnTouchListener(new OnTouchListener() {

@Override

public boolean onTouch(View v, MotionEvent event) {

// TODO 自动生成的方法存根

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

break;

case MotionEvent.ACTION\_MOVE:

break;

case MotionEvent.ACTION\_UP:

break;

default:

break;

}

**return true;**

**//True if the listener has consumed the event, false otherwise.**

}

});

}

**更改控件的位置：**

使用public void layout (int l, int t, int r, int b)方法

但是不能再oncreate方法中使用，原因大概是因为如下：

layout方法在view内部是会被调用的，你layout一下了，但是当它重新布局的时候，又会被重新layout。你在onCreate里面layout没有效果是因为android的View的布局不在onCreate里面进行（大概在onResume之前或之后），后续的布局动作把你的layout给覆盖了。

如果想在oncreate方法中使用，可以使用设置布局参数的方式：

android.widget.LinearLayout.LayoutParams params =

(android.widget.LinearLayout.LayoutParams) tv.getLayoutParams();

params.leftMargin = x;

params.topMargin = y;

t.setLayoutParams(params);

.

**在窗体上显示位置信息：**

仿照Toast的源码，Service\_address显示（服务中去修改UI，服务也是属于主线程的吧）

## 21\_创建黑名单数据库

**创建数据库的方法：**

继承SQLiteOpenHelper类

public class DB\_blackNum extends SQLiteOpenHelper {

final String CREATE\_TABLE\_SQL="create table blackNum(\_id integer primary key autoincrement,number nvarchar(20))";

public DB\_blackNum(Context context, String name, CursorFactory factory,

int version) {

super(context, "blacknumber.db", null, 1);

}

@Override

public void onCreate(SQLiteDatabase db) {

db.execSQL(CREATE\_TABLE\_SQL);

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

}

}

## 24\_短信的备份

**如何反编译android代码？**

先从安卓的system-app目录下，导出想要反编译的APK

使用ApkTool，得到androidManifest.xml

解压缩APK，得到classes.dex，使用dex2jar工具反编译得到classes\_dex2jar.jar

使用jd-gui.exe程序打开classes\_dex2jar.jar就可以了

注意使用CMD命令：cd /d 路径

**什么时候使用service，什么时候使用子线程？？**

最近在做一个android的项目，有一个操作是首先将接收的数据处理分析，然后将处理过的数据存储到数据库中。这将是一个十分耗时的工作。我直接将这一操作写在了UI主线程中。结果log中提示不要把耗时的工作在主线程中完成。于是在网上查阅了一下，发现网上有两种说法：一种说写到一个service中去，一种则是开启子线程。于是我就模糊了，到底应该选择哪一种呢？这 里首先应该清楚什么是service，什么是子线程，以及二者的区别。以下是我从网上整理出来的。

“服务不是单一的进程。服务没有自己的进程，应用程序可以不同，服务运行在相同的进程中。

**服务不是线程。可以在线程中工作。**在应用中，如果是长时间的在后台运行，而且不需要交互的情况下，使用服务。

同样是在后台运行，不需要交互的情况下，如果只是完成某个任务，之后就不需要运行，而且可能是多个任务，需需要长时间运行的情况下使用线程。如果任务占用CPU时间多，资源大的情况下，要使用线程。”

比如：程序的进程关了 service还在运行 但线程没有 。当你要做个在APP没运行的情况下 还能给你各种提示（消息推送什么的）就可以开个服务

网络下载图片什么的 就开个线程 因为耗时的程序段是不能阻塞到ui线程（主线程）的

提问者评价

子线程是可以代替service的，但是容易被kill掉。

**service其实是跑在主线程中的，如果需要大量的后台费时数据处理操作，最好的方式是在service中开子线程，而不是直接开一个子线程，这样是为了提高子线程的优先级，而不会轻易被系统杀掉。**

这段文字让我最终决定使用线程。因为**服务也是工作在主线程中的**，如果服务中要执行耗时的工作也是会阻塞主线程，所以得用主线程。

“如果需要完成一项比较耗时的工作,应该通过发送Intent给Service,由Service来完成.这里不能使用子线程来解决,因为BroadcastReceiver的生命周期很短,子线程可能还没有结束，BroadcastReceiver就先结束了.BroadcastReceiver一旦结束,此时BroadcastReceiver的所在进程很容易在系统需要内存时被优先杀死,因为它属于空进程(没有任何活动组件的进程).如果它的宿主进程被杀死,那么正在工作的子线程也会被杀死.所以采用子线程来解决是不可靠的.”

