CS378 - Mobile Computing

Sensing and Sensors
Sensors

• "I should have paid more attention in Physics 41"
• Most devices have built in sensors to measure and monitor
  – motion
  – orientation (aka position)
  – environmental conditions
• sensors deliver raw data to applications
Sensor Framework

- Determine which sensors are available on a device.
- Determine an individual sensor's capabilities, such as its maximum range, manufacturer, power requirements, and resolution.
- Acquire raw sensor data and define the minimum rate at which you acquire sensor data.
- Register and unregister sensor event listeners that monitor sensor changes.

USING SENSORS
Using Sensors - Basics

• Obtain the SensorManager object
• create a SensorEventListener for SensorEvents
  – logic that responds to sensor event
  – varying amounts of data from sensor depending on type of sensor
• Register the sensor listener with a Sensor via the SensorManager
Using Sensors

```java
private void createSensor() {
    sensorManager =
        (SensorManager) getSystemService(Context.SENSOR_SERVICE);

    showSensors();

    sensorManager.registerListener(sensorEventListener,
        sensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION),
        SensorManager.SENSOR_DELAY_UI);
}
```

- `registerListener(SensorEventListener, Sensor, int rate)`
- `rate` is just a *hint*
- `SENSOR_DELAY_NORMAL`, `SENSOR_DELAY_UI`, `SENSOR_DELAY_GAME`, or `SENSOR_DELAY_FASTEST`, or time in microseconds (millionths of a second)
SensorEventListener

- Interface with two methods:
  - `void onAccuracyChanged (Sensor sensor, int accuracy)`
  - `void onSensorChanged (SensorEvent event)`
    - Sensor values have changed
    - this is the key method to override
  - don't hold onto the event
    - part of pool and the values may be altered soon
private void showSensors() {

    List<Sensor> sensors
        = sensorManager.getSensorList(Sensor.TYPE_ALL);

    Log.d(TAG, sensors.toString());

    for (Sensor s : sensors) {
        Log.d(TAG, s.getName() + " - minDelay: "
            + s.getMinDelay() + ", power: " + s.getPower());
        Log.d(TAG, "max range: " + s.getMaximumRange()
            + ", resolution: " + s.getResolution());
    }
}
Simple Sensor Example

• App that shows linear acceleration
• options to display current
• ... or maximum, disregarding direction
• Linear Layout
• TextViews for x, y, and z
• Buttons to switch between max or current and to reset max
Sensor Coordinate System

• For most motion sensors:
  • +x to the right
  • +y up
  • +z out of front face
  • relative to device
Accelerometer - Includes Gravity

• Sensor. 
  \textit{TYPE\_ACCELEROMETER}

• Device flat on table

• \( g \approx 9.81 \text{ m/s}^2 \)
Sensor Coordinate System

- App that displays max acceleration along each axis
- Hold phone straight up and down and move to ground

![Image showing app display with axes values: x Axis: 3.259, y Axis: 23.662, z Axis: 6.196]
Sensor Coordinate System

- Repeat but hold phone flat
- ... then sideways

![Sensor Test Screen 1](image1)

- x Axis: 5.149
- y Axis: 11.135
- z Axis: 20.017

![Sensor Test Screen 2](image2)

- x Axis: 20.802
- y Axis: 5.436
- z Axis: 8.131
Getting Sensor Data

```java
private void createSensor() {
    sensorManager =
        (SensorManager) getSystemService(Context.SENSOR_SERVICE);

    showSensors();

    sensorManager.registerListener(sensorEventListener,
        sensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION),
        SensorManager.SENSOR_DELAY_UI);
}
```

- `registerListener`
  - `sensorEventListener`
  - `Sensor` - obtain via `SensorManager`
  - `rate of updates, a hint only, or microseconds`
    (not much effect)
- `returns true if successful`
private SensorEventListener sensorEventListener =
    new SensorEventListener() {
        @Override
        public void onSensorChanged(SensorEvent event) {
            // Log.d(TAG, event + "");

            // accelerationValues[0].setText("" + event.values[0])
            if(displayCurrent)
                displayCurrent(event);
            else
                displayMax(event);

            // displayCurrentRotation(event);
        }
    }
private void displayMax(SensorEvent event) {
    for(int i = 0; i < maxVals.length; i++) {
        if(Math.abs(event.values[i]) > maxVals[i]) {
            maxVals[i] = (float) Math.abs(event.values[i]);
            float value = ((int) (maxVals[i] * 1000)) / 1000f;
            accelerationValues[i].setText("" + value);
        }
    }
}

