HW15:

Pedometer App

Martin Mudenda Bbela

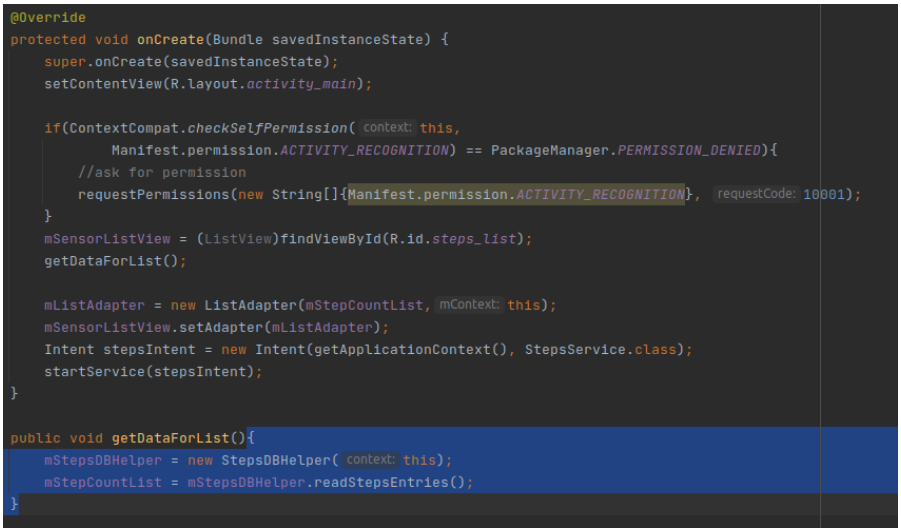
2582912

Overview

Build the pedometer app using the step detector sensor

Method

Using the steps described in the notes I built a step detector that contains a SQL lite Database that will host the number of steps taken on each day. Then Main activity will have a ListView which will be populated through an adapter from the stepCountList. This List pulls from the DB everytime the application starts.



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

    if(ContextCompat.checkSelfPermission( context: this,
            Manifest.permission.ACTIVITY_RECOGNITION) == PackageManager.PERMISSION_DENIED){
        //ask for permission
        requestPermissions(new String[]{Manifest.permission.ACTIVITY_RECOGNITION}, requestCode: 10001);
    }
    mSensorListView = (ListView)findViewById(R.id.steps_list);
    getDataForList();

    mListAdapter = new ListAdapter(mStepCountList, mContext: this);
    mSensorListView.setAdapter(mListAdapter);
    Intent stepsIntent = new Intent(getApplicationContext(), StepsService.class);
    startService(stepsIntent);
}

public void getDataForList(){
    mStepsDBHelper = new StepsDBHelper( context: this);
    mStepCountList = mStepsDBHelper.readStepsEntries();
}
```

The Database is very similar to the DB created for the contact Manager except it will host the date and the number of steps.

```
public class StepsDBHelper extends SQLiteOpenHelper {

    private static final int DATABASE_VERSION = 1;
    private static final String DATABASE_NAME = "StepsDatabase";
    private static final String TABLE_STEPS_SUMMARY = "StepsSummary";
    private static final String ID = "id";
    private static final String STEPS_COUNT = "stepscount";
    private static final String CREATION_DATE ="creationdate";
    private static final String CREATE_TABLE_STEPS_SUMMARY = "CREATE TABLE " + TABLE_STEPS_SUMMARY + "(" + ID +
            " INTEGER PRIMARY KEY AUTOINCREMENT," + CREATION_DATE + " TEXT,"+ STEPS_COUNT + " INTEGER"+")";

    Context context;

    public StepsDBHelper(@Nullable Context context) {
        super(context, DATABASE_NAME,  factory: null, DATABASE_VERSION);
        this.context = context;

    }
```

A service is started from the Main Activity. This service will listen for steps and then output them to the Database everytime a step has been detected.

```
public class StepsService extends Service implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mStepDetectorSensor;
    private StepsDBHelper mStepsDBHelper;
    @Nullable
    @Override
    public IBinder onBind(Intent intent) { return null; }

    @Override
    public void onCreate() {
        super.onCreate();
        Log.d( tag: "TAG",  msg: "onCreate: Starting Service");
        mSensorManager = (SensorManager)this.getSystemService(Context.SENSOR_SERVICE);
        if(mSensorManager.getDefaultSensor(Sensor.TYPE_STEP_DETECTOR) != null)
        {
            Log.d( tag: "TAG",  msg: "onCreate: Enabling step detection");
            mStepDetectorSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_STEP_DETECTOR);
            mSensorManager.registerListener( listener: this, mStepDetectorSensor, SensorManager.SENSOR_DELAY_NORMAL);
            mStepsDBHelper = new StepsDBHelper( context: this);
        }
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        return Service.START_STICKY;
    }

    @Override
    public void onSensorChanged(SensorEvent event) {
        Log.d( tag: "TAG",  msg: "onSensorChanged: creating entry");
        Toast.makeText(getApplicationContext(),  text: "Step detected", Toast.LENGTH_LONG).show();
        mStepsDBHelper.createStepsEntry();
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {

    }
}
```

Result.

User is able to see the number of steps taken each day