当看到上面这段文字后，我曾武断地认为我应该选择服务。但是我们没有想到我定义的BroadcastReceiver是在主线程定义的，所以是不会出现上面的情况。

**XML文件的读写操作：**

**读取：确定读取之后的数据用来插入到Uri（如短信的备份）还是返回对象的集合？**

例如返回对象集合：

XmlPullParser pullParser = Xml.newPullParser(); **//解析器**

pullParser.setInput(**·**, "UTF-8"); //**为Pull解释器设置要解析的XML数据inputstream**

int event = pullParser.getEventType();

while (event != XmlPullParser.END\_DOCUMENT){

switch (event) {

case XmlPullParser.START\_DOCUMENT:

persons = new ArrayList<Person>();

break;

case XmlPullParser.START\_TAG:

if ("person".equals(pullParser.getName())){

int id = Integer.valueOf(pullParser.getAttributeValue(0));

person = new Person();

person.setId(id);

}

if ("name".equals(pullParser.getName())){

String name = pullParser.nextText();

person.setName(name);

}

if ("age".equals(pullParser.getName())){

int age = Integer.valueOf(pullParser.nextText());

person.setAge(age);

}

break;

case XmlPullParser.END\_TAG:

if ("person".equals(pullParser.getName())){

persons.add(person);

person = null;

}

break;

}

event = pullParser.next();

}

return persons;

}

**写入：先要获取写入XML文件中的对象集合，一般使用List存储。**

File file = new File("/sdcard/smsBackup.xml");

OutputStream out = new FileOutputStream(file);

XmlSerializer serializer = Xml.newSerializer();

serializer.setOutput(out, "UTF-8");

serializer.startDocument("UTF-8", true);

serializer.startTag(null, "smss");

然后就可以遍历对象的集合，在依次按照Tag写入XML文件中：

serializer.startTag(null, "count");

serializer.text(smsinfos.size()+"");

serializer.endTag(null, "count");

**子线程中如何显示Toast：**

//子线程中显示Toast

Looper.prepare();

Toast.makeText(getApplicationContext(), "短消息备份完成", Toast.LENGTH\_SHORT).show();

Looper.loop();

## 26\_程序管理器

**如何获取所有已安装程序的信息？**

* 得到Packagemanager
* 通过Packagemanager.getInstalledPackages()方法获取所有已经安装的应用packageInfo信息的集合
* 遍历packageInfo信息的集合，可以得到每个单独应用的图标，包名，应用名

**得到包名的情况下，如何得到这个应用的其他信息：**

使用PackageManager：

PackageInfo packageInfo = getPackageManager().getPackageInfo(

packageName,

PackageManager.GET\_UNINSTALLED\_PACKAGES

| PackageManager.GET\_ACTIVITIES);

packageInfo.applicationInfo可以得到当前应用的applicationInfo

applicationInfo类：

Information you can retrieve about a particular application. This corresponds to information collected from the AndroidManifest.xml's <application> tag.

**如何查看安卓某一功能的源代码？**

* GIt上下载android 的源代码，可以搜索相关的关键词，比如setting
* 下载后，搜索关键词，比如一些label，找到源代码

**如何区分系统/第三方应用？**

源代码：

/\*\*

\* 判断某个应用程序是 不是三方的应用程序

\* @param info

\* @return

\*/

public boolean filterApp(ApplicationInfo info) {

if ((info.flags & ApplicationInfo.FLAG\_UPDATED\_SYSTEM\_APP) != 0) {

return true;

} else if ((info.flags & ApplicationInfo.FLAG\_SYSTEM) == 0) {

return true;

}

return false;

}

## 27\_listview的优化

化的问题:

1. 怎么去优化 listview 怎么优化gridview

2. 怎么优化java虚拟机

所有的优化问题:

1.时间换时间

360安全卫士 qq电脑管家

(禁用一些开机启动项,延时加载一些开机启动)

