안드로이드 기반 센서 구성 및 센서 프로그래밍 기법

최수경*, 박영호*
*숙명여자대학교 멀티미디어과학과
e-mail: {sukyungchoi, yhpark}@sookmyung.ac.kr

A Study on Sensor Organization and Programming based on Android Platform

Su-Kyung Choi*, Young-Ho Park*
*Dept of Multimedia Science, Sookmyung Women's University

요 약

스마트폰이 제공하는 다양한 애플리케이션들은 스마트폰이 탑재한 센서들과 밀접한 관계가 있다. 본 논문에서는 최근 진화를 거듭하고 있는 스마트폰 센서의 종류와 이를 구현하기 위해 센서 매니저를 이용하는 법과 센서 값을 해석하는 법에 대해 분석해본다. 또한 이를 바탕으로 기기의 센서 목록을 읽어내는 간단한 프로그래밍 예를 소개하고 이를 설명한다.

1. 서론

최근 스마트폰은 단순히 인터넷만 가능한 통신 기기가 아니라 마이크, 카메라, 가속도 센서, 나침반, 온도계, 조도 센서 등을 갖추어 사용자의 지각 능력을 확장시켜 주는 기기가 되었다. 전자식 나침반, 중력 센서, 조도 센서, 근접 센서의 결합은 증강 현실이나 신체 움직임에 기반한 입력처럼 새로운 방식으로 기기를 다룰 수 있도록 여러 가지 분야에 활용되고 있다. 따라서 본 논문에서는 스마트폰의 하드웨어 장치 중 물리적 성질과 환경적 특성을 감지하는 센서에 관해 분석하고자 한다.

본 논문의 구성은 다음과 같다. 제 2장에서는 안드로이드에서 사용 가능한 센서들을 소개하고 센서 매니저를 이용하는 법에 대해 알아본다. 제 3장에서는 센서 값을 해석하는 법에 대해 설명한다. 제 4장에서는 모바일 기기의 센서 목록을 읽어내는 프로그래밍 예를 분석한다. 제 5장에서는 본 논문의 결론을 맺는다.

2. 안드로이드 센서 소개

본 장에서는 안드로이드에서 사용가능한 센서들을 소개하고 센서 매니저를 이용하는 법을 설명한다.

센서 매니저는 안드로이드 기기에서 사용할 수 있는 센서 하드웨어를 관리하는 데 사용된다. 센서 매니저를 이용하려면, (그림 1)의 코드에서 보이는 것처럼 getSystemService를 이용해 센서 매니저 서비스의 레퍼런스를 얻어온다[1][2].

```
String service_name = Context.SENSOR_SERVICE;
SensorManager sensormanager =
(SensorManager)getSystemService(service_name);
```

(그림 1) SensorManager 클래스

안드로이드는 위치 기반 서비스와 마찬가지로 각 기기의 센서 구현을 추상화하고 있다. Sensor 클래스는 각각의 하드웨어 센서가 지닌 속성을 기술하는 데 사용된다. 여기에는 센서의 종류, 이름, 제조사, 그리고 정확도와 범위에 세부 정보 등이 포함된다.

Sensor 클래스에는 센서 객체가 추상화하고 있는 하드웨어 센서의 종류를 나타내기 위한 일련의 상수들이 포함되어 있다. 이 상수들은 Sensor.TYPE_<센서 종류>의 형태로 정의되어 있다. 2.1절에서는 안드로이드가 지원하는 센서들의 종류를 설명한다.

2.1 안드로이드가 지원하는 센서들의 종류

<표 1>은 안드로이드에서 이용 가능한 센서들의 종류이다. 애플리케이션에서 실제로 이용할 수 있는 센서들의 종류는 호스트 기기의 하드웨어에 따라 다르기도 하다.

<표 1> 센서 종류

Sensor.TYPE_ACCELEROMETER	3축 가속도 센서. 세 축에 대한 현재 가속도를 ㎧으로 리턴한다.
Sensor.TYPE_GYROSCOPE	자이로스코프 센서. 세 축에 대한 기기의 현재 방향을 도 단위로 리턴한다.

