Aplicaciones Telemáticas

2.6.- Geolocalización y sensores

J. E. López Patiño, F. J. Martínez Zaldívar

ETSIT-UPV

Índice

Índice

Permisos en `AndroidManifest.xml`

- Hijos de `<manifest ...>`

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
ó
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

en función de si se desea acceso a localización precisa y/o *gruesa*. Localización fina implica gruesa también en este caso.

Tipos de permisos y versiones de Android

- Tipos de permisos:
  - Normales: sin (mucho) riesgo para la privacidad del usuario o la operación del dispositivo
  - Peligrosos: potencialmente pueden afectar a la privacidad del usuario o la operación normal del dispositivo. El sistema debe pedir permiso.
- Permisos según versiones
  - Versión Android 5.1 (o API 22) o menor en dispositivo o `targetSdkVersion`$<=$ 22: petición de permisos globalmente en la instalación (o en actualización si hay nuevos permisos solicitados). La revocación es por desinstalación.
  - Versión Android 6.0 (o API 23) o mayor en dispositivo y `targetSdkVersion`$>=$ 23: petición de permisos individuales en tiempo de ejecución. Son revocables, luego deben comprobarse en cada ejecución

Procedimiento

- Versión Android 5.1 (o API 22) o menor en dispositivo o `targetSdkVersion`$<=$ 22:
  - Declaración en fichero `AndroidManifest.xml`
- Versión Android 6.0 (o API 23) o mayor en dispositivo y `targetSdkVersion`$>=$ 23:
  - Declaración en fichero `AndroidManifest.xml`
  - Comprobación de concesión de permiso
  - Petición de permiso condicionada
  - Respuesta a la petición

Procedimiento

```
...
/*
 // Llamada para arrancar GPS
        boolean enabled = manejador.isProviderEnabled(LocationManager.GPS_PROVIDER);
        if (!enabled) {
            Intent intent = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
            startActivity(intent);
        }
*/
...
if ( ContextCompat.checkSelfPermission(
          Actividad.this,
          Manifest.permission.ACCESS_FINE_LOCATION )
        != PackageManager.PERMISSION_GRANTED ) {

    // El permiso fue negado...
    if (ActivityCompat.shouldShowRequestPermissionRationale(estaActividad,
            Manifest.permission.ACCESS_FINE_LOCATION)) {

        // Explicación de la necesidad de proporcionar permiso
        // porque probablemente ya se pidió y se denegó

    } else {

        // Petición de permiso sin haber sido negado previamente

        ActivityCompat.requestPermissions(thisActivity,
                new String[]Manifest.permission.READ_CONTACTS,
                MY_PERMISSIONS_REQUEST_READ_CONTACTS);

    }
} else {
  // El permiso estaba ya concedido
}
...
```

Índice

Índice

LocationManager

- Objeto:

```
LocationManager manejador = (LocationManager) getSystemService( LOCATION_SERVICE );
```

- Da acceso a todos los servicios de localización.

Proveedores del servicio de localización

- Proveedores: identificados con `Strings`
  - `LocationManager.GPS_PROVIDER`, (`"gps"`)
    Proporciona combinación de datos de GPS y aGPS (aided-GPS: antenas de telefonía móvil)
  - `LocationManager.NETWORK_PROVIDER`, (`"network"`)
    Proporciona combinación de datos de células de telefonía móvil (incluyendo aGPS) y Wi-Fi
  - `LocationManager.PASSIVE_PROVIDER`, (`"passive"`)
    Proporciona datos pedidos por otras aplicaciones
- Proveedor adicional: `"fused"`. Elige el mejor proveedor en cada momento. Requiere tener instalado `Google Play services`.

Proveedores del servicio de localización (cont.)