优化android系统的开机启动时间.

sd卡 .mp3 .jpg

2.空间换时间

图片资源 sd->内存 -> 显示

定义内存缓存-> 显示

**Listview中的优化**

设置需要用到的Textview或者ImageView等为Static型，这样内存中就只需要一个对象即可。

利用getview方法中的Converview参数，可以利用缓存的列表框信息。

@Override

public View getView(int position, View convertView, ViewGroup parent) {

View view;

if (convertView == null) {

// Log.i(TAG,"通过资源文件 创建view对象");

view = View.inflate(context, R.layout.app\_item, null);

} else {

// Log.i(TAG,"使用历史缓存view对象");

view = convertView;

}

imageView = (ImageView) view.findViewById(R.id.iv\_app\_icon);

textView = (TextView) view.findViewById(R.id.tv\_app\_name);

imageView.setImageDrawable(appInfos.get(position).getAppIcon());

textView.setText(appInfos.get(position).getAppName());

return view;

}

**分批地加载数据：微博，微信之类就是这么干的啊……**

**分页加载数据：**

**懒加载 ：滚动的时候不加载数据，只有停止滚动才加载……似乎很多购物APP都这么干啊**

## 28\_程序管理器的ui

**popupWindow的创建和使用**

popupwindow 必须设置一个背景（透明也可以），否则易出现问题……

**如何保证当前窗体界面只有一个Popupwindow呢？即一个实例存在**

定义为类相关Static

单击事件，每次都初始化一下，如果对象存在，dismiss掉

控件setOnscrollListener监听器每次都dismiss一次

**为控件设置一个Tag，区分多个控件……也是存储数据的一种方式？**

**如何启动一个程序？**

还是使用意图Intent

获取想要启动的应用程序的包名，主类名就可以启动程序了。

相关的方法如下：

PackageInfo packageInfo = getPackageManager().getPackageInfo(

packageName,

PackageManager.GET\_UNINSTALLED\_PACKAGES

| PackageManager.GET\_ACTIVITIES);

ActivityInfo[] activityInfos = packageInfo.activities;// 获取这个包信息对象下的所有activity

if (activityInfos.length > 0) {

ActivityInfo mainActivityInfo = activityInfos[0];

String mainActivity = mainActivityInfo.name;

Intent intent = new Intent();

intent.setClassName(packageName, mainActivity);// 用包名和类名唯一的确定一个意图

startActivity(intent);

} else {

Toast.makeText(Aty\_2\_AppManage.this, "不能启动当前应用程序！",

Toast.LENGTH\_SHORT).show();

}

**如何分享数据（比如微博之类的）？**

Log.i(TAG, "分享程序" + appInfos.get(positon).getAppName());

Intent shareIntent = new Intent();

**shareIntent.setAction(Intent.ACTION\_SEND);**

// 需要指定意图的数据类型

**shareIntent.setType("text/plain");**

//主题

shareIntent.putExtra(Intent.EXTRA\_SUBJECT, "分享");

//详细文本信息

shareIntent.putExtra(Intent.EXTRA\_TEXT,

"推荐你使用一个程序" + appInfo.getAppName());

//启动

shareIntent = Intent.createChooser(shareIntent, "分享");

startActivity(shareIntent);

**如何卸载一个程序？**

Log.i(TAG, "卸载程序" + appInfos.get(positon).getAppName());

// 需求不能卸载系统的应用程序

if (appInfo.isSystemApp()) {

Toast.makeText(this, "系统应用不能被删除", 0).show();

} else {

String uristr = "package:" + packageName;

Uri uri = Uri.parse(uristr);

Intent deleteIntent = new Intent();

deleteIntent.setAction(Intent.ACTION\_DELETE);

deleteIntent.setData(uri);

startActivityForResult(deleteIntent, 0);

}

break;

## 30\_程序管理器的界面切换

**通知Activity更新ListView控件**

从前是使用

adapter.notifyDataSetChanged();

为何没有刷新？？

原因在于，在使用这个方法之前，必须把实现baseadapter的类中的getView方法中的传入的集合参数数据，也同时改变。

这就要求在适配器Adapter类中再定义一个setlist的方法，改变集合参数。

**切换用户程序和所有程序界面：**

关键：

判断是否是应用程序的方法

使用adapter.notifyDataSetChanged();通知Activity更新ListView

## 31\_程序锁的界面

将某个应用程序锁定，用户打开应用程序的时候，需要输入密码。

分析一下这个功能如何去做？

1. 用户要开启的这个应用是那一个应用.

