Interfaces Persona Computador DSIC 2015-16

Segundo entregable

Seguimiento y análisis de actividad física

Descripción general del proyecto y requisitos funcionales

Los deportistas que realizan recorridos al aire libre, como corredores y ciclistas, suelen realizar el seguimiento de sus sesiones de entrenamiento mediante dispositivos GPS y otros sensores adicionales. En concreto, son muy habituales los pulsómetros para medir la frecuencia cardíaca, y sensores de cadencia para detectar la frecuencia de pedaleo o de zancada. Diferentes fabricantes han desarrollado sus propios formatos para almacenar la información obtenida de estos dispositivos. Sin embargo, para facilitar el intercambio de datos entre diferentes aplicaciones se ha desarrollado el estándar gratuito GPX (https://es.wikipedia.org/wiki/GPX, http://www.topografix.com/gpx.asp).

Se pide desarrollar una aplicación de escritorio para evaluar el rendimiento de corredores o ciclistas mediante el análisis de sesiones de entrenamiento y pruebas de competición almacenadas en archivos GPX.

Requisitos funcionales

A) Resumen de una sesión de actividad. La descripción resumida de una sesión debe incluir los siguientes datos
   a. Fecha y Hora de inicio
   b. Duración (horas, minutos y segundos)
   c. Tiempo en movimiento (horas, minutos y segundos)
   d. Distancia total recorrida (Km)
   e. Desnivel acumulado (de subida y de bajada) (en metros)
   f. Velocidad máxima y media (Km/h)
   g. Frecuencia Cardiaca (FC) máxima, mínima y media (pulsaciones por segundo)
   h. Cadencia de pedaleo máxima y media (pedaladas por minuto)

B) Gráficas que muestren
   a. Altura x Distancia (Perfil del recorrido), usando un AreaChart
   b. Velocidad x Distancia, usando un LineChart
   c. FC x Distancia, usando un LineChart
   d. Cadencia x Distancia, usando un LineChart

