

Android 性能优化实验

一、实验目的

- 1、了解 Android 平台性能优化的原理
- 2、掌握 TraceView 的使用
- 3、掌握 Android 应用的优化方法

二、实验条件

- ✓ IBM-PC 兼容机
- ✓ Windows、Ubuntu11.04 或其他兼容的 Linux 操作系统
- ✓ JDK（建议安装 JDK8 及其以上版本）、Android Studio 或 Eclipse with ADT
- ✓ INTEL ATOM 平板

三、实验原理

1、工具 monitor

英语	中文
Incl	调用方法占用时间百分比
Inclusive	调用方法时间(ms)(包括了所有方法的调用)
Excl	执行方法占用时间百分比
Exclusive	执行方法占用时间(ms)(不包括子方法的调用)
Calls+Recur Calls/Total	调用和重复调用的次数
Time/Call	总的时间(ms)

2、常见的优化技巧

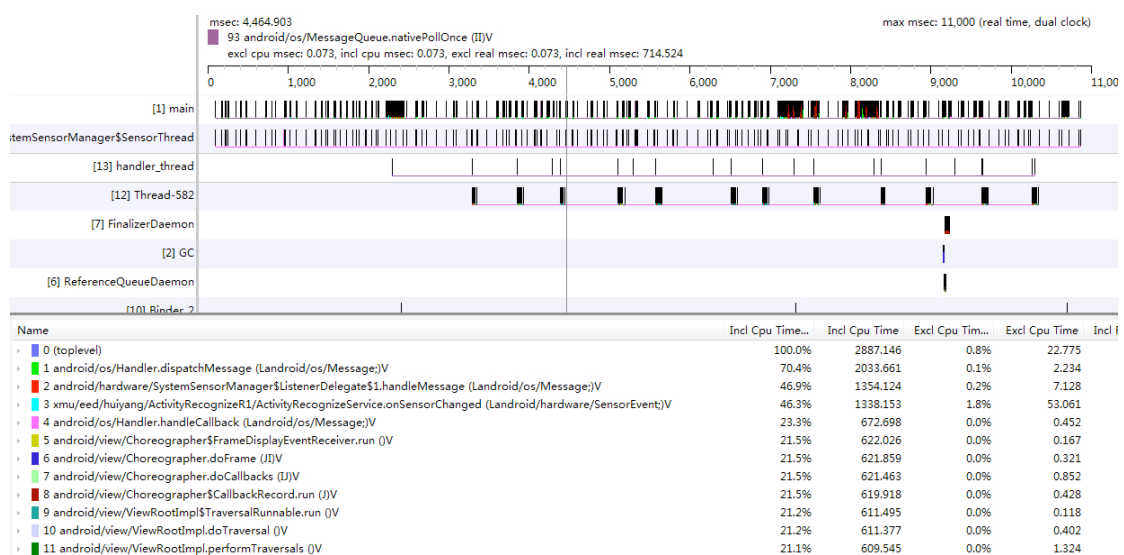
一般的性能优化的技巧：

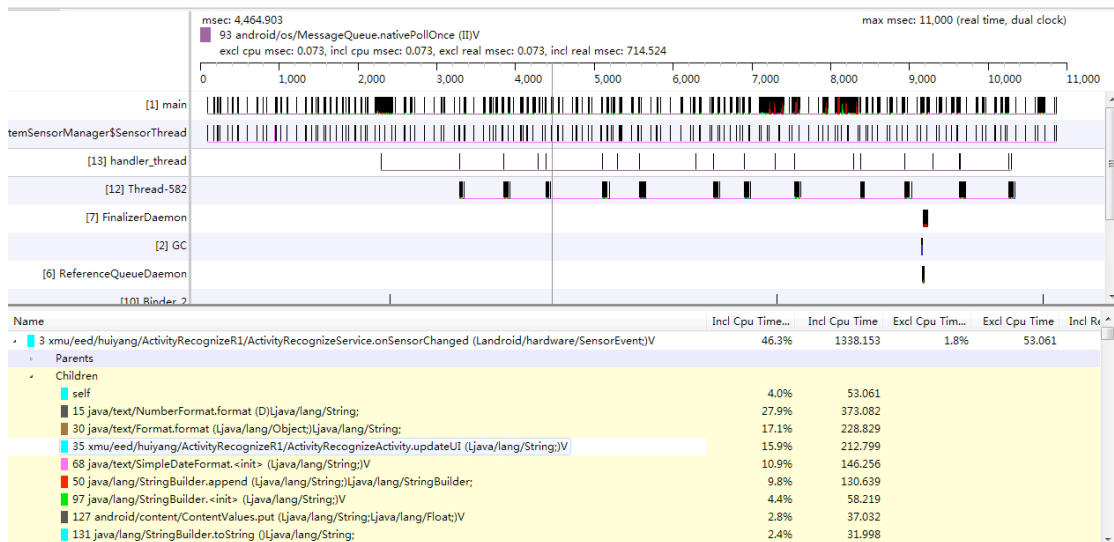
- Avoid Creating Unnecessary Objects
- Prefer Static Over Virtual
- Use Static Final For Constants
- Avoid Internal Getters/Setters
- Use Enhanced For Loop Syntax
- Consider Package Instead of Private Access with Private Inner Classes
- Avoid Using Floating-Point
- Know and Use the Libraries
- Use Native Methods Carefully
- Know And Use The Libraries
- Use Native Methods Judiciously