* 寻找系统里面是不是这样的广播事件(打开一个应用程序) ，查看Intent的源代码，没有发现，舍弃该方法。
* LogCat窗体中可以发现，每次打开一个应用，都会有Tag为ActivityManager的条目提示，打开了某个Activity，Intent之类的信息。

**获取最近打开应用信息的方案：**

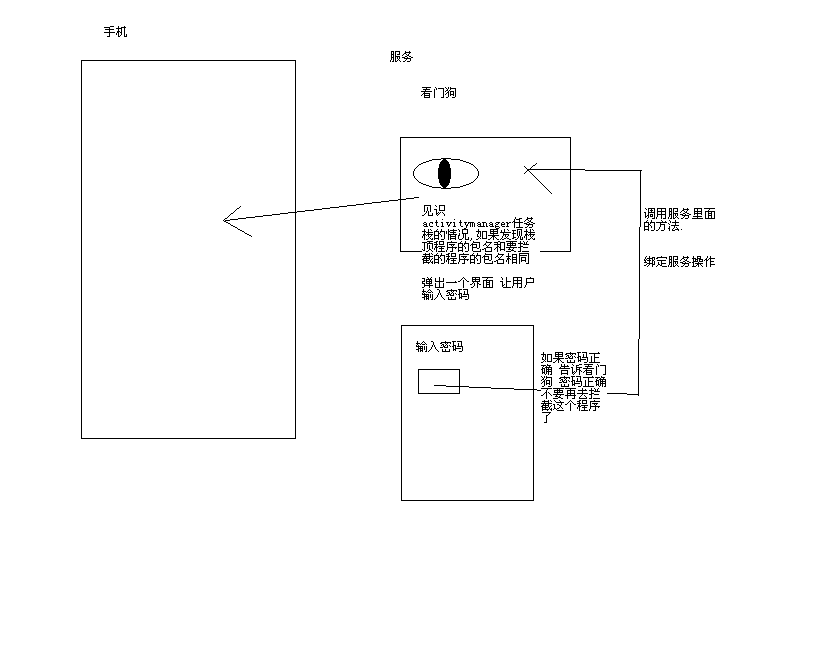
getSystemService(Activity\_Service);可以得到**ActivityManager**.

然后可以调用ActivityManager对象的getRunningTasks(Num)返回最近的N个任务集合。

用户看得到的Activity是当前应用的栈顶

需要权限：android.permission.getTask

2. 判断这个应用的包名 程序名 是否是要锁定的应用名字一致



既要使服务长时间运行，又要可以和服务交互：

可以先StartService，然后在BindService

**想清楚什么时候是需要物理性的去存储一些数据……不要钻牛角尖了……**

**利用Cmd窗口对手机进行操作**

Adb shell进入……

Cd 命令调整当前窗口

Ls命令展现当前窗口下的文件夹

Sqlite3 数据库名 打开数据库，然后可以使用Sql语句进行查询操作了。

<https://www.evernote.com/shard/s209/sh/2c73f0a3-bd09-4b46-8127-16b0b8da3357/7c9c3e02cca4cd08359fda5e5994b0b8>

注意，**删除数据库**貌似不能直接drop 数据库名来实现。

只能借助于直接删除文件的命令

#rm -r xxx //删除名字为xxx的文件夹及其里面的所有文件

#rm xxx //删除文件xxx

#rmdir xxx //删除xxx的空文件夹

**查询表结构的命令：**

select \* from sqlite\_master where type = "table";

我的Lg手机屏蔽了data目录，可以通过cd命令找到，但是ls命令提示没有权限……

cd ..命令可以返回上一级文件夹！

解决方式 su 命令，然后在手机上点授权就OK拉!

