Sensor의 활용

목차

- Sensor의 사용 절차
- SensorManager
- SensorListener
- Device Orientation

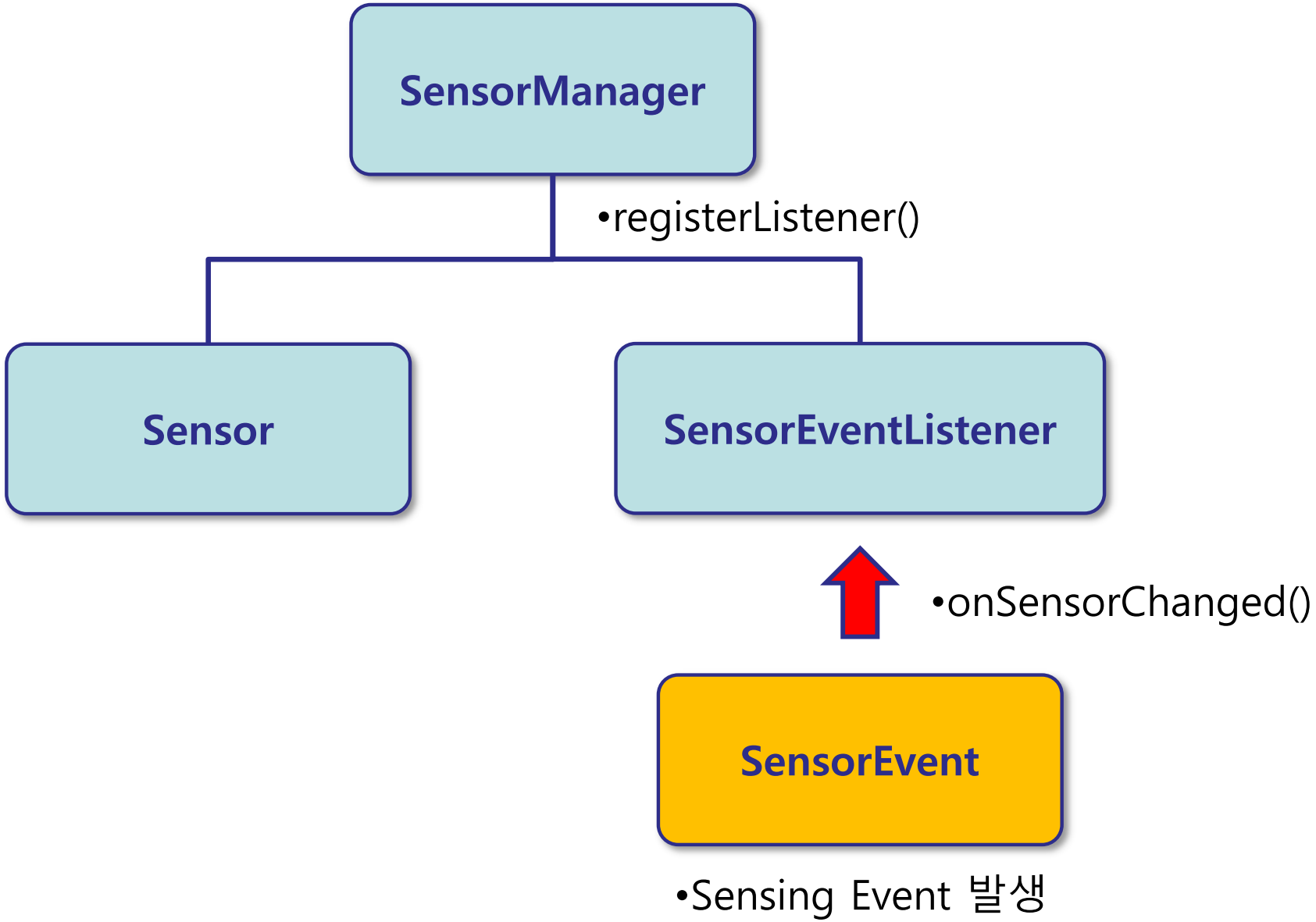
센서 개요

- 움직임, 방향 및 환경조건 측정 센서 내장

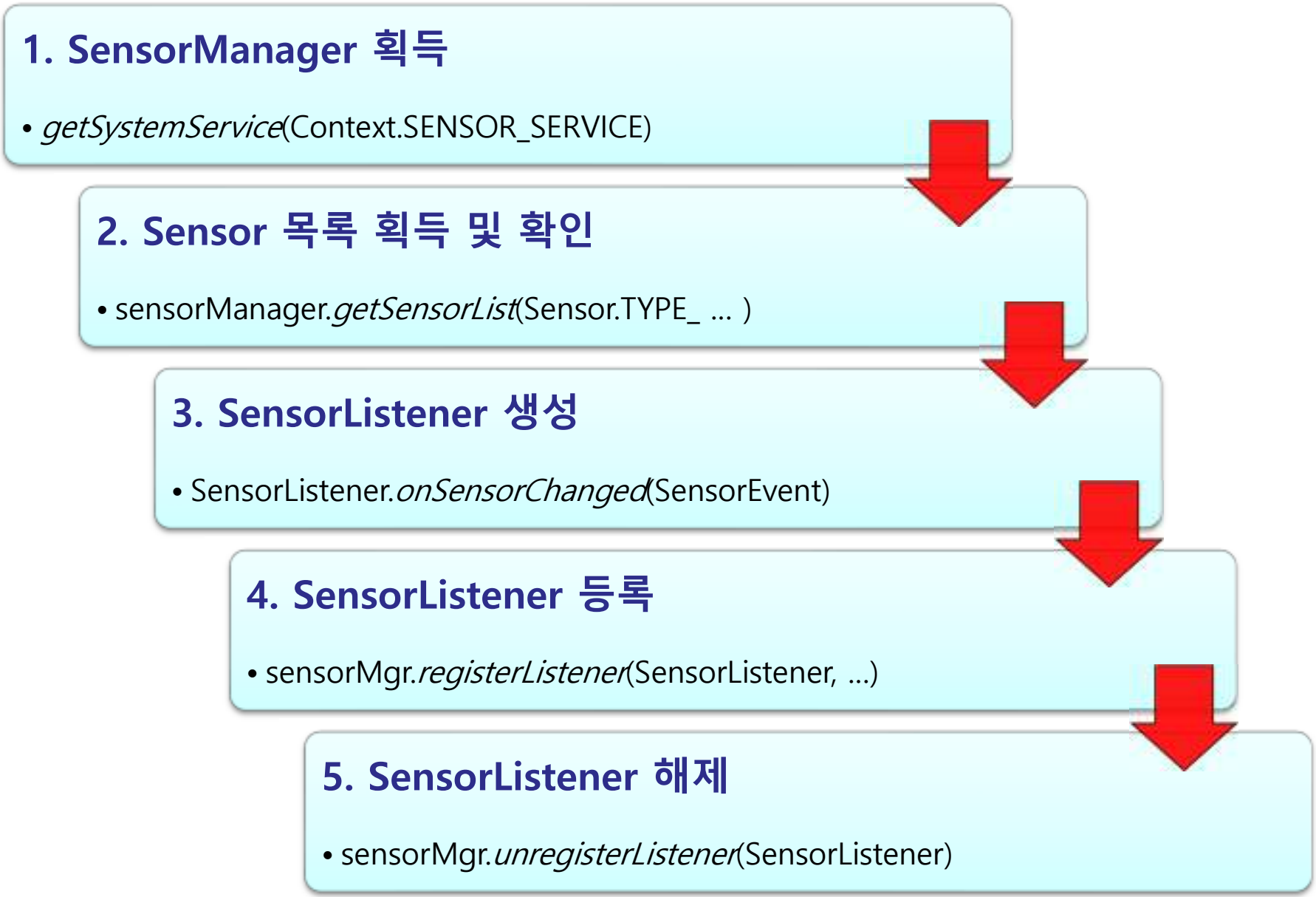
- 센서의 분류
  - 움직임 감지 센서
    - 가속력 및 회전력 측정
    - 가속도계, 중력 센서, 자이로스코프 및 회전 벡터 센서 등
  - 환경 센서
    - 주변의 환경 매개변수 측정
    - 기압계, 광도계, 온도계 등
  - 위치 센서
    - 물리적 위치 측정
    - 방향 센서, 자기 센서 등

