Juan Menduiña

Trabajo Práctico N° 5

Ejercicio 1.

Investigar el uso de los diferentes sensores del dispositivo en la documentación oficial de Android: https://developer.android.com/guide/topics/sensors/sensors_environment (Sección: Use the light, pressure, and temperature sensors).

(a) *¿Cuál es la función de la clase SensorManager?*

(b) *¿Cómo se genera una instancia de un Sensor?*

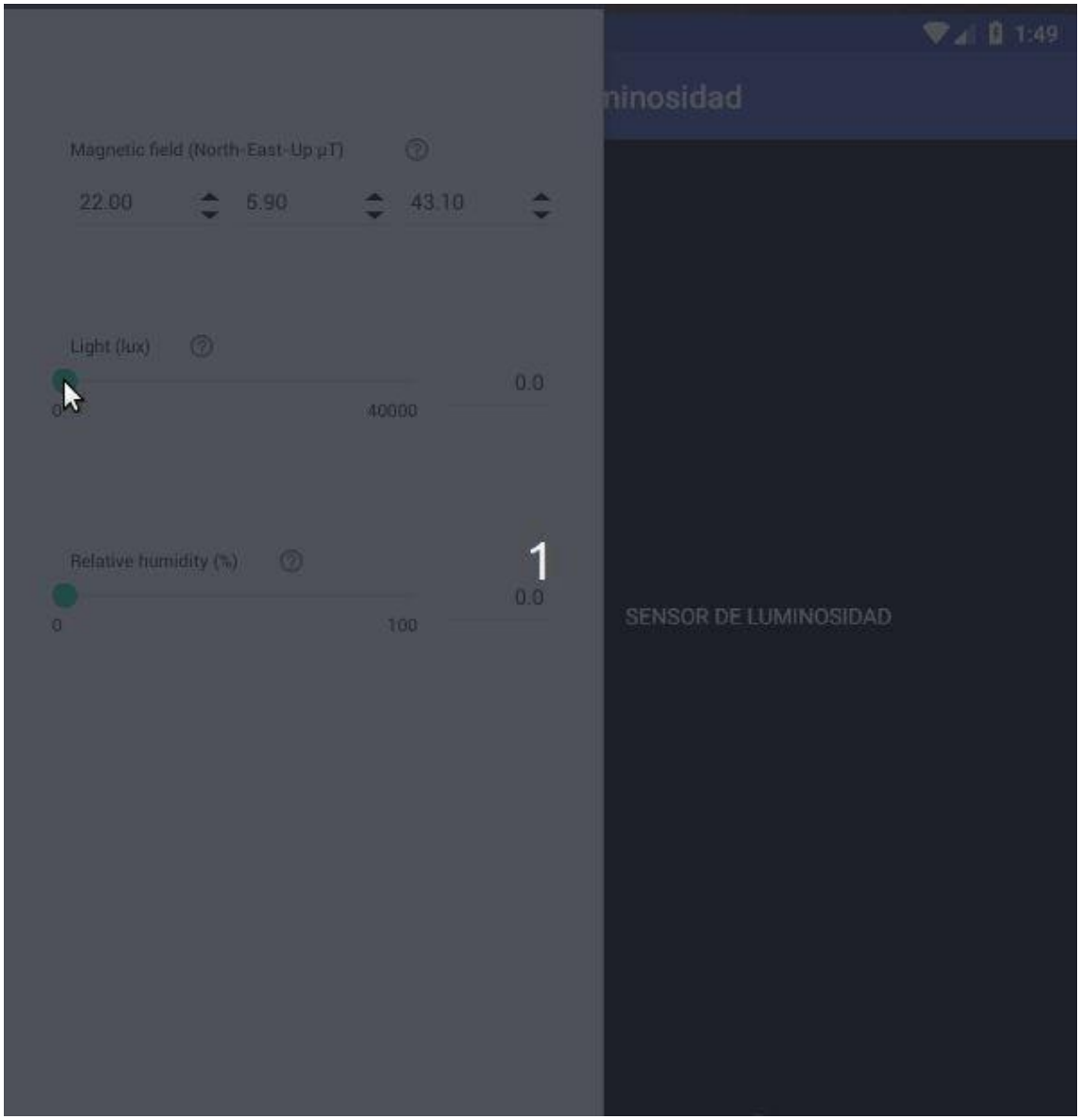
(c) *¿Para qué sirven los métodos registerListener y unregisterListener de la clase Sensor?*

(d) *¿Por qué se utilizan las transiciones de estado onResume y onPause para registrar/eliminar el listener del Sensor?*

<u>Ejercicio 2:</u> Sensor de Luminosidad.