ls -l 命令，可以得到当前目录下的所有文件详细信息

cat命令 打开一个文件

## 33\_程序锁的看门狗

使用服务启动子线程轮询当前的启动的Activity是否在被锁名单中。

* 得到当前ActivityManage中提供的当前运行程序的信息
* 判断是否在被锁清单内（数据库操作）
* 如果是，在当前服务类中启动一个Activity，要求用户输入密码
* 密码正确，finish掉Activity，也就是遮挡物撤去，可以观察到程序的界面
* 同时，为了避免Activity重新遮挡的Bug（监视服务开启的子线程是while true循环……），需要让看门狗服务启动的线程，暂时忽略掉这个应用程序。 这样就涉及到让Activity和看门狗service交互的问题了，而要和一个服务交互，只能使用BindService方法。这就需要在看门狗服务中开发一个返回到Activity中serviceConnection的Ibinder对象了。
* 什么时候程序再上锁？
* 过60秒后
* 回到桌面后
* 锁屏后----最佳方案啦

**获取当前锁屏状态**

KeyguardManager keyguardManager = (KeyguardManager) getSystemService(KEYGUARD\_SERVICE);

if(keyguardManager.inKeyguardRestrictedInputMode()){

StopAppProtect\_PackNames.clear();

Log.i(TAG, "进入锁屏……清空忽略名单");

}

* 要实现对锁定程序清单实时的更新，也就是：锁定清单数据发生改变，就应该可以锁定相应的程序。问题是，如果每次循环都去刷新，开销太大了。所以方法应该是采用ContentProvider的方式，自定义一个内容提供者，然后就可以通过一个Observer观察者来监视，其中的数据是否改变了。（自定义一个ContentProvider有点繁琐，是否有其他更好的方法？）
* 另外出现的一个问题是：弹出锁屏界面，密码输入正确之后，锁屏界面finish之后，出现的是 手机卫士的界面，这是为什么？因为锁屏界面的Activity是存放在手机位置的Activity栈中的，当然就会出现这样的情况。那怎么样才能避免这样的情况？只有让锁屏界面的Activity始终在一个单独的栈中，在xml文件中定义Activity的启动模式为singleInstance，单例模式。
* **Activity的四种加载模式复习**

**服务中启动Activity：**

lockappintent = new Intent(this, LockScreenActivity.class);

// 服务是不存在任务栈的 要在服务里面开启activity的话 必须添加这样一个flag

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

**startService bindservice的区别 应用场景**

同时startsevice开启服务 保证服务长期后台运行

Bindservice 绑定服务 ,调用服务里面的方法.

停止的处理 ->

如果绑定过了服务->显示停止绑定服务.

**阻止按键事件**

Onkeydown方法，返回true

**自定义ContentProvider，并监听她的改变**

其实ContentProvider就是一个包装数据库操作的东西……类似网站上的链接包装内部的数据操作？

**使用Receiver监听屏幕关闭事件：**

LockScreenReceiver 注册

android.intent.action.SCREEN\_OFF

android.intent.action.SCREEN\_ON

## 38\_进程管理器的ui&自定义adapter

分类显示系统任务和用户任务：

复杂ListView的自定义：

控制好getView方法返回的不同position的不同View就可以了。

研究一下这种情况下，如何去复用convertView?提高性能？

**ListView中的CheckBox存在Bug……**

手动的去选择，然后拖动，会使之前的选择失效……

## 39\_进程管理器\_杀死后台进程

**创建一个桌面小控件：**

* 创建类继承AppWidgetProvider
* AndroidMainifest.xml文件中注册Receiver,指定相关的Resource资源文件……
* Widget的尺寸
* 更新时间
* 初始的布局，Layout
* 配置文件，注意这个如果没有配置好就不要使用，否则报应用未安装提示
* 部署应用，预览桌面小控件

**Widget工具的生命周期：**

OnEnable

OnRceive

OnUpdate

OnDelete

**Widget是显示在桌面的应用程序中，因此不同的桌面他们的widget的创建和销毁对应的毁掉事件顺序上可能不同！**

**异常：**

10-15 20:05:14.660: W/AppWidgetServiceImpl(299):

updateAppWidgetProvider:

providerdoesn'texist: ComponentInfo

{ com.OldHandsomeTiger.tigersafe/com.OldHandsomeTiger.service.Service\_UpdateWidget}