Recall, max range of linear acceleration on dev phone is 19.613 + gravity = 29.423
- a baseball pitcher throwing a fastball reaches 350 m/s² or more (various "physics of baseball" articles)
Display Current

```java
private void displayCurrent(SensorEvent event) {
    if(!zeroingComplete)
        gatherZeroData(event);

    for(int i = 0; i < accelerationValues.length; i++) {
        float value = event.values[i];
        value = ((int) (value * 1000)) / 1000f;
        accelerationValues[i].setText("" + value);
    }
}
```

- Lots of jitter
- Attempt to zero out???
Types of Sensors - Dev Phone - Older

- accelerometer, linear acceleration, magnetic field, orientation, light, proximity, gyroscope, gravity
TYPES OF SENSORS
Sensor Capabilities - Dev Phones - Older

KR3DM 3-axis Accelerometer - minDelay: 20000, power: 0.23
max range: 19.6133, resolution: 0.019153614

AK8973 3-axis Magnetic field sensor - minDelay: 16667, power: 6.8
max range: 2000.0, resolution: 0.0625

GP2A Light sensor - minDelay: 0, power: 0.75
max range: 3626657.8, resolution: 1.0

GP2A Proximity sensor - minDelay: 0, power: 0.75
max range: 5.0, resolution: 5.0

K3G Gyroscope sensor - minDelay: 1190, power: 6.1
max range: 34.906586, resolution: 0.0012217305

Rotation Vector Sensor - minDelay: 20000, power: 13.13
max range: 1.0, resolution: 5.9604645E-8

Gravity Sensor - minDelay: 20000, power: 13.13
max range: 19.6133, resolution: 0.019153614

Linear Acceleration Sensor - minDelay: 20000, power: 13.13
max range: 19.6133, resolution: 0.019153614

Orientation Sensor - minDelay: 20000, power: 13.13
max range: 360.0, resolution: 0.00390625

Corrected Gyroscope Sensor - minDelay: 1190, power: 13.13
max range: 34.906586, resolution: 0.0012217305
Types of Sensors - Dev Phone - Newer

| SensorTest | GP2A Light sensor Sharp       |
| SensorTest | GP2A Proximity sensor Sharp   |
| SensorTest | BMP180 Pressure sensor Bosch |
| SensorTest | MPL Gyroscope Invensense      |
| SensorTest | MPL Accelerometer Invensense  |
| SensorTest | MPL Magnetic Field Invensense |
| SensorTest | MPL Orientation Invensense    |
| SensorTest | MPL Rotation Vector Invensense|
| SensorTest | MPL Linear Acceleration Invensense |
| SensorTest | MPL Gravity Invensense        |
| SensorTest | Rotation Vector Sensor Google Inc. |
| SensorTest | Gravity Sensor Google Inc.    |
| SensorTest | Linear Acceleration Sensor Google Inc. |
| SensorTest | Orientation Sensor Google Inc. |
| SensorTest | Corrected Gyroscope Sensor Google Inc. |
Sensor Capabilities - Dev Phone - Newer

GP2A Light sensor - minDelay: 0, power: 0.75
max range: 646239.5, resolution: 1.0
GP2A Proximity sensor - minDelay: 0, power: 0.75
max range: 5.0, resolution: 5.0
BMP180 Pressure sensor - minDelay: 20000, power: 0.67
max range: 1100.0, resolution: 0.01
MPL Gyroscope - minDelay: 10000, power: 6.1
max range: 34.90656, resolution: 0.57246757
MPL Accelerometer - minDelay: 10000, power: 0.139
max range: 19.6133, resolution: 0.038344003
MPL Magnetic Field - minDelay: 10000, power: 4.0
max range: 8001.0, resolution: 0.012
MPL Orientation - minDelay: 10000, power: 10.239
max range: 360.0, resolution: 1.0E-5
MPL Rotation Vector - minDelay: 10000, power: 10.239
max range: 1.0, resolution: 1.0E-5
MPL Linear Acceleration - minDelay: 10000, power: 0.5
max range: 10240.0, resolution: 1.0
MPL Gravity - minDelay: 10000, power: 10.239
max range: 19.6133, resolution: 0.038344003
Rotation Vector Sensor - minDelay: 10000, power: 10.239
max range: 1.0, resolution: 5.9604645E-8
Gravity Sensor - minDelay: 10000, power: 10.239
max range: 19.6133, resolution: 0.038344003
Linear Acceleration Sensor - minDelay: 10000, power: 10.239
max range: 19.6133, resolution: 0.038344003
Orientation Sensor - minDelay: 10000, power: 10.239
max range: 360.0, resolution: 0.00390625
Corrected Gyroscope Sensor - minDelay: 10000, power: 10.239
max range: 34.90656, resolution: 0.57246757
Types of Sensors

• Three main classes of sensors:
  • motion (acceleration and rotational forces)
    – accelerometers, gravity sensors, gyroscopes, and rotational vector sensors
  • environmental (ambient air temperature and pressure, illumination, and humidity)
    – barometers, photometers, and thermometers.
  • position (physical position of a device)
    – orientation sensors and magnetometers
Types of Sensors

