

HW12:

Sensor List Activity

Martin Mudenda Bbela

2582912

Overview

Build the sensor listing app using the code provided. In the homework submission, please include the screenshot for all the sensors that are supported by your phone. Please also show at least three sensors' capabilities and their values.

Method

Main Activity

The Main activity is simply a recyclerView linked to a sensor List that gets populated on initializing the Activity.

```
RecyclerView rv;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    SensorManager mSensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
    List<Sensor> mSensorList = new ArrayList<Sensor>();

    List<Sensor> mSensorList = mSensorManager.getSensorList(Sensor.TYPE_ALL);

    Sensor s = mSensorList.get(1);

    LinearLayoutManager mLayoutManager = new LinearLayoutManager(context, this);

    SensorAdapter mSensorAdapter = new SensorAdapter(getApplicationContext(), mSensorList);
    rv = findViewById(R.id.recyclerViewSensors);
    rv.setHasFixedSize(true);
    rv.setAdapter(mSensorAdapter);
    rv.setLayoutManager(mLayoutManager);
}
```

SensorAdapter

This implements a recyclerView adapter. It is setup to take in the SensorList and Populate the recycler view with the names of the sensors. It will also setup an onClickListener for each item to open the SensorOverview Activity to give information about the sensor

```
public class SensorAdapter extends RecyclerView.Adapter<SensorAdapter.MyViewHolder> {
    public static String SENSOR_TYPE_CONSTANT = "sensor_type";
    List<Sensor> mSensorList;
    Context ctx;

    public SensorAdapter(Context ctx, List<Sensor> mSensorList) {
        Log.d(tag, "TAG", "SensorAdapter: ");
        this.mSensorList = mSensorList;
        this.ctx = ctx;
    }

    @Override
    public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        Log.d(tag, "TAG", "onCreateViewHolder: ");
        View v = LayoutInflater.from(parent.getContext())
            .inflate(R.layout.sensor_single_layout, parent, false);
        return new MyViewHolder(v);
    }

    @Override
    public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {
        Log.d(tag, "TAG", "onBindViewHolder: ");
        Sensor s = mSensorList.get(position);
        holder.setName(s.getName());
        holder.setType(s.getType());
        holder.initListener();
    }
}
```

```
public class MyViewHolder extends RecyclerView.ViewHolder {
    View itemView;
    int type;

    public MyViewHolder(@NonNull View itemView) {
        super(itemView);
        this.itemView = itemView;
    }

    public void setName(String sensorName) {
        Log.d(tag, "TAG", "onBindViewHolder: setting name");
        ((TextView) itemView.findViewById(R.id.textViewSensorName)).setText(sensorName);
    }

    public void setType(int type) {
        this.type = type;
    }

    private void initListener() {
        itemView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(ctx, "onClicked", Toast.LENGTH_LONG).show();
                Log.d(tag, "TAG", "onClick: type = " + type);
                Intent startSensorOverviewIntent = new Intent(v.getContext(), SensorOverviewActivity.class);
                //pass sensor type
                startSensorOverviewIntent.putExtra(SENSOR_TYPE_CONSTANT, type);
                v.getContext().startActivity(startSensorOverviewIntent);
            }
        });
    }
}
```

SensorOverviewAvctivity.

This will take the selected sensor and give an overview of information about it. It uses the passed in intent to retrieve all the information about the current Sensor. You can Also click into the Sensor Values to see the readings the current sensor is outputting

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_sensor_overview);

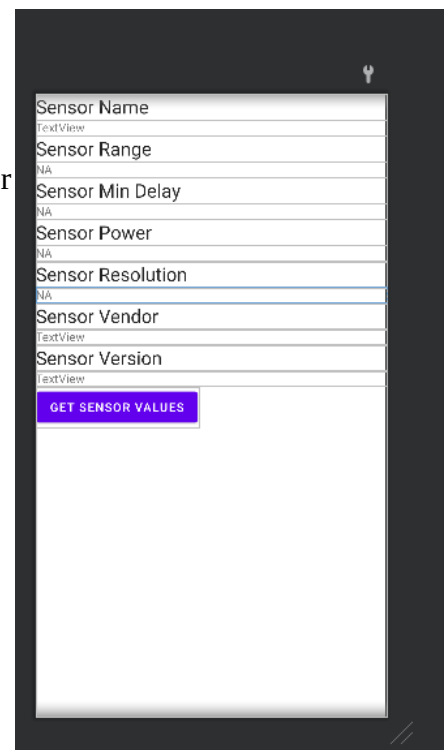
sensorType = getIntent().getIntExtra(SENSOR_TYPE_CONSTANT, defaultValue: -1);