10-15 20:08:24.252: W/AppWidgetServiceImpl(299): updateAppWidgetProvider: provider doesn't exist: ComponentInfo{ com.OldHandsomeTiger.tigersafe/com.OldHandsomeTiger.receiver.Service\_UpdateWidget}

因为组件的包名没有正确……

ComponentName componentName = new ComponentName(

" com.OldHandsomeTiger.tigersafe",

"com.OldHandsomeTiger.receiver.ProcessWidgetReceiver");

**对pendingIntent类的理解：**

pendingIntent是一种特殊的Intent。主要的区别在于Intent的执行立刻的，而pendingIntent的执行不是立刻的。pendingIntent执行的操作实质上是参数传进来的Intent的操作，但是使用pendingIntent的目的在于它所包含的Intent的操作的执行是需要满足某些条件的。

主要的使用的地方和例子：通知Notificatio的发送，短消息SmsManager的发送和警报器AlarmManager的执行等等。

PendingIntent主要持有的信息是它所包装的Intent和当前Application的Context。正由于PendingIntent中保存有当前Application的Context，使它赋予带他程序一种执行的Intent的能力，就算在执行时当前Application已经不存在了，也能通过存在PendingIntent里的Context照样执行Intent。

要得到一个pendingIntent对象，使用方法类的静态方法 getActivity(Context, int, Intent, int),getBroadcast(Context, int, Intent, int),getService(Context, int, Intent, int) 分别对应着Intent的3个行为，跳转到一个activity组件、打开一个广播组件和打开一个服务组件。

**如何获取手机的内存大小？和手机的相关硬件信息？**

要获取Android手机总内存大小，只需读取"/proc/meminfo"文件的第1行，并进行简单的字符串处理即可。

**正则表达式的使用**

**如果需要调用一个类，但是这个类被Android隐藏了，应该使用反射的技术：**

//通过反射得到被隐藏的类，构造器，方法。并构造对象执行方法

需要注意，只能拿到android.jar包下面的类，别的应用程序下面的类是无法拿到的，比如com.android.setting包下面的类，就属于是另一个应用程序了。所以，反射并不是万能的啊……

**使用反射的步骤详解**

try {

//首先，你得知道需要调用的全路径包类名android.widget.AppSecurityPermissions

Class clazz = getClass().getClassLoader().loadClass("android.widget.AppSecurityPermissions");

//其次，你的知道需要得到哪个构造方法。如果不确定，先得到全部构造方法，遍历一下。

Constructor constructor=clazz.getConstructor(new Class[]{Context.class,String.class});

//然后，你就可以拿着这个构造方法去new Instance了

Object object=constructor.newInstance(new Object[]{this,"com.OldHandsomeTiger.tigersafe"});

//再然后，你就可以拿着这个对象去得到方法，当然你得知道方法的名字和参数，不知道的话，遍历吧

Method method=clazz.getMethod("getPermissionsView", new Class[]{});

/最后，执行方法吧，传入的是之前构造的对象，还有参数

View view=(View) method.invoke(object, new Object[]{});

//于是，一切水到渠成……

setContentView(view);

} catch (Exception e) {

e.printStackTrace();

}

**为了跨Activity传递对象，必须知道：(但是在做这件事情事前，想一想，是否有必要必须传递对象，可否用其他简单的方式代替)**

**如何通过Intent传递一个对象？有两个方法：**

* **让对象实现序列化接口（serializable（通过磁盘） 或 parcelable（通过内存））**

对象实现parcelable接口：看提示实现必要的3个方法：

public static final Parcelable.Creator<Book> CREATOR = new Parcelable.Creator<Book>() {

public Book createFromParcel(Parcel in) {

return new Book(in);

}

public Book[] newArray(int size) {

return new Book[size];

}

};

private Book(Parcel in) {

bookName = in.readString();

author=in.readString();

publishTime=in.readInt();

}

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeString(bookName);

dest.writeString(author);

dest.writeInt(publishTime);