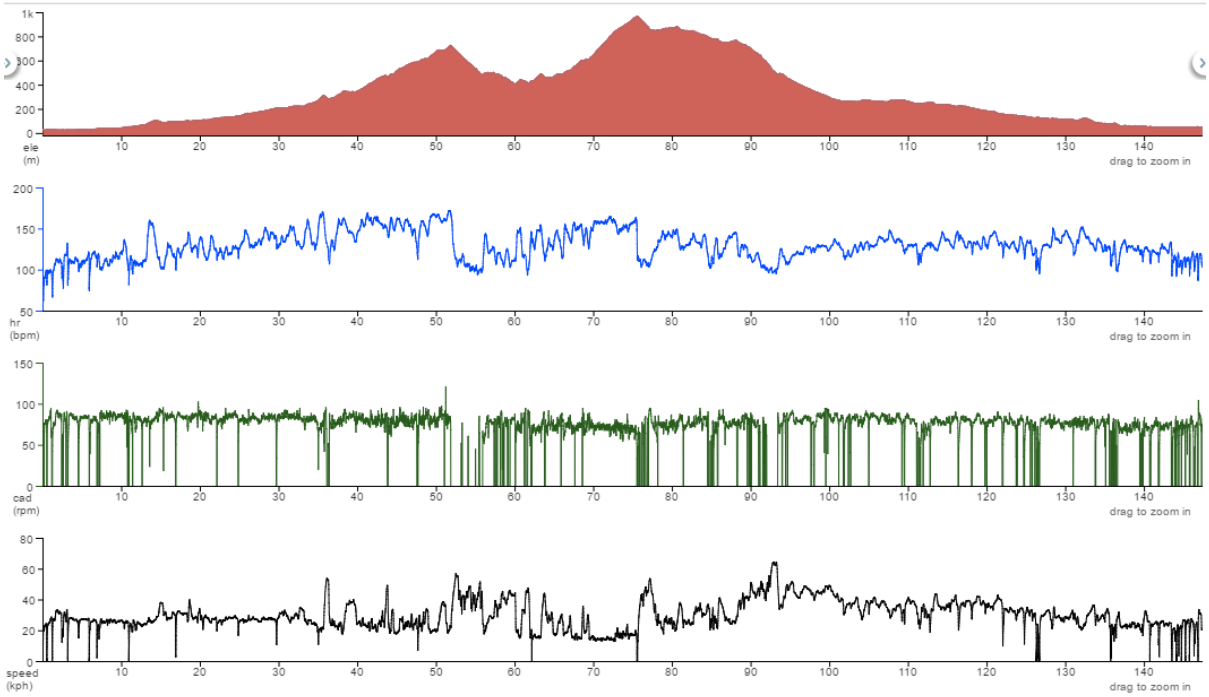

Ilustración 1 Gráficas basadas en la distancia

C) Gráfica de tarta (PieChart) que muestre la distribución de tiempo que pasa el deportista en cada una de las siguientes 5 zonas cardiacas (véase ejemplo en **¡Error! No se encuentra el origen de la referencia.**). Para conseguir estos datos es necesario indicar la FC máxima del deportista, pues las zonas cardíacas dependen de ese valor. Una vez conocidos los valores de FC concretos que separan una zona de otra se calculará el tiempo pasado en cada una de las zonas. Esos tiempos acumulados serán los que usaremos para construir la gráfica.

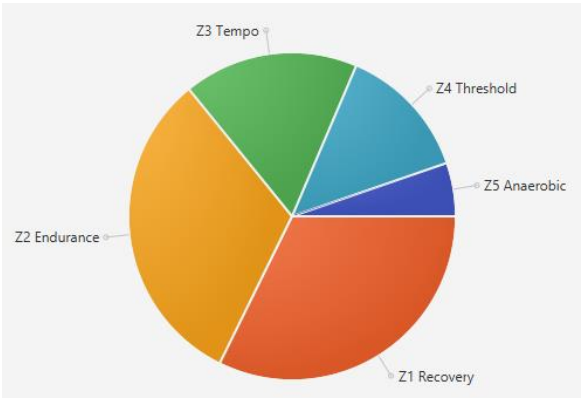

Ilustración 2 Ejemplo de gráfica de esfuerzo cardíaco

Z1 Recuperación	<60% de la FC máxima
Z2 Fondo	60%-70%
Z3 Tempo	70%-80%
Z4 Umbral	80%-90%
Z5 Anaeróbico	>90%-100%

Ampliaciones

Se proponen las siguientes ampliaciones opcionales

A) Permitir que los gráficos muestren la información en base al tiempo en vez de a la distancia. Es decir, que tengan tiempo en el eje X (Altura x Tiempo, Velocidad x Tiempo, etc.). No se trata de crear nuevos gráficos, sino de usar los mismos gráficos pero cambiando la información mostrada. Se usará algún control para indicar si se desea usar Tiempo o Distancia en el eje X.

B) Combinar los datos de FC, cadencia y velocidad en una sola gráfica con varias series de datos. Añadir controles para ocultar/visualizar las diferentes series.

C) Diario de entrenamiento: Ofrecer un mecanismo para cargar múltiples archivos GPX y resumir la actividad realizada durante el último mes o los 3 últimos meses. Por ejemplo, se puede crear una gráfica de barras que muestre las horas de actividad o la distancia recorrida cada día/cada semana/cada mes. Véase la Ilustración 3 a modo de ejemplo.

