# CS378 - Mobile Computing

Sensing and Sensors

# Sensors

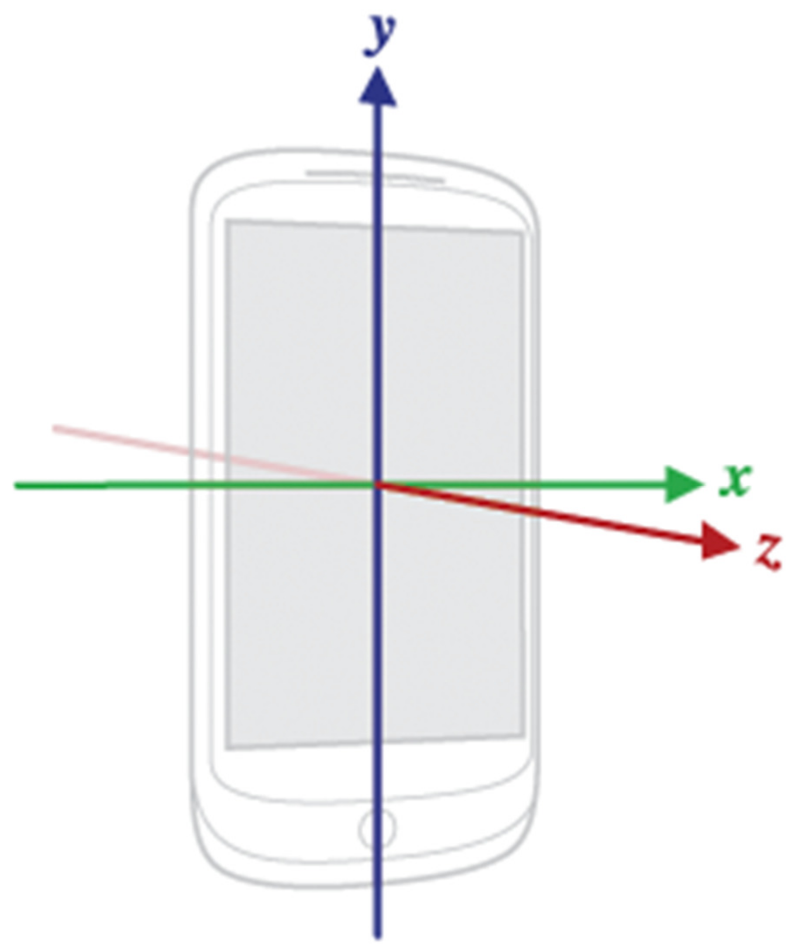
- "I should have paid more attention in Physics 51"
- Most devices have built in sensors to measure and monitor
  - motion
  - orientation (aka position)
  - environmental conditions
- sensors deliver raw data to application

# Using Sensors - Basics

- Obtain the *SensorManager* object
- create a *SensorEventListener* for *SensorEvents*
  - logic that responds to sensor event
  - various amounts of data from sensor depending on type of sensor
- Register the sensor listener with a *Sensor* via the SensorManager

# Sensor Coordinate System

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



# Sensor Coordinate System

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

# Sensor Coordinate System

- Repeat but hold phone flat


... then sideways

# Types of Sensors

- Not every device has every kind of sensor
- Constants from Sensor class
- 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_MAGNETC_FIELD
  - 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
  - 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

| Sensor | Android 4.0 (API Level 14) | Android 2.3 (API Level 9) | Android 2.2 (API Level 8) | Android 1.5 (API Level 3) |
|---|---|---|---|---|
| TYPE_ACCELEROMETER | Yes | Yes | Yes | Yes |
| TYPE_AMBIENT_TEMPERATURE | Yes | n/a | n/a | n/a |
| TYPE_GRAVITY | Yes | Yes | n/a | n/a |
| TYPE_GYROSCOPE | Yes | Yes | n/a[1] | n/a[1] |
| TYPE_LIGHT | Yes | Yes | Yes | Yes |
| TYPE_LINEAR_ACCELERATION | Yes | Yes | n/a | n/a |
| TYPE_MAGNETIC_FIELD | Yes | Yes | Yes | Yes |
| TYPE_ORIENTATION | Yes[2] | Yes[2] | Yes[2] | Yes |
| TYPE_PRESSURE | Yes | Yes | n/a[1] | n/a[1] |
| TYPE_PROXIMITY | Yes | Yes | Yes | Yes |
| TYPE_RELATIVE_HUMIDITY | Yes | n/a | n/a | n/a |
| TYPE_ROTATION_VECTOR | Yes | Yes | n/a | n/a |
| TYPE_TEMPERATURE | Yes[2] | Yes | Yes | Yes |