}

* **通过整个应用程序的上下文**

创建一个MyApplication类继承application类

**监测锁屏事件无法通过在Xml中注册Receiver的方式实现**

猜测是Bug，只能使用在程序中进行注册的方式实现,模拟器和真机都不行……

定义IntentFilter，实例化Receiver，注册Receiver

IntentFiler filter=new IntentFilter();

filter.addAction(比如说是锁屏事件的的字符串android.intent.action.SCREEN\_OFF)

filter.setpriority(1000);

Receiver receiver=new Receiver();

registerReceiver(receiver,filter);

## 43\_常用号码查询

**ExpandableListView的用法：**

**继承BaseExpandableListAdapter，**

**主要实现5个方法：**

* **/**/返回分组group的总数

@Override

public int getGroupCount()

* //返回对应group处child的数目

@Override

public int getChildrenCount(int groupPosition)

* //分组信息的View对象

@Override

public View getGroupView(int groupPosition, boolean isExpanded,

View convertView, ViewGroup parent)

* //分组的chileView对象

@Override

public View getChildView(int groupPosition, int childPosition,

boolean isLastChild, View convertView, ViewGroup parent)

* Child的点击事件

@Override

public boolean isChildSelectable(int groupPosition, int childPosition)

**对数据库的操作，rawQuery的用法**

## 44\_流量统计功能

**流量统计原理：**

android 其实就是linux 上面包装了一个java的框架.

linux 系统下 所有的硬件,设备 都是以文件的方式来表示.

文件里面包含的有很多设备的状态信息.

所有的流量相关的信息 都是记录在文件上的.

注意:模拟器　是不支持流量查询的．

proc　系统的状态信息

adb -s 3835197E43F100EC shell

在proc------uid\_stat 的目录下有一堆文件夹

文件夹的名字是以应用程序的uid作为名字的.（系统应用<1000，用户应用=10000+uid号码）

内容就是这个应用程序 上传和下载产生的流量信息

tcp\_rcv文件 记录的是采用tcp协议 接收到的数据的大小

tcp receive

tcp\_snd 采用tcp协议 发送的数据的byte大小

snd send

tcp\_rcv\_pkt 采用tcp协议 接收到的包的数目

使用cat命令查看

以上……就是流量统计的原理，但是谷歌提供了API，所以不必去读文件解析字符串啦……如此繁琐啦!

**流量统计API**

long mobilerx = TrafficStats.getMobileRxBytes(); 得到移动网络接受到的数据大小

long mobiletx = TrafficStats.getMobileTxBytes(); 得到移动网络发送出去的数据大小

long mobiletotal = mobilerx+mobiletx; 总计移动网络产生的流量大小

long totalrx = TrafficStats.getTotalRxBytes(); 得到所有介绍到的数据大小

long toatltx = TrafficStats.getTotalTxBytes(); 得到所有发送出去的数据大小

long total = toatltx+ totalrx; 总计所有流量数据

long wifitotal = total-mobiletotal; wifi网络产生的流量总计

通过应用的uid码获取他的流量信息

PackageInfo packinfo = pm.getPackageInfo(packname, 0);

nt uid = packinfo.applicationInfo.uid;

System.out.println("下载流量"+ TextFormater.getDataSize( TrafficStats.getUidRxBytes(uid)));

System.out.println("上传流量"+ TextFormater.getDataSize( TrafficStats.getUidTxBytes(uid)));

获取有图标的应用程序

PackageManager pm = getPackageManager();

Intent intent = new Intent();

intent.setAction("android.intent.action.MAIN");

intent.addCategory("android.intent.category.LAUNCHER");

List<ResolveInfo>resovleInfos=pm.queryIntentActivities(intent, PackageManager.MATCH\_DEFAULT\_ONLY);

**通过 PackageInfo 获取具体信息方法：**

包名获取方法：packageInfo.packageName

icon获取获取方法：packageManager.getApplicationIcon(applicationInfo)

应用名称获取方法：packageManager.getApplicationLabel(applicationInfo)

使用权限获取方法：

packageManager.getPackageInfo(packageName,PackageManager.GET\_PERMISSIONS)

.requestedPermissions

**通过 ResolveInfo 获取具体信息方法：**

包名获取方法：resolve.activityInfo.packageName

icon获取获取方法：resolve.loadIcon(packageManager)

应用名称获取方法：resolve.loadLabel(packageManager).toString()

