感知技术与应用

实
验
报
告

学　　院　网安学院

年　　级　2023 级

班　　级　1065

学　　号　2310422

姓　　名　谢小珂

2025 年 3 月 21 日

目录

一、实验目的

本次实验的目的是让大家了解 Android 中光线传感器的基本知识，掌握 Android 中光线传感器的使用方法。

二、实验内容

- 获取光线传感器的值：

编写布局文件 activity_main.xml，包含一个 TextView 用于显示光照强度。

编写程序文件 MainActivity.java，实现以下功能：

1. 初始化 SensorManager 和传感器监听。
2. 在 onResume 中注册监听器，在 onPause 和 onStop 中注销监听器。
3. 在 onSensorChanged 中获取并显示光照强度。

- 设计一个光线传感器应用：

1. 显示当前光照强度。
2. 显示传感器名称、耗电量（mA）和最大测量范围（通过 getName()、getPower()、getMaximumRange()获取）。
3. 实现自动调整屏幕亮度。

三、实验步骤及实验结果

· 实验步骤

1. 编写布局文件 activity_main.xml，具体实现代码如下所示

```
    <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="ht
tp://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:background="@drawable/img"
    >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
/>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toBottomOf="@+id/textView1"

        />
    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toBottomOf="@+id/textView2"
        />
    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toBottomOf="@+id/textView3"
        />
</androidx.constraintlayout.widget.ConstraintLayout>
```

UI 规范要求：

在 activity_main.xml 中需实现：

1. 光强显示组件：动态 TextView
2. 传感器信息组件：多行 TextView 包含传感器型号、功耗(mA)和量程范围(lux)
3. 在现有布局基础上增加背景图配置

2.编写程序文件 MainActivity.java，具体实现代码如下所示

```
package com.example.my_applicationxxk;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.renderscript.Sampler.Value;
import android.app.Activity;
import android.view.Menu;
import android.widget.TextView;
```

```
public class mainactivity extends Activity implements SensorEventListener
{private SensorManager sensor;
private TextView text;
        TextView text2 ;//= (TextView)findViewById(R.id.textView2);
        TextView text3 ;//= (TextView)findViewById(R.id.textView3);
        TextView text4 ;//= (TextView)findViewById(R.id.textView4);
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.mainlayout);
sensor = (SensorManager)getSystemService(SENSOR_SERVICE);
text = (TextView)findViewById(R.id.textView1);
         text2 = (TextView)findViewById(R.id.textView2);
         text3 = (TextView)findViewById(R.id.textView3);
         text4 = (TextView)findViewById(R.id.textView4);

}
@Override
protected void onResume() {
// TODO Auto-generated method stub
sensor.registerListener(this,sensor.getDefaultSensor(Sensor.TYPE_LIGHT)
,
            SensorManager.SENSOR_DELAY_GAME);
super.onResume();
}

@Override
protected void onStop() {
// TODO Auto-generated method stub
sensor.unregisterListener(this);
super.onStop();
}

@Override
public void onAccuracyChanged(Sensor sensor, int accuracy) {
// TODO Auto-generated method stub
}
@Override
public void onSensorChanged(SensorEvent event) {
// TODO Auto-generated method stub
float[] values = event.values;
int sensorType = event.sensor.TYPE_LIGHT;
if(sensorType==Sensor.TYPE_LIGHT)
{
```

```
text.setText("光照强度"+String.valueOf(values[0]));
text2.setText("耗电量"+Float.toString(event.sensor.getPower()));
text3.setText("名称"+event.sensor.getName());
text4.setText("最大范围"+event.sensor.getMaximumRange());
}
}
}
```

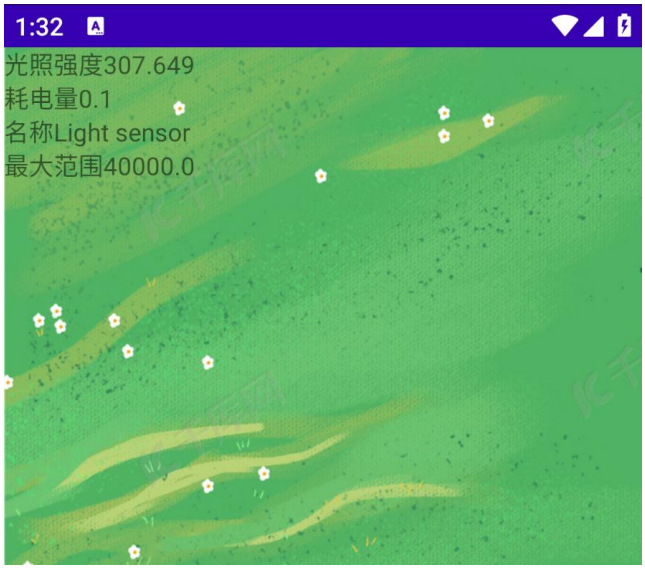
1. 核心功能架构：继承 Activity 并实现 SensorEventListener 接口,同时使用系统 SensorManager 管理光照传感器，再通过 4 个 TextView 组件展示传感器数据。

2.数据实时处理：onSensorChanged()回调中获取 TYPE_LIGHT 数据，将原始数据转换为可读格式如当前光照强度（values[0]）、耗电量（getPower()）、名称（getName()）和最大范围（getMaximumRange()）。

3.动态响应测试

该应用通过 Android 模拟器进行功能验证，部署应用后，使用模拟器的虚拟传感器控制面板动态调节光照强度,观察界面能否正确显示实时光照数值及传感器元数据（名称、耗电量和最大范围）。

·实验结果



四、实验遇到的问题及其解决方法

1.多 TextView 布局错乱，传感器信息显示区域在小屏幕设备上出现重叠。原因是使用绝对布局且未适配不同分辨率，改用 ConstraintLayout 并设置动态约束关系，添加滚动容器 ScrollView。

```
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

2.应用无法通过 USB 连接到测试手机进行调试，需要启用开发者选项，进入手机设置激活开发者模式。在开发者选项中启用 USB 调试 和 USB 安装权限。

3.导出 APK 文件流程，定位项目文件，生成签名密钥。

五、实验结论

1. 实现了 Android 光线传感器的简单功能：实时获取并显示光照强度、完整展示传感器元数据。
2. 通过解决布局重叠问题,深入理解了 ConstraintLayout 的约束原理,在 USB 调试过程中，熟悉了 Android 设备的开发者模式配置，APK 签名导出实践使我掌 握了应用发布前的标准化流程。

这次实验让我真正动手做了一个能用的传感器 APP，虽然有些问题与瑕疵，但最后都想方设法地解决了。不仅学会了技术知识，更重要的是明白了遇到问题要怎么去查资料、想办法。做 APP 开发让我获益匪浅，以后可以尝试做其他传感器的应用。