# Listing Types of Sensor on Device

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

    showSensors();
```

```
private void showSensors() {

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

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

    for(Sensor s : sensors)
        Log.d(TAG, "Sensor: "
                + s.getName() + ", " + s.getVendor());

}
```

# Types of Sensors - Dev Phone

```
Sensor: KR3DM 3-axis Accelerometer, STMicroelectron
Sensor: AK8973 3-axis Magnetic field sensor, Asahi
Sensor: AK8973 Orientation sensor, Asahi Kasei Micr
Sensor: GP2A Light sensor, Sharp
Sensor: GP2A Proximity sensor, Sharp
Sensor: K3G Gyroscope sensor, STMicroelectronics
Sensor: Gravity Sensor, Google Inc.
Sensor: Linear Acceleration Sensor, Google Inc.
Sensor: Rotation Vector Sensor, Google Inc.
```

- accelerometer, linear acceleration, magnetic field, orientation, light, proximity, gyroscope, gravity

# Sensor Capabilities

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

# Sensor Capabilities - Dev Phones

```
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
```

# Getting Sensor Data

```
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

# SensorEventListener

```
private SensorEventListener sensorEventListener =
        new SensorEventListener() {
            @Override
            public void onSensorChanged(SensorEvent event) {
                // Log.d(TAG, event + "");

                // accelerationValues[0].setText("" + event.valu

                // displayCurrent(event);

                displayMax(event);
            }
```

# Display Max

```
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 $m/s^2$ or more (various "physics of baseball" articles)

# Display Current

```
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

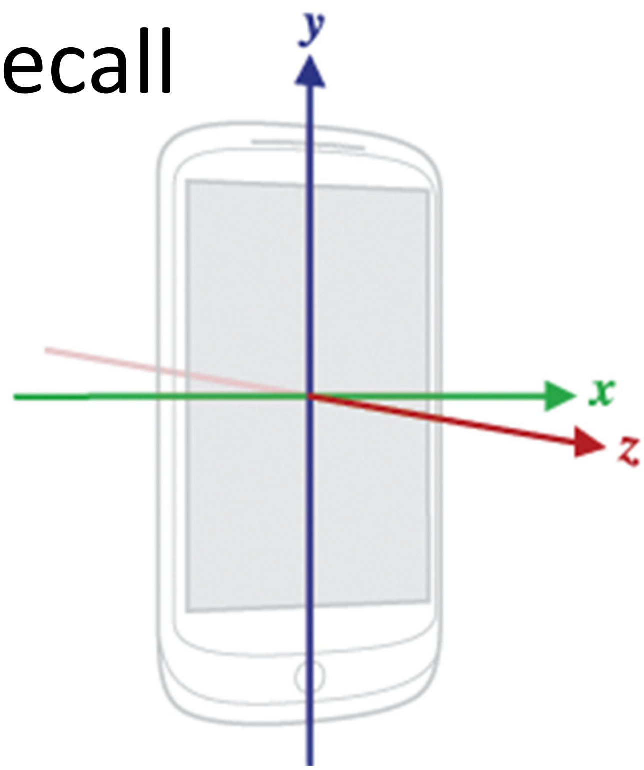
# Accelerometer - Includes Gravity

- Sensor. *TYPE_ACCELEROMETER*
- Device flat on table
- *g* ~= 9.81 $m/s^2$


| | |
|---|---|
| SensorTest | i: 0, zerovalue: -0.19007805 |
| SensorTest | i: 1, zerovalue: -0.42606363 |
| SensorTest | i: 2, zerovalue: 9.776483 |

# Linear Acceleration

- At rest of table
- Recall




- units are m/s$^2$

# Zoeroing 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
- $2x10^4$ x $1x10^3 = 2x10^7$ = 20 seconds