ApplicationInfo是从一个特定的应用得到的信息。这些信息是从相对应的Androdimanifest.xml的< application>标签中收集到的。

ResolveInfo这个类是通过解析一个与IntentFilter相对应的intent得到的信息。它部分地对应于从AndroidManifest.xml的< intent>标签收集到的信息。

**通过调用PackageManager的方法可以得到两种不同的信息：**

首先要得到manager:

PackageManager manager = getPackageManager();

　　Java代码：

List< ApplicationInfo> appList =

manager.getInstalledApplications(PackageManager.GET.UNINSTALLED\_PAKAGES);

它是通过解析AndroidManifest.xml的< application>标签中得到的，所以它能得到所有的app。

　　Java代码：

　　Intent intent = new Intent(Intent.A CTION\_MAIN,null);

　　intent.addCategory(Intent.CATEGORY\_LAUNCHER);

//通过Intent查找相关的Activity，更准确

　　List< ResolveInfo> appList = manager.queryIntentActivities(intent,0);

　　//它是通过解析< Intent-filter>标签得到有

　　< action android:name=”android.intent.action.MAIN”/>

　　< action android:name=”android.intent.category.LAUNCHER”/>

//这样的app,所以得到的要比第一种方法少(前者比它多那种service、previder等app)。

SlidingDrawer hides content out of the screen and allows the user to drag a handle to bring the content on screen.

SlidingDrawer must define the id of the handle and of the content:

<SlidingDrawer

android:id="@+id/drawer"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:handle="@+id/handle"

android:content="@+id/content">

<ImageView

android:id="@id/handle"

android:layout\_width="88dip"

android:layout\_height="44dip" />

<GridView

android:id="@id/content"

android:layout\_width="match\_parent"

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

</SlidingDrawer>

流量信息 :　上一次开机到现在这个程序产生的流量　．

世面安全软件：

定义一个数据库

packname uid appname lasttrafficdata

手机在关机的时候 是会有一个关机的广播事件ACTION\_SHUTDOWN

下一次开机的时候.

再去展示这个流量统计的界面

为了避免意外的关机……比如说是掉电

每隔5分钟去存储一次……

流量矫正.

其实大体的流量数据.

参考

**禁止某一款应用上网.**

root权限

原理 :使用了一个ｌｉｎｕｘ系统下的防火墙　iptables

iptables\_armv5 allow uid 10035

disallow uid 10074

iptables\_armv5 需要有linux的一些核心库才运行

模拟器不支持iptables

miui cymod

网上有一个开源的项目

droidwall

google code.

## 46\_屏幕适配

屏幕适配:

尽量使用相对布局和线性布局.

不推荐使用绝对布局.

图片,控件的大小,最好都是用dip的单位作为控件的大小.

**dip==dp**

文件的大小使用sp

hvga 48\*48px

qvga 32\*32px

wvga 60\*60px

代码里面有hard code的大小

根据不同的分辨率做不同的适配.

## 50\_手机杀毒

手机杀毒

log管理

混淆打包

有米广告

自动化测试 robotium

mantis bug管理

**杀毒软件的原理:**

1.基于签名的特征码的扫描(hash码-md5特征码的扫描)

有一个病毒数据库, 保存的有 病毒对应的hash.

特征:只能查杀已知的病毒，不能查杀未知的病毒．

瑞星 8个小时

卡巴斯基 6个小时

诺顿 xx小时

杀毒引擎: 获取hash 查询数据库的算法.

病毒库上: 千万条数据

联网->实时更新病毒数据库

服务器集群:　蜜罐．－＞病毒分析工程师．

2.主动防御：

监控敏感ａｐｉ　，

更改浏览器主页.

注册开机启动的行为

应用程序的内存注入

3.启发式扫描:

根据程序敏感的api 提示风险代码

4.云查杀

人工智能:一套复杂的ｉｆ语句

if t t t

模糊逻辑

1万个左右 anroid

几千种病毒

1.窃取通讯费用

2.切用用户隐私 后台手机用户手机的log 进行一些非法.

3.纯恶作剧

4.偷取流量的软件

大点评网 云中书城 企业会去推广软件 2块钱推广费用.

30万 10块钱 300万