Environment Sensors

The Android platform provides four sensors that let you monitor various environmental properties. You can use these sensors to monitor relative ambient humidity, illuminance, ambient pressure, and ambient temperature near an Android-powered device. All four environment sensors are hardware-based and are available only if a device manufacturer has built them into a device. With the exception of the light sensor, which most device manufacturers use to control screen brightness, environment sensors are not always available on devices. Because of this, it's particularly important that you verify at runtime whether an environment sensor exists before you attempt to acquire data from it.

Unlike most motion sensors and position sensors, which return a multi-dimensional array of sensor values for each `SensorEvent`, environment sensors return a single sensor value for each data event. For example, the temperature in °C or the pressure in hPa. Also, unlike motion sensors and position sensors, which often require high-pass or low-pass filtering, environment sensors do not typically require any data filtering or data processing. Table 1 provides a summary of the environment sensors that are supported on the Android platform.

Table 1. Environment sensors that are supported on the Android platform.

Sensor	Sensor event data	Units of measure	Data description
`TYPE_AMBIENT_TEMPERATURE`	`event.values[0]`	°C	Ambient air temperature.
`TYPE_LIGHT`	`event.values[0]`	lx	Illuminance.
`TYPE_PRESSURE`	`event.values[0]`	hPa or mbar	Ambient air pressure.
`TYPE_RELATIVE_HUMIDITY`	`event.values[0]`	%	Ambient relative humidity.
`TYPE_TEMPERATURE`	`event.values[0]`	°C	Device temperature.[1]

[1] Implementations vary from device to device. This sensor was deprecated in Android 4.0 (API Level 14).

Using the Light, Pressure, and Temperature Sensors

The raw data you acquire from the light, pressure, and temperature sensors usually requires no calibration, filtering, or modification, which makes them some of the easiest sensors to use. To acquire data from these sensors you first create an instance of the `SensorManager` class, which you can use to get an instance of a physical sensor. Then you register a sensor listener in the `onResume()` method, and start handling incoming sensor data in the `onSensorChanged()` callback method. The following code shows you how to do this:

```
public class SensorActivity extends Activity implements SensorEventListener {
  private SensorManager mSensorManager;
  private Sensor mPressure;

  @Override
  public final void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    // Get an instance of the sensor service, and use that to get an instance of
    // a particular sensor.
    mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
    mPressure = mSensorManager.getDefaultSensor(Sensor.TYPE_PRESSURE);
  }

  @Override
  public final void onAccuracyChanged(Sensor sensor, int accuracy) {
    // Do something here if sensor accuracy changes.
  }

  @Override
  public final void onSensorChanged(SensorEvent event) {
    float millibars_of_pressure = event.values[0];
    // Do something with this sensor data.
  }

  @Override
  protected void onResume() {
    // Register a listener for the sensor.
    super.onResume();
    mSensorManager.registerListener(this, mPressure, SensorManager.SENSOR_DELAY_NORMAL);
  }

  @Override
  protected void onPause() {
    // Be sure to unregister the sensor when the activity pauses.
    super.onPause();
    mSensorManager.unregisterListener(this);
  }
}
```

You must always include implementations of both the `onAccuracyChanged()` and `onSensorChanged()` callback methods. Also, be sure that you always unregister a sensor when an activity pauses. This prevents a sensor from continually sensing data and draining the battery.

Using the Humidity Sensor

You can acquire raw relative humidity data by using the humidity sensor the same way that you use the light, pressure, and temperature sensors. However, if a device has both a humidity sensor (`TYPE_RELATIVE_HUMIDITY`) and a temperature sensor (`TYPE_AMBIENT_TEMPERATURE`) you can use these two data streams to calculate the dew point and the absolute humidity.

Dew point

The dew point is the temperature at which a given volume of air must be cooled, at constant barometric pressure, for water vapor to condense into water. The following equation shows how you can calculate the dew point:

$$t_d(t, RH) = T_n \cdot \frac{\ln(\frac{RH}{100}) + \frac{m \cdot t}{T_n + t}}{m - [\ln(\frac{RH}{100}) + \frac{m \cdot t}{T_n + t}]}$$

Where,

t_d = dew point temperature in degrees C

t = actual temperature in degrees C

RH = actual relative humidity in percent (%)

m = 17.62

T_n = 243.12

Absolute humidity

The absolute humidity is the mass of water vapor in a given volume of dry air. Absolute humidity is measured in grams/meter^3. The following equation shows how you can calculate the absolute humidity:

$$d_v(t, RH) = 216.7 \cdot \frac{\frac{RH}{100} \cdot A \cdot \exp(\frac{mt}{T_n + t})}{273.15 + t}$$

Where,

d_v = absolute humidity in grams/meter^3

t = actual temperature in degrees C

RH = actual relative humidity in percent (%)

m = 17.62

T_n = 243.12 degrees C

A = 6.112 hPa