- Obtención de todos los proveedores:

```
List<String> proveedores = manejador.getAllProviders();
```

- Propiedades de un proveedor:

```
LocationProvider locationprovider = manejador.getProvider("NOMBRE_PROVEEDOR");
locationprovider[.getAccuracy() | .getName() | .getPowerRequirement() | .hasMoneyCost() ... ]
(http://developer.android.com/reference/android/location/LocationProvider.html)
```

- Mejor proveedor

```
Criteria criterios = new Criteria();
// Ajuste de precision, rumbo, consumo, altura, ... en criterios
String proveedor = manejador.getBestProvider( criterios );
```

Índice

Localización

- Última localización conocida:

```
Location localizacion = manejador.getLastKnownLocation( proveedor );
```

- Método `requestLocationUpdates` del objeto `LocationManager`:

```
manejador.requestLocationUpdates(LocationManager.GPS_PROVIDER, minTime, minDist, locListener);
manejador.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, minTime, minDist, locListener);
        ...
```

`minTime`/`minDist`: mínimo tiempo/distancia entre actualizaciones

- Localización actual:

```
LocationListener locListener = new LocationListener() {
            public void onLocationChanged(Location location) {
               // empleo de coordenadas
               // location.getLongitude(), location.getLatitude(), ...
            }
            public void onProviderDisabled(String provider){
               // proveedor deshabilitado
            }
            public void onProviderEnabled(String provider){
               // proveeder habilitado
            }
            public void onStatusChanged(String provider, int status, Bundle extras){
               // cambio de estado del proveedor
            }
};
```

No acaba...

- La petición de actualización de localización no cesa aunque la aplicación pare: hay que desactivarla explícitamente.
  - Control en sobreescritura de métodos `onResume` y `onPause`:

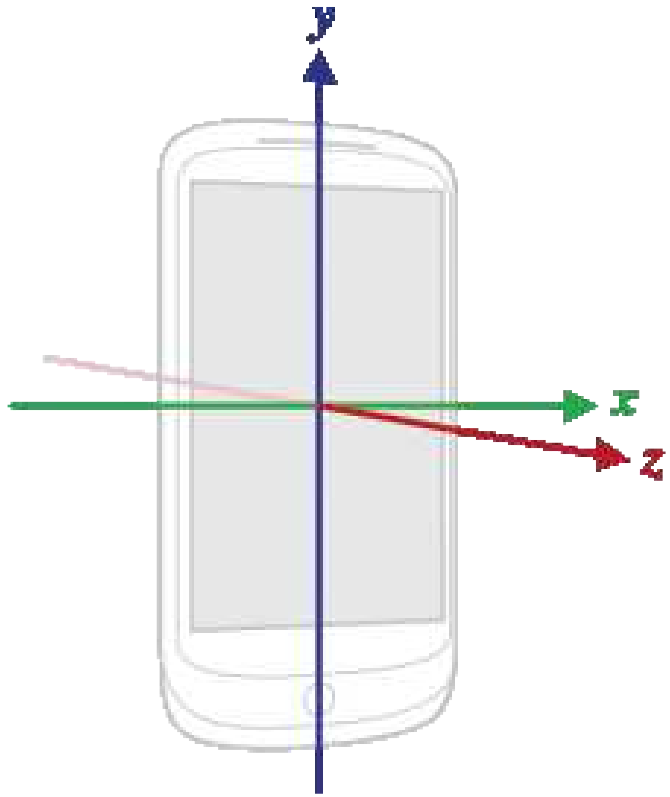
```
@Override protected void onResume( ) {
    super.onResume( );
    manejador.requestLocationUpdates( proveedor, tiempo, distancia, locListener );
}

@Override protected void onPause( ) {
    super.onPause( );
    manejador.removeUpdates( locListener );
}
```

- Ejemplo:
  `https://github.com/AATT-ETSIT/U2T6-Ej-Geolocalizacion.git`

Índice

Tipos

- Acelerómetro
- Sensor magnético
- Giróscopo
- Altímetros
- Sensores de luz
- Termómetros
- Detectores de proximidad
- ...



Cada tipo de sensor puede requerir un tratamiento muy distinto

Tratamiento genérico

- Activiades deben implementar interfaz `SensorEventListener`
- Ello obliga a implementar los métodos:
  - `onAccuracyChanged()`: cambio de precisión
  - `onSensorChanged()`: cambio en la medida
- Y además
  - Se debe *registrar* para escuchar sensores mediante `SensorManager` (servicio accesible mediante llamada `getSystemService` indicando `SENSOR_SERVICE`)
  - Indicando
    - Sensor que desea escuchar
    - Cadencia
- Como en GPS, conviene eliminar del registro cuando no interesa (ahorro energía)
  - Eliminación del registro en `onPause()`
  - Reanudación del registro en `onResume()`
- Ejemplo:
  `https://github.com/AATT-ETSIT/U2T6-Ej1-Sensores.git`