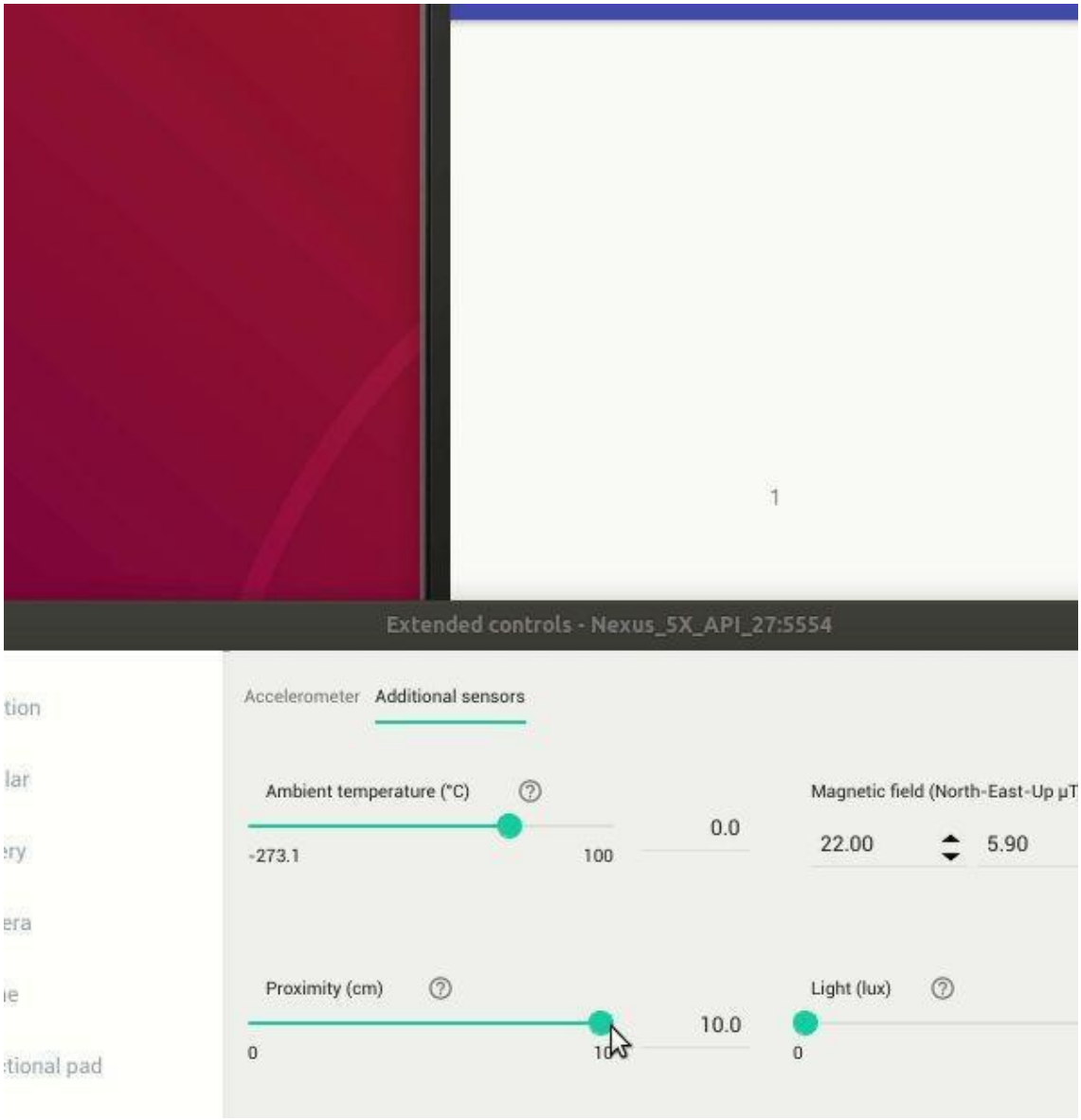
(a) *Implementar una aplicación que muestre, en un TextView, el valor del Sensor en tiempo real.*

(b) *Implementar una aplicación con un TextView que ocupe toda la pantalla y que contenga un texto fijo. El color de fondo y el texto del TextView deberá reaccionar al nivel de luminosidad de manera que se vea el fondo blanco y letras negras en condiciones de mucha luz, mientras que, en condiciones de poca luz, se visualizará el fondo negro con letras blancas.*



Ejercicio 3: Sensor de Proximidad.

Implementar una aplicación que muestre un TextView con un contador numérico. El contador debe incrementarse cada vez que el usuario pase su mano por encima del dispositivo. Nota: Para determinar que el usuario realizó el gesto de pasar su mano por encima del dispositivo, se deberá detectar un estado de "no proximidad", seguido de un estado de "proximidad".