Sensor.TYPE_LIGHT	주변 광 센서. 주변 조도를 럭스(lux) 단위로 리턴한다 광 센서는 화면 밝기를 동적으로 조절하는 데 흔히 쓰인다.
Sensor.TYPE_MAGNETIC_FIELD	자기장 센서. 세 축에 대한 현재 자기장을 마이크로테슬라 단위로 측정한다.
Sensor.TYPE_ORIENTATION	방향 센서. 세 축에 대한 기기의 현재 방향을 도 단위로 리턴한다.
Sensor.TYPE_PRESSURE	압력 센서. 기기에 가해진 현재 압력을 킬로파스칼 단위로 리턴한다.
Sensor.TYPE_PROXIMITY	근접 센서. 기기와 대상 물체 간의 거리를 미터 단위로 표시한다. 대상 물체를 선택하는 방법과 지원되는 거리는 근접 센서의 하드웨어 구현에 따라 다르다.
Sensor.TYPE_TEMPERATURE	온도 센서. 온도를 섭씨 단위로 리턴한다. 하드웨어 구현에 따라 주변 실온을 리턴할 수도 있고, 기기의 배터리 온도를 리턴할 수도 있으며, 원격 센서 온도를 리턴할 수도 있다.

2.2 센서 찾기

안드로이드 기기는 같은 종류의 센서를 여러 개 가질 수 있다. 같은 종류의 센서들 중 기본 구현을 찾으려면 2.1절에서 설명한 센서들의 종류를 나타내는 상수들 중에서 원하는 것을 가지고 센서 매니저의 getDefaultSensor 메서드를 호출한다.

(그림 2)의 코드는 기본 자이로스코프를 리턴한다. 주어진 종류에 대한 기본 센서가 존재하지 않는 경우에는 null이 리턴된다.

```
Sensor defaultGyroscope =
sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
```

(그림 2) 자이로스코프 리턴

getSensorList를 이용하면, 이용 가능한 센서들 중 주어진 종류에 해당하는 모든 센서들의 리스트를 얻어올 수 있다. (그림 3)의 코드는 이용 가능한 모든 압력 센서 객체들을 얻어온다.

```
List<Sensor> pressureSonsors =
sensormanager.getSensorList(Sensor.TYPE_PRESSURE);
```

(그림 3) getSensorList 메소드

2.3 센서 이용하기

(그림 4)의 코드는 하드웨어 센서의 결과 값을 모니터하기 위한 표준 패턴을 보여준다. SensorEventListener를 정의한 다음, 센서 값을 모니터하려면 onSensorchanged 메서드를, 센서의 정확도 변경을 다루려면 onAccuracyChanged 메서드를 구현한다.

```
final SensorEventListener mySensorListener = new
SensorEventListener() {
public void onSensorChanged(SensorEvent
sensorEvent){ }
public void onAccuracyChanged(Sensor sensor, int
accuracy){ }
};
```

(그림 4) 센서 결과 값 보기

onSensorChanged 메서드의 SensorEvent 매개변수는 센서 이벤트를 기술하는 네 가지 속성을 가지고 있다. 그것들은 sensor, accuracy, values, timestamp 이다.

onAccuracyChanged 메서드를 이용하면, 센서의 정확도에 생기는 변화를 별도로 모니터할 수 있다. 이 두 핸들러 모두에서 accuracy 값은 모니터링되고 있는 센서의 정확도에 대한 피드백을 나타낸다.

센서 이벤트를 받으려면, 센서 매니저에 센서 이벤트 리스너를 등록해야 한다. 이때, 관찰할 센서 객체와 원하는 업데이트 속도를 지정한다. (그림 5)의 코드는 기본 근접 센서에 대한 센서 이벤트 리스너를 보통의 업데이트 속도로 등록한다.

```
Sensor sensor =
sensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY);
sensorManager.registerListener(mySensorEventListener, sensor,
SensorManager.SENSOR_DELAY_NORMAL);
```

(그림 5) 센서 이벤트 리스너의 속도 지정

마지막으로 애플리케이션이 더 이상 센서 값을 업데이트 받을 필요가 없을 때는 센서 이벤트 리스너를 등록 해제해야 한다.

