物 联 网（2023 秋季）

考 核 报 告

班　　级 _____

学　　号 _____

姓　　名 _____

评　　分 _____

中国地质大学（武汉）计算机学院

2024年1月

一、 题目与意义

1.1 题目

现代手机拥有许多传感器，包括地磁、姿态、GPS、光照、温度、气压、摄像、声音、电磁等。自己做一个小应用，获得手机传感器数据，并将它传到 PC 端进行展示。

参考：

https://blog.csdn.net/goldenhawking/article/details/128137382

https://blog.csdn.net/qq_26848623/article/details/119488030

要求：

1、 手机端读取一种或多种传感器数据；

2、 手机端与 PC 端形成网络连接；

3、 手机端的数据发送到 PC 端

4、 PC 端对接收的数据进行存储并显示

加分项：

PC 端对数据可视化显示后，能进行数据加工，以得到数据后面的物理信息。如震动传感器数据可以反映人的走路、跑步状态；GPS 传感器数据能反映位置变化等。思考可应用场景，如何服务于物联网，以满足快速连接、低成本、实时服务的功能？

1.2 意义

本次实验设计多方面的技术，从移动应用开发到网络通信再到数据可视化和物联网应用等等，通过完成这个实验，我可以培养综合性的技能，为未来的移动应用开发和物联网领域做好准备。

二、 题目分析

2.1 题目需求分析

（1）手机端设计一个移动应用。

（2）应用可以读取一种或多种传感器数据。

（3）手机端与 PC 端形成网络连接。

（4）手机端的数据发送到 PC 端。

（5）PC 端对接收的数据进行存储并显示。

2.2 所需技术分析

（1）移动应用开发：使用移动应用开发框架，如 Android Studio 或其它开发工具，设计一个移动端应用，获取手机传感器数据。

（2）网络通信：通过 Socket 编程或使用网络库，确保手机与 PC 端建立可靠的网络连接。

（3）数据传输：实现数据的序列化和反序列化，选择适当的数据传输协议，确保数据在网络中正确传输。

（4）数据存储：在 PC 端存储从手机端接收到的传感器数据。

三、 详细设计

3.1 概要设计

（1）用户界面设计

移动端用户界面：简单设计一个用户界面，提供按钮使用户能选择获取传感器数据和将数据发送到 PC 端，显示获取的传感器数据。

（2）移动端模块设计

传感器数据获取模块：利用 Android 提供的传感器 API，注册传感器监听器以获取实时数据。针对不同传感器类型（加速度计、陀螺仪、磁场等），实现相应的回调函数用于数据获取，并将数据显示在屏幕上。

传感器数据发送模块：获取传感器数据后，将其封装，然后利用 HTTP 或 Socket 协议，建立与 PC 端的连接，然后将数据通过网络发送。

（3）PC 端模块设计

数据接收与解析模块：建立一个服务监听指定端口，接收从手机端发送的传感器数据。实现数据解析逻辑，将接收到的数据提取为可用格式。

数据存储模块：将传感器数据存储到本地文件系统中。

3.2 实验内容

3.2.1 基本信息

IDE：Android Studio Giraffe | 2022.3.1　IntelliJ IDEA 2023.2.1(Community)

PC 端操作系统：Windows 11 22H2

语言：Java

移动端设备：IQOO Z3

移动端操作系统：Android 13

3.2.2 用户界面设计

使用的控件：

表 1 使用的控件

控件名	意义
textView	显示文本“加速度传感器：”
textView2	显示文本“磁场传感器：”
textView3	显示文本“陀螺仪传感器：”
textView_Acc	显示获取的加速度传感器的数据文本
textView_Magnetic	显示获取的磁场传感器的数据文本
textView_Gyroscope	显示获取的陀螺仪传感器的数据文本
button_get	当点击时，获取传感器数据
button_send	当点击时，发送传感器数据

3.2.3 数据结构设计

```
// 标识用于日志或调试的标签
private final String TAG = "sensors";
// 用于显示加速度传感器数据的 TextView
private TextView textViewAcc;
// 用于显示磁场传感器数据的 TextView
private TextView textViewMagnetic;
// 用于显示陀螺仪传感器数据的 TextView
private TextView textViewGyroscope;
// 传感器管理器，用于管理传感器的获取和监听
private SensorManager mSensorManager;
// 自定义的传感器事件监听器，处理传感器数据变化事件
private MySensorEventListener mySensorEventListener;
// 存储加速度传感器数据的数组
private float[] mAccelerometerReading = new float[3];
// 存储磁场传感器数据的数组
private float[] mMagneticFieldReading = new float[3];
// 存储陀螺仪传感器数据的数组
private float[] mGyroscopeReading = new float[3];
```

3.2.4 注册传感器监听器模块

注册传感器监听器模块负责在用户点击获取数据按钮时，启动传感器监听器以获取手机各个传感器的实时数据。具体步骤如下：

（1）获取传感器管理器的实例，用于管理各种传感器的操作。

```
mSensorManager =(SensorManager) getSystemService(SENSOR_SERVICE);
```