• Hardware sensors
  – built into the device

• Software sensors
  – takes data from a hardware sensors and manipulates it
  – from our perspective acts like a hardware sensor
Types of Sensors

• TYPE_ACCELEROMETER
  – hardware
  – acceleration force in m/s$^2$
  – x, y, z axis
  – includes gravity
Types of Sensors

• TYPE_AMBIENT_TEMPERATURE
  – hardware
  – "room" temperature in degrees Celsius
  – no such sensor on dev phones

• TYPE_GRAVITY
  – software or hardware
  – just gravity
  – if phone at rest same as TYPE_ACCELEROMETER
Types of Sensors

• TYPE_GYROSCOPE
  – hardware
  – measure device's rate of rotation in radians / second around 3 axis

• TYPE_LIGHT
  – hardware
  – light level in lx,
  – lux is SI measure illuminance in luminous flux per unit area
Types of Sensors

• **TYPE_LINEAR_ACCELERATION**
  - software or hardware
  - measure acceleration force applied to device in three axes excluding the force of gravity

• **TYPE_MAGNETICFIELD**
  - hardware
  - ambient geomagnetic field in all three axes
  - uT micro Teslas
Types of Sensors

• TYPE_ORIENTATION [deprecated]
  – software
  – measure of degrees of rotation a device makes around all three axes

• TYPE_PRESSURE
  – hardware
  – ambient air pressure in hPa or mbar
  – force per unit area
  – 1 Pascal = 1 Newton per square meter
  – hecto Pascals (100 Pascals)
  – milli bar - 1 mbar = 1hecto Pascal
Types of Sensors

• TYPE_PROXIMITY
  – hardware
  – proximity of an object in cm relative to the view screen of a device
  – most just binary (see range, resolution)
  – typically used to determine if handset is being held to person's ear during a call

• TYPE_RELATIVE_HUMIDITY
  – ambient humidity in percent (0 to 100)
Types of Sensors

• **TYPE_ROTATION_VECTOR (ABSOLUTE)**
  – hardware or software
  – orientation of device, three elements of the device's rotation vector

• **TYPE_TEMPERATURE**
  – hardware
  – temperature of the device in degrees Celsius
## Availability of Sensors

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Android 4.0 (API Level 14)</th>
<th>Android 2.3 (API Level 9)</th>
<th>Android 2.2 (API Level 8)</th>
<th>Android 1.5 (API Level 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TYPE_ACCELEROMETER</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>TYPE_AMBIENT_TEMPERATURE</strong></td>
<td>Yes</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td><strong>TYPE_GRAVITY</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td><strong>TYPE_GYROSCOPE</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>n/a¹</td>
<td>n/a¹</td>
</tr>
<tr>
<td><strong>TYPE_LIGHT</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>TYPE_LINEAR_ACCELERATION</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td><strong>TYPE_MAGNETIC_FIELD</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>TYPE_ORIENTATION</strong></td>
<td>Yes²</td>
<td>Yes²</td>
<td>Yes²</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>TYPE_PRESSURE</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>n/a¹</td>
<td>n/a¹</td>
</tr>
<tr>
<td><strong>TYPE_PROXIMITY</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>TYPE_RELATIVE_HUMIDITY</strong></td>
<td>Yes</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td><strong>TYPE_ROTATION_VECTOR</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td><strong>TYPE_TEMPERATURE</strong></td>
<td>Yes²</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Sensor Capabilities

• Various methods in Sensor class to get capabilities of Sensor
• minDelay (in microseconds)
• power consumption in mA (microAmps)
• maxRange
• resolution
Linear Acceleration

- At rest of table
- Recall
- Units are m/s²

SensorTest

x Axis
-0.017

y Axis
-0.084

z Axis
-0.034
Zeroing out

• Take average of first multiple (several hundred) events and average
  – shorter time = more error

• Potential error
  – should be 0 at rest

| SensorTest | i: 0, zerovalue: 7.4665865E-4 |
| SensorTest | i: 1, zerovalue: -0.003574672 |
| SensorTest | i: 2, zerovalue: -0.02909316 |

| i: 0, zerovalue: -0.0035472375 |
| i: 1, zerovalue: -0.0018564985 |
| i: 2, zerovalue: -0.022586245 |
Rate of Events

• 1000 events
• SensorManager.SENSOR_DELAY_UI
  – times in seconds: 21, 21, 21
  – 21 seconds / 1000 events
• SensorManager.SENSOR_DELAY_FASTEST
  – times in seconds: 21, 21, 21
• Recall delay of 20,000 micro seconds
• \(2 \times 10^4 \times 1 \times 10^3 = 2 \times 10^7 = 20\) seconds