3. 센서 값 해석

본 장에서는 센서의 결과 값을 해석하는 법에 대해 설명한다. onSensorChanged에 전달되는 SensorEvent 객체의 values 값은 모니터링하고 있는 센서의 종류에 따라 크기와 구성이 다르다. <표 2>는 이에 대한 센서 리턴 값을 설명한다.

<표 2> 센서 리턴 값

센서 종류	값 구성
TYPE_ACCELEROMETER	values[0]: X축(Lateral) values[1]: Y축(Longitudinal) values[2]: Z축(Vertical)
TYPE_GYROSCOPE	values[0]: Z축(Azimuth) values[1]: X축(Pitch) values[2]: Y축(Roll)
TYPE_LIGHT	values[0]: 조도
TYPE_MAGNETIC_FIELD	values[0]: X축(Lateral) values[1]: Y축(Longitudinal) values[2]: Z축(Vertical)
TYPE_ORIENTATION	values[0]: Z축(Azimuth) values[1]: X축(Pitch) values[2]: Y축(Roll)
TYPE_PRESSURE	values[0]: 압력
TYPE_PROXIMITY	values[0]: 거리
TYPE_TEMPERATURE	values[0]: 온도

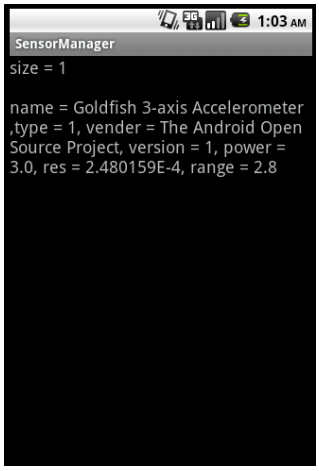
4. 센서관리자 구현

본 절에서는 모바일 기기가 제공하는 센서의 목록을 조사하는 프로그래밍 기법을 소개한다. 외부의 입력을 읽어내는 센서는 하드웨어 장치여서 기기마다 제공하는 센서 목록이 다르다. 센서의 목록을 얻으려면 센서를 관리하는 센서 관리자 객체부터 구해야 한다. (그림 6)의 호출문으로 구할 수 있다[3].

```
(SensorManager)getSystemService(Context.SENSOR_SERVICE);
```

(그림 6) 센서 관리자 객체 구하기

또한 사용 가능한 모든 센서를 한꺼번에 조사하려면 getSensorList 메서드에 TYPE_ALL 인수 전달한다. (그림 7)은 기기가 제공하는 센서의 목록을 보여주는 구현 예이다.



(그림 7) 구현 결과

5. 결론

본 논문에서는 안드로이드에서 사용 가능한 센서들을 알아보고 센서 매니저를 이용하는 법, 센서 값을 해석하는 법에 대해 알아보았다. 또한 이를 바탕으로 기기의 센서 목록을 읽어내는 프로그래밍 예를 분석해 보았다.

최근에도 증강현실이나 3D 게임 등에서 다양하게 센서가 활용되고 있다. 현재 스마트폰 센서는 위치, 방향 등 주변 상황을 감지하는데 주로 활용되고 있지만 향후 신체 정보, 사용자 행동, 감정 등을 인식하는 센서로의 기능 확대가 예상되며, 인터넷과 연결되어 다양한 정보가 연계된 서비스로의 발전이 전망된다. 또한 스마트폰 센서를 기반으로 인간의 감성을 자극할 수 있는 개인에 맞춤화된 지능형 어플리케이션 개발 시장이 확대될 것으로 예상된다[4][5].

이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (2011-0002707)

참고문헌

[1] 리토 마이어, "프로페셔널 안드로이드 개발," 2010, 제이펍
[2] 김상형, "안드로이드 프로그래밍 정복," 2010, 한빛미디어
[3] Kandroid, http://www.kandroid.org
[4] 조선비즈, http://biz.chosun.com/site/data/html_dir/2011/11/25/2011112501122.html
[5] DIGIECO, http://www.digieco.co.kr