PEC02 - Sensores

Diseñar e implementar un nuevo **sensor** “virtual” que devolverá un valor según la velocidad a la que se desplace el dispositivo que lo utilice.

Trabajo realizado por: **Iván Morata Giménez**

1. Arquitectura de la solución elegida

Para la resolución de la presente práctica, se ha optado por la primera de las opciones planteadas para ello: el uso de un **dispositivo móvil como plataforma sensorial**.

MOVIL/TABLET COMO PLATAFORMA SENSORIAL	*Se trata de utilizar el sensor "acelerómetro" disponible en los diferentes equipos para resolver el nuevo sensor que se debe implementar.*	*Android* *iOS* *Windows* *Entornos multiplataforma (PhoneGap, Xamarin, Flutter ..)*

Se ha empleado un dispositivo Android cuyo acelerómetro se emplea para desarrollar el sensor de velocidad customizado.

2. Diseño del sensor

2.1. Descripción del sensor

Tal como se describe en el guión de prácticas, este sensor realizará un cálculo de la velocidad valiéndose del **acelerómetro** incorporado en el dispositivo Android. En función del valor que muestre en ese momento el sensor, se mostrará el estado del dispositivo:

- 0Km/h - 1Km/h: Parado.
- 1Km/h - 4Km/h: Caminando.
- 4Km/h - 6Km/h: Marchando
- 6Km/h - 12Km/h: Corriendo
- 12Km/h - 25Km/h: Sprint
- 25Km/h - 170Km/h: Veh. Motor Terrestre
- +170Km/h: Veh. Motor Aéreo

Además, será necesario plantear la configuración de las **bandas muertas** existentes al pasar los límites de un estado a otro. Para ello, se hace uso de una variable *deadband* cuyo valor cambiará con las actualizaciones del sensor siempre que se pase el límite de un estado a otro. La variable se utiliza para registrar el Listener del sensor asociado al acelerómetro, que mediante la llamada al método public boolean *registerListener (SensorEventListener listener, Sensor sensor, int samplingPeriodUs)* de la clase SensorManager, cuyo tercer parámetro permite establecer el retardo con el que el sensor ajusta nuevos valores.

La llamada a este método se realiza en el procedimiento principal de la Actividad o pantalla Android, *onCreate()*. Éste se ejecuta continuamente, refrescándose frame a frame, lo que mantiene el Listener asociado al sensor ajustado a la banda muerta correspondiente en cada frame.

2.2. Desarrollo realizado

La aplicación se ha desarrollado en el IDE Android Studio empleando el lenguaje de programación Java para la lógica de la aplicación, y XML para el diseño de una interfaz de usuario básica.

Se ha unificado la lógica de la aplicación en el fichero MainActivity.java, que contiene el método onCreate() que arranca la ejecución de la aplicación, y los métodos asociados que provienen de la interfaz SensorEventListener: *OnSensorChanged(SensorEvent event)* y *onAccuracyChanged(Sensor sensor, int accuracy)*.

Además, se ha implementado la interfaz gráfica en el fichero *activity_main.xml*, realizando una interfaz de usuario sencilla compuesta por las siguientes vistas, ordenadas de arriba a abajo:

- Una vista *imageView* seguida de otra *textView* con el logo de la UNED y el autor de la aplicación.
- Una vista *textView* con el título de la aplicación.
- Una vista *textView* que se actualizará en función de los datos recibidos por el acelerómetro.
- Una vista *textView* con el estado del dispositivo, que irá cambiando en función del valor especificado en la vista anterior según los valores especificados en el apartado 2.1.

2.3. Pruebas realizadas

Dada la dificultad para simular los valores del acelerómetro, se ha empleado un dispositivo físico Google Pixel 6 con el sistema Android, y se ha probado la aplicación en el mismo para asegurar la correcta lectura de los sensores.

Para probar los rangos de valores más altos y difíciles de probar en situaciones reales, y las bandas muertas asociadas, se ha modificado manualmente el valor de la variable *vFinal* según la que se actualizan el estado y la variable *deadband*; posteriormente, se ha probado la aplicación en el emulador de Android incorporado en el entorno de desarrollo.

2.4. Recursos utilizados

Se han empleado los siguientes recursos para el desarrollo de la aplicación:

- Herramientas de desarrollo:
  - IDE Android Studio.
  - Lenguaje de Programación Java.
  - Android SDK 13 (Tiramisu).
  - Dispositivo Android Google Pixel 6.
- Herramientas de documentación:
  - Android para desarrolladores.
  - Stack Overflow.

3. Bibliografía

Se han empleado los siguientes recursos bibliográficos:

- Temas 5-0 a 5-2 de la asignatura de **Computación Ubicua**.
- Android para desarrolladores.
- Stack Overflow.