- 1) 避免创建不必要的对象
- 2) 如果方法用不到成员变量，可以把方法声明为 `static`，性能会提高 15%到 20%
- 3) 避免使用 `getters/setters` 存取 `Field`，可以把 `field` 声明为 `public`，直接访问
- 4) `Static` 的变量如果不需要修改，应使用 `static final` 修饰符定义为常量
- 5) 使用增强 `for` 循环语法 `for ()`
- 6) 私有内部类要访问外部类的 `field` 或方法，可以把外部类的 `field` 或方法声明为包访问权限
- 7) 合理利用浮点数，浮点数比整形数慢两倍

3、方法与步骤

运行最初实验的代码，下载到平板，用 `monitor` 测试，得到优化前的占用时间数据

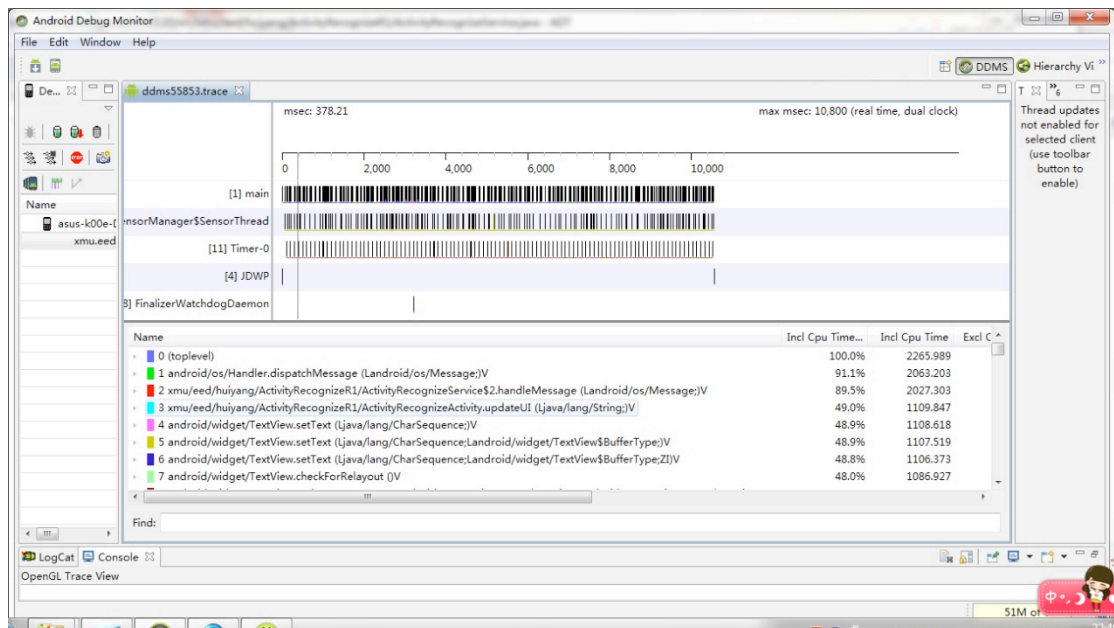




4、根据 method 的位置名称，找到，找到问题修改

因为对于 Android 本身的程序是不能改，还有 java 一些自己的程序都不能改的，所以我们只能修改我们自己写的，也就是我们调用的不能改，我们只能控制使用这些方式，通过修改调用的方式、次数，或者可以找到一些其它的函数替换它，那个函数的效率要高一些，减少这个 method 运行所占用的时间。这里注意，monitor 后面有测试调用多少次的