- 센서의 유형
  - https://developer.android.com/guide/topics/sensors/sensors_overview#sensors-intro
  - 기기 및 버전에 따라 지원하는 센서가 다름

센서 프레임워크


SensorManager
•registerListener()
Sensor
SensorEventListener
•onSensorChanged()
SensorEvent
•Sensing Event 발생

Sensor의 사용 절차



SensorManager

SensorManager 객체 생성

```
val sensorManager : SensorManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
```

Sensor 목록 획득 및 확인

- Sensor의 타입
- 한 종류의 센서도 여러 개가 있을 수 있음

```
val deviceSensors: List<Sensor> = sensorManager.getSensorList(Sensor.TYPE_ALL)

for (sensor in deviceSensors) {
    mainBinding.tvResult.setText( mainBinding.tvResult.text.toString() + "\n"
        + sensor.toString()
    )
}
```

- Sensor 정보
  - http://developer.android.com/reference/android/hardware/Sensor.html

Google Play 필터링

- 특정 센서가 필요할 경우 AndroidManifest에 지정

```
<uses-feature android:name="android.hardware.sensor.accelerometer"
    android:required="true" />
```

SensorManager 를 활용한 센서 확인

- 센서의 유무 및 종류는 제조업체 별로 상이 → 기기의 센서 내장 유무 및 종류 확인 필요

```
private lateinit var sensorManager: SensorManager
...
sensorManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
if (sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD) != null) {
    // Success! There's a magnetometer.
} else {
    // Failure! No magnetometer.
}
```

```
private lateinit var sensorManager: SensorManager
private var mSensor: Sensor? = null
...
sensorManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
...
if (sensorManager.getDefaultSensor(Sensor.TYPE_GRAVITY) != null) {
    val gravSensors: List<Sensor> = sensorManager.getSensorList(Sensor.TYPE_GRAVITY)
    // Use the version 3 gravity sensor.
    mSensor = gravSensors.firstOrNull { it.vendor.contains("Google LLC") && it.version == 3 }
}
if (mSensor == null) {
    // Use the accelerometer.
    mSensor = if (sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER) != null) {
        sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)
    } else {
        // Sorry, there are no accelerometers on your device.
        // You can't play this game.
        null
    }
}
```

- 예) Sensor의 공급업체가 Google LLC 이고 버전이 3일 경우 사용

- TYPE_GRAVITY 센서가 없을 경우 TYPE_ACCELEROMETER 로 대체

출처: https://developer.android.com/guide/topics/sensors/sensors_overview#sensors-identify

SensorListener 01

SensorListener 생성

```
val sensorEventListener = object: SensorEventListener {
    override fun onSensorChanged(event: SensorEvent?) {
        // Sensor 값 사용
    }
    override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {
        // 정확도 변경 시 동작
    }
}
```

• Sensor 에서 Sensing한 값을 전달받을 때 실행

SensorEvent 객체

- event.accuracy
- event.sensor
- event.timestamp
- event.values

• SENSOR_STATUS_ACCURACY_LOW
• SENSOR_STATUS_ACCURACY_MEDIUM
• SENSOR_STATUS_ACCURACY_HIGH
• SENSOR_STATUS_UNRELIABLE // SensorManager 상수

• Sensor 확인 시간

• sensor 측정값
• float 배열: values
• event?.values?.get(0)
~ event?.values?.get(2)

SensorListener 02

SensorListener 등록

```
val sensorType : Sensor = sensorManager.getDefaultSensor( Sensor.TYPE_LIGHT)

val sensorDelay : Int = SensorManager.SENSOR_DELAY_UI

sensorManager.registerListener(
    sensorEventListener,
    sensorType,
    sensorDelay
    )
```

- 등록 Sensor의 종류 지정

- 센싱 빈도 지정
  - Sensor.SENSOR_DELAY_NORMAL
  - Sensor.SENSOR_DELAY_UI
  - Sensor.SENSOR_DELAY_GAME
  - Sensor.SENSOR_DELAY_FASTEST