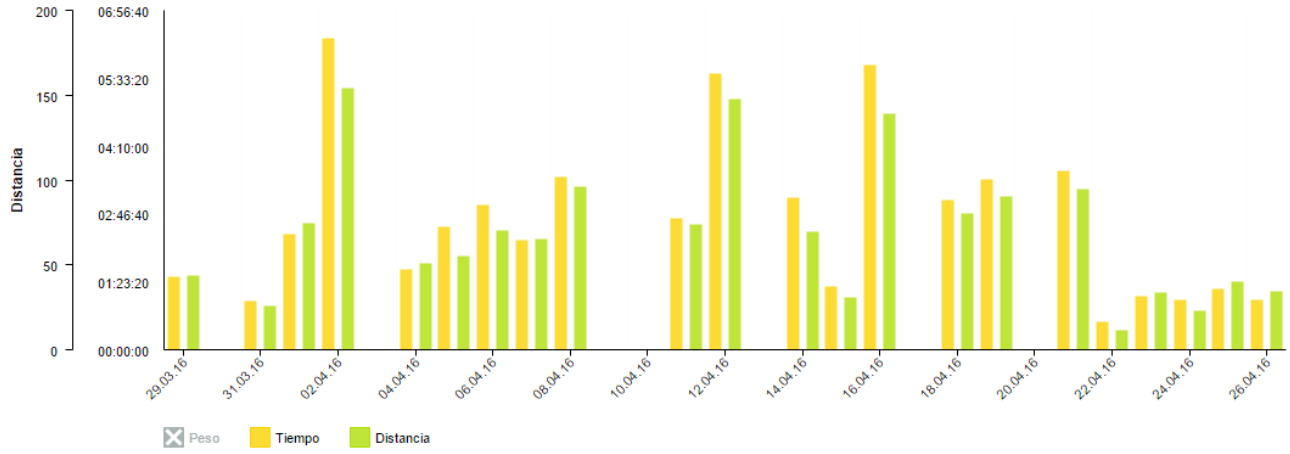

Ilustración 3 Ejemplo de gráfica mostrando actividad diaria a lo largo de un mes

Guía para la realización de la práctica

Para la realización de la práctica se proporciona una librería con toda la funcionalidad necesaria para leer y procesar archivos GPX. Esta librería se encuentra en el archivo **jGPX.jar**. Para utilizar la librería se recomienda añadirla a una carpeta específica dentro del proyecto (se sugiere llamarla "lib" por ser una convención bastante habitual), e importarla al proyecto mediante el comando Properties > Libraries > Add JAR/Folder. De esta forma, la librería se exportará automáticamente al exportar el proyecto a un zip.

Una vez añadida la librería, sin abandonar la ventana de gestión de librerías, podemos seleccionar jGPX y darle al botón Editar, donde podremos indicar la ruta de su documentación, contenida en el archivo **javadoc.zip**.

La Ilustración 4 muestra la pantalla de edición de la librería en el momento de añadir el archivo de documentación (javadoc.zip).

Tanto el archivo de la librería (jGPX.jar), así como su documentación (javadoc.zip), se encuentran en la carpeta "lib" dentro del proyecto **jGPXDemo** proporcionado a través de Poliformat. Para acceder a la documentación desde Netbeans se puede usar la combinación de teclas **Alt + F1** sobre alguna de las clases incluidas en la librería. También se puede pinchar con el botón derecho sobre la librería en el panel de navegación de Netbeans y pinchar en la opción "Show javadoc". Las clases más importantes para realizar la práctica son: **TrackData** y **Chunk**.