（2）创建自定义的传感器事件监听器（MySensorEventListener），该监听器继承自 SensorEventListener 接口。

```
mySensorEventListener = new MySensorEventListener();
```

（3）通过按钮点击事件触发的 registerSensorListeners 方法，注册加速度、磁场和陀螺仪传感器的监听器。如果设备支持相应的传感器，则注册监听器；否则，通过日志输出提示设备不支持该传感器。

```
protected void registerSensorListeners() {
    if (mSensorManager == null) {
        return;
    }
    // 获取加速度传感器
    Sensor accelerometerSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    if (accelerometerSensor != null) {
        // 注册加速度传感器监听器
        mSensorManager.registerListener(mySensorEventListener, accelerometerSensor, SensorManager.SENSOR_DELAY_NORMAL);
    } else {
        Log.d(TAG, "当前设备不支持加速度传感器！");
    }
    // 获取磁场传感器
    Sensor magneticSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
    if (magneticSensor != null) {
        // 注册磁场传感器监听器
        mSensorManager.registerListener(mySensorEventListener, magneticSensor, SensorManager.SENSOR_DELAY_NORMAL);
    } else {
        Log.d(TAG, "当前设备不支持磁场传感器！");
    }
    // 获取陀螺仪传感器
    Sensor gyroscopeSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
    if (gyroscopeSensor != null) {
        // 注册陀螺仪传感器监听器
        mSensorManager.registerListener(mySensorEventListener, gyroscopeSensor, SensorManager.SENSOR_DELAY_NORMAL);
    } else {
        Log.d(TAG, "当前设备不支持陀螺仪传感器！");
    }
}
```

（4）在 Activity 的生命周期方法中，当 Activity 暂停时（onPause），注销所有传感器监听器，释放资源。

```
protected void onPause() {
    super.onPause();
    if (mSensorManager == null) {
        return;
    }
    // 注销所有传感器监听器
    mSensorManager.unregisterListener(mySensorEventListener);
}
```

3.2.5 传感器数据获取模块

传感器数据获取模块负责在用户点击获取数据按钮时，获取传感器的数据并显示在屏幕上。具体步骤如下：

（1）利用 Android 提供的传感器 API，注册传感器监听器以获取实时数据。

（2）在 onSensorChanged 回调中更新 mAccelerometerReading、mMagneticFieldReading 和 mGyroscopeReading 数组。

（3）将获取的数据显示在相应的 TextView 上。

```
public void onSensorChanged(SensorEvent event) {
    if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
        mAccelerometerReading = event.values;
        textViewAcc.setText("[x:" + event.values[0] + ", y:" + event.values[1] + ", z:" + event.values[2] + "]");
    }
    else if (event.sensor.getType() == Sensor.TYPE_MAGNETIC_FIELD) {
        mMagneticFieldReading = event.values;
        textViewMagnetic.setText("[x:" + event.values[0] + ", y:" + event.values[1] + ", z:" + event.values[2] + "]");
    }
    else if (event.sensor.getType() == Sensor.TYPE_GYROSCOPE) {
        textViewGyroscope.setText("[x:" + event.values[0] + ", y:" + event.values[1] + ", z:" + event.values[2] + "]");
    }
}
```

3.2.6 数据封装与发送模块

数据封装与发送模块负责在用户点击发送数据到 PC 按钮时，封装传感器数据并通过 Socket 发送。具体步骤如下：

（1）在按钮点击事件中，获取传感器数据并构建传感器数据字符串。

（2）创建异步任务 SensorDataSenderTask，通过 Socket 将传感器数据发送至 PC 端。

```
String accelerometerData = Arrays.toString(mAccelerometerReading);
String magneticData = Arrays.toString(mMagneticFieldReading);
String gyroscopeData = Arrays.toString(mGyroscopeReading);

String sensorData = "Accelerometer: " + accelerometerData +
                    "\nMagnetic: " + magneticData +
                    "\nGyroscope: " + gyroscopeData;

new SensorDataSenderTask().execute(sensorData);
```

3.2.7 网络通信设计：

在 SensorDataSenderTask 中，通过 Socket 连接 PC 端，发送传感器数据。

```
public class SensorDataSenderTask extends AsyncTask<String, Void, Void> {
    @Override
    protected Void doInBackground(String... params) {
        try {
            Socket socket = new Socket("192.168.137.1", 5000); // 修改为你电脑的 IP 地址和服务器端口
            PrintWriter writer = new PrintWriter(socket.getOutputStream(), true);

            for (String data : params) {
                writer.println(data);
            }

            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }//异常处理
        return null;
    }
}
```