Toast.makeText(getApplicationContext(), "Sensor: " + sensorType, Toast.LENGTH_LONG).show();

tvName = findViewById(R.id.textViewSensorOverviewName);
tvRange = findViewById(R.id.textViewSensorOverviewRange);
tvMinDelay = findViewById(R.id.textViewSensorOverviewMinDelay);
tvPower = findViewById(R.id.textViewSensorOverviewPower);
tvResolution = findViewById(R.id.textViewSensorOverviewResolution);
tvVendor = findViewById(R.id.textViewSensorOverviewVendor);
tvVersion = findViewById(R.id.textViewSensorOverviewVersion);

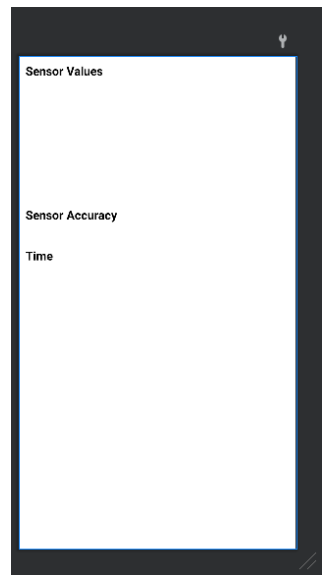
btGetSensorValues = findViewById(R.id.buttonGetSensorValues);

btGetSensorValues.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(getApplicationContext(),
            SensorValuesActivity.class);
        intent.putExtra(SENSOR_TYPE_CONSTANT, sensorType);
        startActivity(intent);
        finish();
    }
});
```



SensorValuesActivity

This Activity will take the current sensor and brought in through the intent and register it to output the values to this screen.



```
private Sensor mSensor

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_sensor_values);

    Intent intent = getIntent();
    mSensorType = intent.getIntExtra(SENSOR_TYPE_CONSTANT, defaultValue: 0);
    mSensorManager = (SensorManager)this.getSystemService(Context.SENSOR_SERVICE);
    mSensor = mSensorManager.getDefaultSensor(mSensorType);
    mEventValue_0 = (TextView)findViewById(R.id.event0);
    mEventValue_1 = (TextView)findViewById(R.id.event1);
    mEventValue_2 = (TextView)findViewById(R.id.event2);
    mEventValue_3 = (TextView)findViewById(R.id.event3);
    mEventValue_4 = (TextView)findViewById(R.id.event4);
    mEventValue_5 = (TextView)findViewById(R.id.event5);
    mEventValue_6 = (TextView)findViewById(R.id.event6);
    mTime = (TextView)findViewById(R.id.time);
    mAccuracy = (TextView)findViewById(R.id.accuracy);
}
```

```
@Override
public void onSensorChanged(SensorEvent event) {
    Log.d("TAG", "onSensorChanged: ");
    mEventValue_0.setText(String.valueOf(event.values[0]));
    mAccuracy.setText(String.valueOf(event.accuracy));
    mTime.setText(String.valueOf(event.timestamp));
    if(event.values.length>1) {
        mEventValue_1.setText(String.valueOf(event.values[1]));
    } if(event.values.length>2) {
        mEventValue_2.setText(String.valueOf(event.values[2]));
    } if(event.values.length>3) {
        mEventValue_3.setText(String.valueOf(event.values[3]));
    } if(event.values.length>4) {
        mEventValue_4.setText(String.valueOf(event.values[4]));
    } if(event.values.length>5) {
        mEventValue_5.setText(String.valueOf(event.values[5]));
    } if(event.values.length>6) {
        mEventValue_6.setText(String.valueOf(event.values[6]));
    }
}
```

Result

User is able to see all the sensors in their current device. The user can further look into MetaData about that sensor and even see what sensor values are being output.