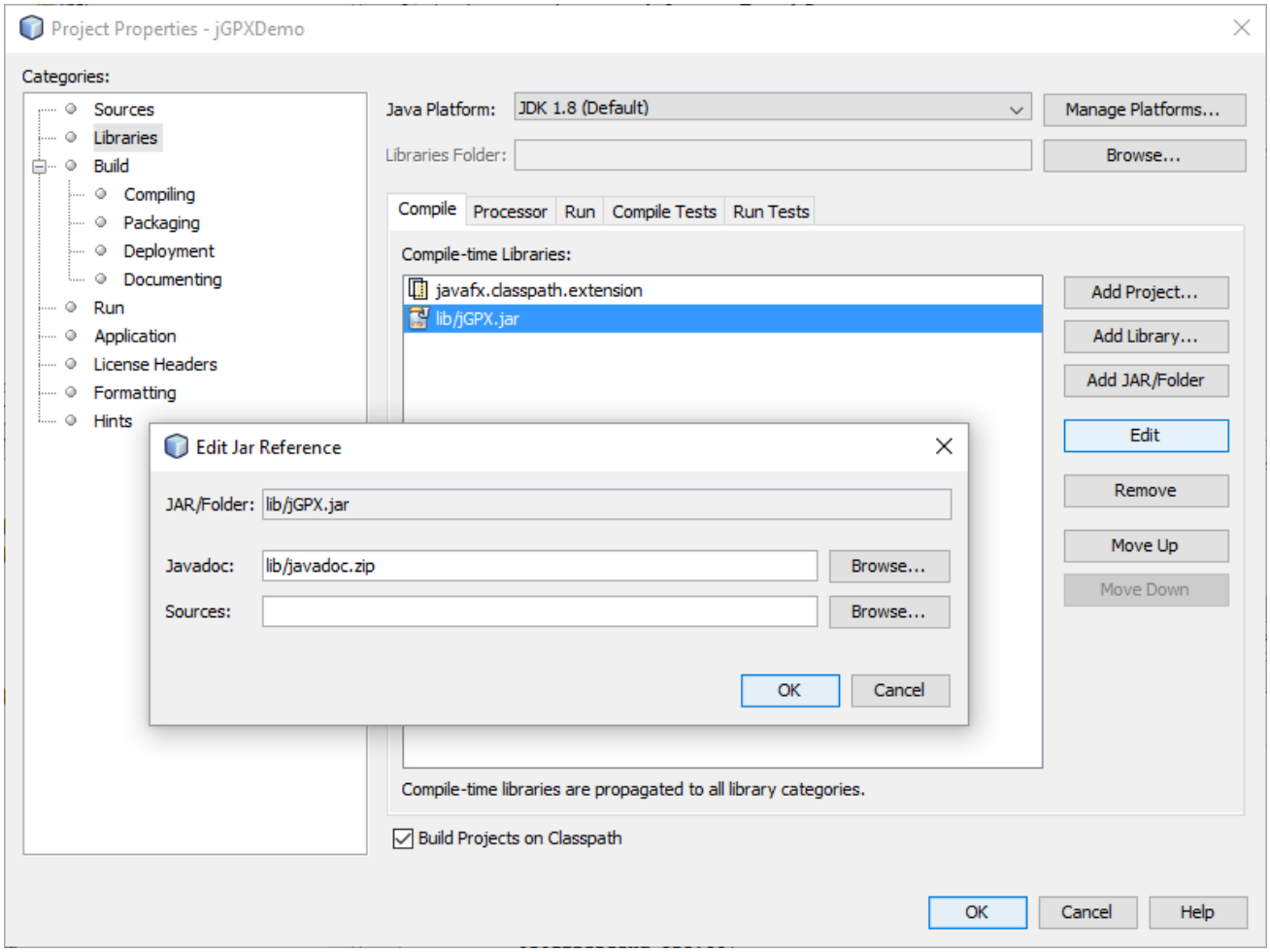

Ilustración 4 Añadiendo la documentación a la librería jGPX

Para la lectura de un archivo GPX se proporcionan unas clases listas para la utilización de la API JAXB. En particular, la clase encargada de representar el contenido de un archivo GPX completo se denomina **GpxType**. A continuación, se incluye el código necesario para cargar un archivo GPX en un objeto GpxType.

```
JAXBContext jaxbContext = new JAXBContext(GpxType.class,
TrackPointExtensionT.class);
Unmarshaller unmarshaller = jaxbContext.createUnmarshaller();
JAXBElement<Object> root = (JAXBElement<Object>)
unmarshaller.unmarshal(file);
GpxType gpx = (GpxType) root.getValue();
```

Una vez obtenido un objeto GpxType ya podemos obtener un objeto de tipo **TrackData**, el cual procesa la información contenida en el GPX y obtiene toda la información derivada necesaria para analizar la actividad, como duración, distancia, velocidad, etc. Fíjate que las 2 últimas líneas, la forma de cargar el XML es un poco distinta a la vista en otras prácticas. Debes hacerlo como se indica en este boletín o no te funcionará el método unmarshal.

```
TrackData trackData = new TrackData(new Track(gpx.getTrk().get(0)));
```

Nota: en teoría un archivo GPX puede contener varios “tracks”, por eso es necesario obtener el primero [get(0)].

Una vez obtenido el objeto **TrackData** podemos usar sus métodos públicos para acceder a toda la información relevante, como por ejemplo:

- `getAverageSpeed()`
- `getMovingTime()`
- `getTotalDuration()`
- Etc.

Para poder generar las gráficas será necesario acceder a la información relativa a cada punto del track. Como hay ciertas informaciones que necesitan de 2 puntos para ser calculadas (velocidad, tiempo, distancia, etc.), en vez de puntos de ruta se trabaja con pares de puntos. Esta información se encapsula en objetos de tipo Chunk. Se obtienen así:

```
ObservableList<Chunk> chunks = trackData.getChunks();
```

Ahora ya podemos obtener información relativa a cada uno de los chunks usando métodos como.

- `getSpeed()`
- `getDistance()`
- `getClimb()`
- Etc.

Normativa de entrega:

- Exporta el proyecto NetBeans en un ZIP (http://politube.upv.es/play.php?vid=68666) y súbelo a la tarea de poliformaT correspondiente. Cada grupo de alumnos hará una sola entrega, incluyendo en el campo de comentarios los nombres de los dos alumnos.

Evaluación:

- Aquellos proyectos que no compilen o que no muestren la pantalla principal al arrancar se calificarán con un cero.
- La interfaz de usuario no se debe quedar bloqueada en ningún momento. Ten en cuenta que procesar ficheros GPX grandes para obtener las series de datos, o procesar muchos ficheros GPX seguidos, puede costar bastante tiempo.
- Se deberán incluir los diálogos de confirmación, errores, etc. que se considere necesario.
- Para evaluar el diseño de la interfaz de la aplicación se tendrán en cuenta las directrices estudiadas en clase de teoría.
- Debe ser posible redimensionar la pantalla principal, cuyos controles se ajustarán adecuadamente para usar el espacio disponible (usa los contenedores vistos en clase).