CMSC436: Programming Handheld Systems
Sensors
Today’s Topics

Sensor & SensorManager
SensorEvent & SensorEventListener
Filtering sensor values
Example applications
Sensors

Hardware devices that measure the physical environment

Motion
Position
Environment
Some Example Sensors

Motion - 3-axis Accelerometer
Position - 3-axis Magnetic field
Environment - Pressure
Sensor Types

int TYPE_MOTION_DETECT
int TYPE_GRAVITY
int TYPE_AMBIENT_TEMPERATURE
int TYPE_ACCELEROMETER
int TYPE_ALL
Some Sensor Methods

float getResolution()
float getPower()
int getReportingMode()
int getMinDelay()
float getMaximumRange()
SensorEvent

Represents a Sensor event

Data includes

- Sensor type
- Time-stamp
- Accuracy
- Sensor-specific measurement data
Sensor Coordinate System

When default orientation is portrait & the device is lying flat, face-up on a table, axes run

- X – Left to right
- Y – Bottom to top
- Z – Down to up
Sensor Coordinate System

Coordinate system does not change when device orientation changes
SensorManager

System service that manages sensors
Get instance with
   getSystemService(Context.SENSOR_SERVICE)
Access a specific sensor with
   SensorManager.getDefaultSensor(int type)
Some Sensor Type Constants

Accelerometer - Sensor.TYPE_ACCELEROMETER
Magnetic field - Sensor.TYPE_MAGNETIC_FIELD
Pressure – Sensor.TYPE_PRESSURE
Some SensorManager Methods

open fun getSensorList(type: Int): MutableList<Sensor<!>>!
open fun getDefaultSensor(type: Int): Sensor!
SensorEventListener

Interface for SensorEvent callbacks
SensorEventListener

Called when a sensor’s accuracy has changed

abstract fun onAccuracyChanged(
    sensor: Sensor!, accuracy: Int): Unit
Accuracy Constants

SENSOR_STATUS_ACCURACY_HIGH
SENSOR_STATUS_ACCURACY_MEDIUM
SENSOR_STATUS_ACCURACY_LOW
SENSOR_STATUS_NO_CONTACT
SENSOR_STATUS_UNRELIABLE
SensorEventListener

Called when sensor values have changed

    abstract fun onSensorChanged(event: SensorEvent!): Unit

Note: This method should not keep a reference to the SensorEvent
Registering for SensorEvents

Use the SensorManager to register/unregister for SensorEvents
Registering for SensorEvents

Register SensorEventListener for a given sensor

registerListener(listener: SensorEventListener!,
            sensor: Sensor!, samplingPeriodUs: Int): Boolean
Registering for SensorEvents

Unregisters a listener for the sensors with which it is registered

unregisterListener(listener: SensorEventListener!, sensor: Sensor!): Unit
SensorRawAccelerometer

Displays the raw values read from the device’s accelerometer
SensorRawAccelerometer

Raw X: 0.0
Raw Y: 9.809989
Raw Z: 0.0
If the device were standing straight up, the accelerometer would ideally report:

- $X \approx 0 \text{ m/s}^2$
- $Y \approx 9.81 \text{ m/s}^2$
- $Z \approx 0 \text{ m/s}^2$
Accelerometer values

But these values will vary due to natural movements, non-flat surfaces, noise, etc.
Filtering Accelerometer Values

Two common transforms

- Low-pass filter
- High-pass filter
Low-Pass Filter

Deemphasize transient force changes
Emphasize constant force components
High-Pass Filter

Emphasize transient force changes
Deemphasize constant force components
Percussion Instrument
SensorFilteredAccelerometer

Applies both a low-pass and a high-pass filter to raw accelerometer values
Displays the filtered values
SensorFilteredAccelerometer

- Raw X: 0.0
- Raw Y: 9.809989
- Raw Z: 0.0
- LowPass X: 8.15635E-4
- LowPass Y: 9.807158
- LowPass Z: 0.011243063
- HighPass X: -8.15635E-4
- HighPass Y: 0.0028305054
- HighPass Z: -0.011243063
SensorCompass

Uses the device’s accelerometer and magnetometer to orient a compass
Next Time

Maps & Location
Example Applications

SensorRawAccelerometer
SensorFilteredAccelerometer
SensorCompass