优化方法：主要是因为它一直一直在更新显示，就会占用比较多的时间，考虑让它更新一段时间再显示，其实就是减少函数调用的次数，就用到定时器，让它规定一段时间在显示。



Name	Incl Cpu Time %	Incl Cpu Time	Excl Cpu Time %	Excl Cpu Time	Calls+RecurCa...	Cpu Time/Call
6 android/widget/FrameLayout.draw (Landroid/graphics/Canvas;)V	50.4%	3404.827	0.1%	3.677	89+178	12.752
7 android/view/View.draw (Landroid/graphics/Canvas;)V	50.4%	3403.790	1.1%	73.673	89+801	3.824
8 android/view/ViewGroup.dispatchDraw (Landroid/graphics/Canvas;)V	49.7%	3352.986	1.0%	64.991	89+801	3.767
9 android/view/ViewGroup.drawChild (Landroid/graphics/Canvas;Landroid/view/View;)L	49.6%	3346.946	2.1%	141.944	89+1246	2.507
10 xmu/eed/huiyang/ActivityRecognizeR1/ActivityRecognizeService\$2.handleMessage (L	44.3%	2993.922	1.4%	94.187	88+0	34.022
Parents						
Children						
11 xmu/eed/huiyang/ActivityRecognizeR1/ActivityRecognizeActivity.updateUI (Ljava/lan	27.3%	1842.311	0.0%	1.316	88+0	20.935
12 android/widget/TextView.setText (Ljava/lang/CharSequence;)V	27.3%	1840.995	0.0%	1.178	88+0	20.920
13 android/widget/TextView.setText (Ljava/lang/CharSequence;Landroid/widget/TextVie	27.2%	1839.817	0.0%	1.423	88+0	20.907
14 android/widget/TextView.setText (Ljava/lang/CharSequence;Landroid/widget/TextVie	27.2%	1838.394	0.2%	14.695	88+0	20.891
15 android/widget/TextView.checkForRelayout ()V	26.8%	1810.583	0.1%	9.550	88+0	20.575
16 android/widget/TextView.onLayout (Landroid/widget/TextView;Landroid/widget/TextVie	26.4%	1719.216	0.1%	0.408	88+0	19.536

修改代码：

占用资源函数

原因是传感器得到的数据一直在刷新显示，但实际上这是不用的，因为我们要观察数据也是应该是得到一段时间一段时间抽样数据，所以不用一直一直调用显示，所以加入一个定时器，一段时间过后再显示一次。

@Override//原来程序

```
public void onSensorChanged(SensorEvent mSensorEvent) {
    String message = new String();

    switch (mSensorEvent.sensor.getType()) {

        case Sensor.TYPE_MAGNETIC_FIELD:
            /*
             * data from magnetic field sensor
             */
            magValues[0] = mSensorEvent.values[0];
            magValues[1] = mSensorEvent.values[1];
            magValues[2] = mSensorEvent.values[2];
            // calculate orientation through magnetic field
            calculateOrientation();
            break;

        case Sensor.TYPE_GYROSCOPE:
            /*
             * data from gyroscope sensor
             */
            gyroValues[0] = mSensorEvent.values[0];
            gyroValues[1] = mSensorEvent.values[1];
            gyroValues[2] = mSensorEvent.values[2];
            break;

        case Sensor.TYPE_ACCELEROMETER:
            /*
             * data from accelerometer sensor
             */
            accValues[0] = mSensorEvent.values[0];
```

```

accValues[1] = mSensorEvent.values[1];
accValues[2] = mSensorEvent.values[2];

message = "Acc:\n";
message += "accX:\t" + df.format(accValues[0]) + "\n";
message += "accY:\t" + df.format(accValues[1]) + "\n";
message += "accZ:\t" + df.format(accValues[2]) + "\n";

message += "Gyro:\n";
message += "gyroX:\t" + df.format(gyroValues[0]) + "\n";
message += "gyroY:\t" + df.format(gyroValues[1]) + "\n";
message += "gyroZ:\t" + df.format(gyroValues[2]) + "\n";

message += "Mag:\n";
message += "magX:\t" + df.format(magValues[0]) + "\n";
message += "magY:\t" + df.format(magValues[1]) + "\n";
message += "magZ:\t" + df.format(magValues[2]) + "\n";

message += "Ori:\n";
message += "oriX:\t" + df.format(oriValues[0]) + "\n";
message += "oriY:\t" + df.format(oriValues[1]) + "\n";
message += "oriZ:\t" + df.format(oriValues[2]) + "\n";

mActivity.updateUI(message);

if (doWrite) {

    SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    curDated = format.format(System.currentTimeMillis());
    ContentValues valuesa = new ContentValues();
    valuesa.put(ACCX, accValues[0]);
    valuesa.put(ACCY, accValues[1]);
    valuesa.put(ACCZ, accValues[2]);
    valuesa.put(GYROX, gyroValues[0]);
    valuesa.put(GYROY, gyroValues[1]);
    valuesa.put(GYROZ, gyroValues[2]);
    valuesa.put(MAGX, magValues[0]);
    valuesa.put(MAGY, magValues[1]);
    valuesa.put(MAGZ, magValues[2]);
    valuesa.put(ORIX, oriValues[0]);
    valuesa.put(ORIY, oriValues[1]);
    valuesa.put(ORIZ, oriValues[2]);
}