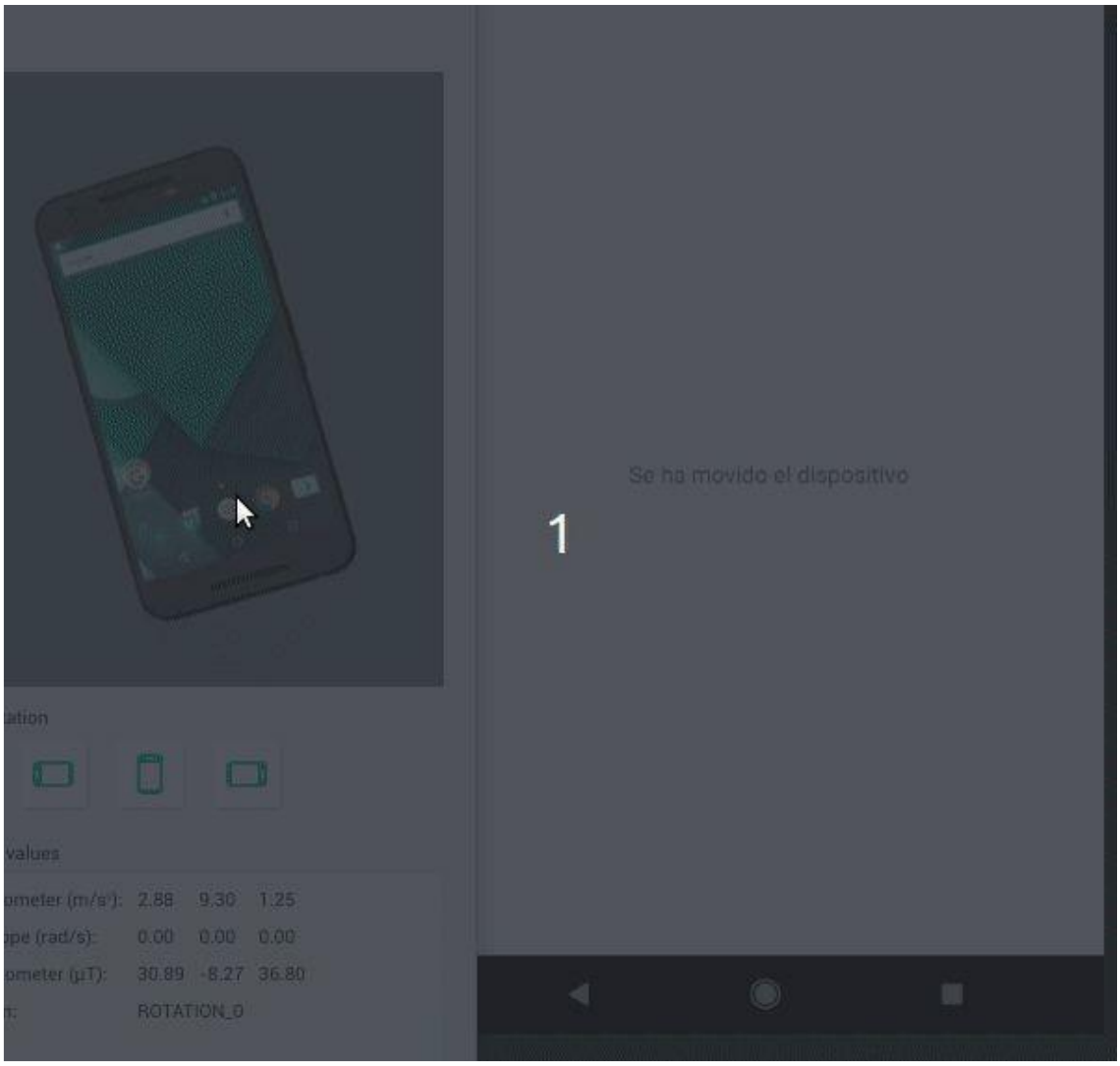
Ejercicio 4: Acelerómetro.

Para estos ejercicios, se deberá utilizar el sensor de Aceleración LINEAL, dado que el mismo no tiene en cuenta la fuerza de gravedad.

(a) *Generar una aplicación que muestre en un TextView los valores de aceleración en los tres ejes X, Y, Z.*

(b) *Cuando el dispositivo está en reposo, ¿los valores son estrictamente cero?*

(c) *Generar una aplicación que sea capaz de detectar el estado de reposo (completamente quieto) del dispositivo y que genere un aviso en un TextView cuando el mismo pierda su estado de reposo (alguien mueva el dispositivo).*


Se ha movido el dispositivo
1
Values
2.88 9.30 1.25
0.00 0.00 0.00
30.89 -8.27 36.80
ROTATION_0

Ejercicio 5: Permisos.