3.2.8 数据接收与解析、存储

（1）在 PC 端建立一个服务，监听指定端口，等待手机端的连接。

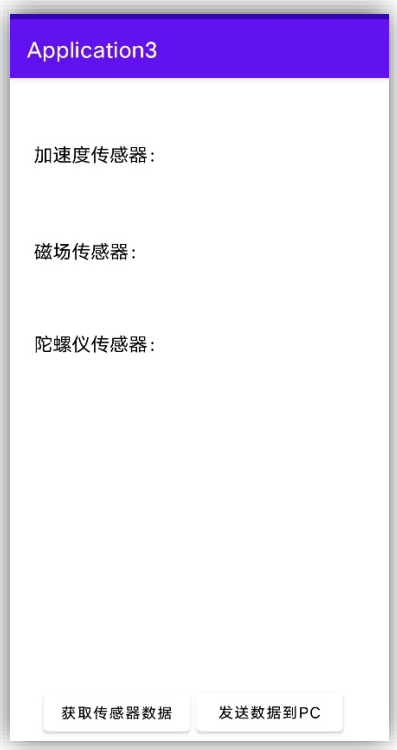
（2）接收传感器数据并解析，确保提取到有效的传感器数据。

（3）将解析得到的数据写入文件。

```java
public class SensorDataReceiver {
    private static final String FILE_PATH = "sensor_data.txt";
    public static void main(String[] args) {
        try {
            // 创建一个文件用于存储传感器数据
            File file = new File(FILE_PATH);
            BufferedWriter writer = new BufferedWriter(new FileWriter(file));
            ServerSocket serverSocket = new ServerSocket(5000); // 使用端口5000
            System.out.println("Server is running and waiting for connection...");

            Socket socket = serverSocket.accept(); // 等待客户端连接
            System.out.println("Client connected!");

            BufferedReader reader = new BufferedReader(new InputStreamReader(socket.getInputStream()));

            while (true) {
                String sensorData = reader.readLine();
                if (sensorData == null) {
                    break;
                }
                System.out.println(sensorData);
                // 将传感器数据写入文件
                writer.write(sensorData);
                writer.newLine();
                writer.flush();
            }

            socket.close();
            serverSocket.close();
            writer.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

四、 效果演示

4. 1 用户界面

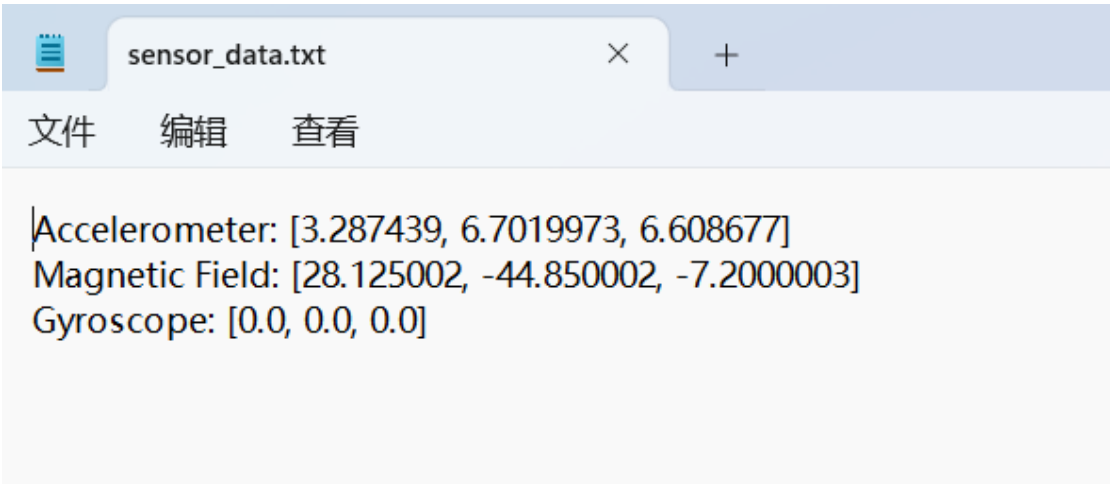


4. 2 获取传感器数据

4.3 发送数据到 PC

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:
Server is running and waiting for connection...
Client connected!
Accelerometer: [3.287439, 6.7019973, 6.608677]
Magnetic Field: [28.125002, -44.850002, -7.2000003]
Gyroscope: [0.0, 0.0, 0.0]
```

4.4 存储数据到本地



五、 总结

本次实验让我接触到了全新的开发平台，在此之前，我只开发过 Windows 平台的程序，这次的安卓开发对我来说是从零开始。首先我遇到的第一个困难就是开发环境的配置。Android Studio 这个 IDE 在下载安装后还需要下载组件，而由于国内是无法访问谷歌服务的，因此需要使用代理或国内镜像源，需要修改好几个文件，并且不能同时使用代理和国内镜像源，不然就等着报错吧。总之，成功配置好开发环境花了我不少时间，一度都想换个 IDE 了。不过配置好后就容易多了，设计控件、实现函数等等，其中在实现 Socket 网络通信时关于接收方的 IP 地址的填写误导了我不久，我以为电脑和手机在同一 WIFI 下即可，事实上得手机连接电脑的热点，然后 IP 地址也得使用这个热点网络的才行。

总得来说，本次实验让我学到了很多新东西，像安卓开发、网络通信等等，但还有很多需要改进的地方，比如应用界面可以设计得更好一点，PC 端可以对数据进行处理，以实现物联网方面的需求，我会在未来努力改进。