```

```

        valuesa.put(LAT, lat);
        valuesa.put(LONG, lng);
        valuesa.put(TIME, curDated );
        saveDataToBuffer(valuesa);
    }
    break;
}
}

```

修改加入定时器:

//启动定时器

```

        timer.schedule(task, 0, 100); //100ms执行一次
    }

```

/**/

```

private Timer timer = new Timer(true);

```

//任务

```

private TimerTask task = new TimerTask() {
    public void run() {
        Message msg = new Message();
        msg.what = 1;
        handler.sendMessage(msg);
    }
};

```

```

private Handler handler = new Handler() {
    public void handleMessage(Message msg) {
        super.handleMessage(msg);
        if(msg.what == 1) {
            //if (D) Log.d(TAG, "message");
            String message;
            message = "Acc:\n";
            message += "accX:\t" + df.format(accValues[0]) + "\n";
            message += "accY:\t" + df.format(accValues[1]) + "\n";
            message += "accZ:\t" + df.format(accValues[2]) + "\n";

            message += "Gyro:\n";
            message += "gyroX:\t" + df.format(gyroValues[0]) + "\n";
            message += "gyroY:\t" + df.format(gyroValues[1]) + "\n";
            message += "gyroZ:\t" + df.format(gyroValues[2]) + "\n";

            message += "Mag:\n";

```

```

        message += "magX:\t" + df.format(magValues[0]) + "\n";
        message += "magY:\t" + df.format(magValues[1]) + "\n";
        message += "magZ:\t" + df.format(magValues[2]) + "\n";

        message += "Ori:\n";
        message += "oriX:\t" + df.format(oriValues[0]) + "\n";
        message += "oriY:\t" + df.format(oriValues[1]) + "\n";
        message += "oriZ:\t" + df.format(oriValues[2]) + "\n";

        mActivity.updateUI(message);

        if (doWrite) {

            SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
            curDated = format.format(System.currentTimeMillis());
            ContentValues valuesa = new ContentValues();
            valuesa.put(ACCX, accValues[0]);
            valuesa.put(ACCY, accValues[1]);
            valuesa.put(ACCZ, accValues[2]);
            valuesa.put(GYROX, gyroValues[0]);
            valuesa.put(GYROY, gyroValues[1]);
            valuesa.put(GYROZ, gyroValues[2]);
            valuesa.put(MAGX, magValues[0]);
            valuesa.put(MAGY, magValues[1]);
            valuesa.put(MAGZ, magValues[2]);
            valuesa.put(ORIX, oriValues[0]);
            valuesa.put(ORIY, oriValues[1]);
            valuesa.put(ORIZ, oriValues[2]);
            valuesa.put(LAT, lat);
            valuesa.put(LONG, lng);
            valuesa.put(TIME, curDated);
            saveDataToBuffer(valuesa);
        }
    }
}

```

四、实验报告要求

实验报告中要包含以下几个部分：

- 1、实验目的
- 2、实验条件
- 3、实验原理
- 4、实验步骤分析
- 5、实验结果与总结