(a) *¿Por qué algunos sensores requieren solicitar permisos en tiempo de ejecución y otros pueden usarse directamente? Investigar los niveles de los permisos en https://developer.android.com/guide/topics/permissions/overview#normal-dangerous.*

(b) *Implementar una aplicación que solicite permisos en tiempo de ejecución para usar la ubicación del usuario. Se pueden utilizar los métodos para consultar y pedir los permisos definidos a continuación. Recordar declarar el permiso tanto en el Manifest como en la Activity.*

```
override fun onStart() {
    super.onStart()
    if (!chequearPermisosLocalizacion()) {
        pedirPermisosLocalizacion()
    } else {
        obtenerLocalizacion()
    }
}

override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<String?>,
    grantResults: IntArray
) {
    super.onRequestPermissionsResult(requestCode, permissions,
grantResults)
    if (requestCode == PERMISSION_GRANTED) {
        if (grantResults.size > 0
            && grantResults[0] ==
PackageManager.PERMISSION_GRANTED
        ) {
            // El usuario concedió el permiso, ya podemos usar
la ubicación
            obtenerLocalizacion()
        }
    }
}

private fun chequearPermisosLocalizacion(): Boolean {
    val permissionState =
        ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION)
    return permissionState == PackageManager.PERMISSION_GRANTED
}

private fun pedirPermisosLocalizacion() {
    ActivityCompat.requestPermissions(
        this@MainActivity,
arrayOf(Manifest.permission.ACCESS_COARSE_LOCATION),
        PERMISSION_GRANTED
    )
}
fun obtenerLocalizacion() {
```

```
    //Acá usamos el método de geolocalización que hayamos
elegido
}
```

Ejercicio 6: Cámara.

(a) *Implementar una aplicación que tome una fotografía usando un intent implícito y la muestre en un ImageView, tal como lo visto en la clase teórica.*

(b) *Al usar la cámara con un intent implícito, no se necesita solicitar permiso al usuario en tiempo de ejecución. ¿Qué diferencia hay si se quiere usar la cámara de manera directa dentro de la app? Investigar en https://developer.android.com/guide/topics/media/camera#manifest.*

(c) *Si se quiere guardar la imagen en la memoria de almacenamiento, ¿alcanza con haber solicitado el permiso de uso de la cámara?*

Ejercicio 7: Posicionamiento.

(a) *Investigar las diferentes estrategias de obtener el posicionamiento del usuario (GPS, WiFi, Celular), sus ventajas y desventajas. Ver https://developer.android.com/guide/topics/location/strategies.*

(b) *Implementar una aplicación que muestre la latitud y longitud del usuario en un TextView, usando las APIs de ubicación de Google Play Services. Para esto, se debe agregar Google Play Services al archivo Gradle del proyecto, ubicado en directorio principal del proyecto: classpath 'com.google.gms:google-services:4.3.15'. Debe quedar de la siguiente forma:*

```
// Top-level build file where you can add configuration options common to all
buildscript {

    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.1.3'
        classpath 'com.google.gms:google-services:3.2.0'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        jcenter()
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
```

Luego, se agrega la dependencia de las APIs de ubicación en el archivo gradle de /app/build.gradle: implementation 'com.google.android.gms:play-services-location:15.0.1'.

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 27
    defaultConfig {
        applicationId "com.example.alfonso.cameraexample"
        minSdkVersion 23
        targetSdkVersion 27
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-ru
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:27.1.1'
    implementation 'com.android.support:support-v4:27.1.1'
    implementation 'com.android.support:support-media-compat:27.1.1'
    implementation 'com.android.support.constraint:constraint-layout:1.1.1'
    implementation 'com.google.android.gms:play-services-location:15.0.1'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
}
```

Una vez hecho esto, ya se incluye la librería de ubicación de Google Play Services en la aplicación. Por lo tanto, se pueden usar las clases que provee la librería para solicitar la ubicación:

```
private var client: FusedLocationProviderClient? = null
private var locationCallback: LocationCallback? = null
```

En el método onCreate, inicializamos estas variables:

```
client = LocationServices.getFusedLocationProviderClient(this)
locationCallback = object :
    LocationCallback() {
    override fun onLocationResult(locationResult: LocationResult) {
        super.onLocationResult(locationResult)
        //usamos locationResult.getLastLocation() para tener
        //la ubicación más reciente
    }
}
```

Y una vez que se tengan los permisos, se le solicita al cliente que envíe la información de localización al callback:

```
client?.requestLocationUpdates(LocationRequest.create(),
    locationCallback as LocationCallback, null
)
```

(c) *Implementar una aplicación que, al iniciar, muestre a qué distancia se encuentra el usuario de la Facultad de Informática. Investigar los métodos que provee la clase Location. La ubicación de la Facultad es -34.9037905, -57.9378442.*