SensorListener 해제

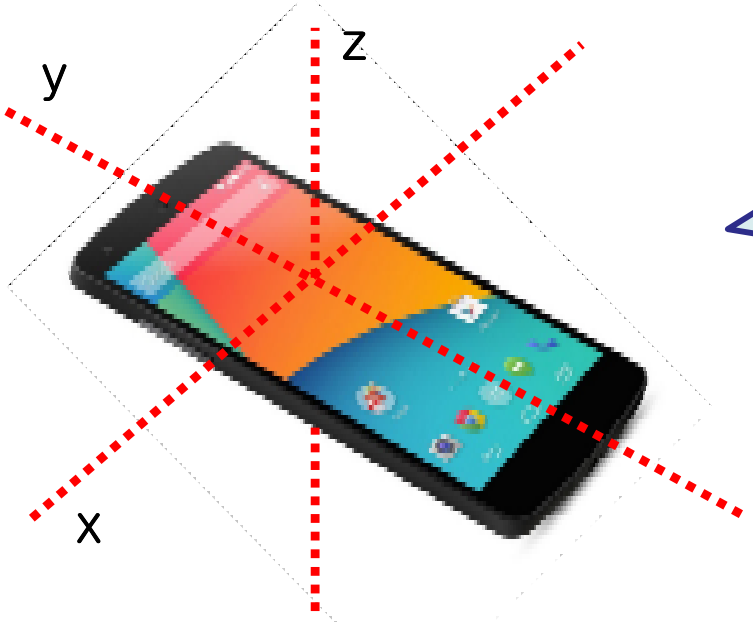
```
override fun onPause() {
    super.onPause()
    sensorManager.unregisterListener(sensorEventListener)
}
```

- sensor 작업은 자원을 많이 소비하므로 반드시 해제 필요
  → onPause() 안에도 구현 권장

센서 활용 예 - Device Orientation 01

Device의 방향

출처: https://developer.android.com/guide/topics/sensors/sensors_position?hl=ko



- Azimuth → event.values[0]
- Pitch → event.values[1]
- Roll → event.values[2]

필요 Sensor 선언 및 등록

- 가속도계(Accelerometer), 지자기계(Magnetometer) 필요

```
lateinit var accelerometer : Sensor
lateinit var magentometer : Sensor

accelerometer = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)
magentometer = sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD)
```

센서 활용 예 - Device Orientation 02

SensorListener 의 onSensorChanged()구현

```
val listener : SensorEventListener = object : SensorEventListener {
    override fun onSensorChanged(event: SensorEvent?) {
        if (event?.sensor?.type == Sensor.TYPE_ACCELEROMETER) {
            System.arraycopy(event.values, 0, mAccmeterReading, 0, mAccmeterReading.size)
        } else if (event?.sensor?.type == Sensor.TYPE_MAGNETIC_FIELD) {
            System.arraycopy(event.values, 0, mMagnetometerReading, 0, mMagnetometerReading.size)
        }

        if (mAccmeterReading.size != 0 && mMagnetometerReading.size != 0) {
            val rotationMatrix = FloatArray(9)

            val isSuccess: Boolean = SensorManager.getRotationMatrix(
                rotationMatrix, null, mAccmeterReading, mMagnetometerReading
            )

            if (isSuccess) {
                var values = FloatArray(3)
                SensorManager.getOrientation(rotationMatrix, values)
                for (i in values.indices) {
                    val degrees: Double = Math.toDegrees(values[i].toDouble())
                    values[i] = degrees.toFloat()
                }
                val azimuth: Float = values[0]
                val pitch: Float = values[1]
                val roll: Float = values[2]
                Log.d(TAG, "azimuth: ${azimuth}, pitch: ${pitch}, roll: ${roll} ")

            }
        }
    }
```

- 각 Sensor 정보를 멤버변수에 보관
- 두 Sensor 의 정보가 모두 확인될 때 수행
- device 회전정보를 토대로 orientation 정보 획득
- 각도 계산
- radian 값을 degree 로 변환

실습

- 붉은색 원이 폰의 기울기에 따라 움직이는 앱 구현
  - Pitch 변경 → y 좌표 변경
  - Roll 변경 → x 좌표 변경
- 화면이 가려질 경우 색상을 변경
  - 빨강 → 파랑, 파랑 → 빨강
- 원이 화면 밖으로 나가지 않도록 제어하는 코드를 추가
  - 가로: 0 ~ canvas.width
  - 세로: 0 ~ canvas.height