Curso 2018-2019

Entrega 2. Prácticas IPC

INTERVAL TIMER PARA UNA SALA DE CROSSFIT

IPC

UPV-DSIC

Contenido

1. *Interval Timer* para una sala de crossfit. Descripción general

Para la entrega 2 de prácticas de IPC se tiene que desarrollar una aplicación de escritorio para ser utilizada en una sala de crossfit/fitness, que servirá de ayuda a la realización de las sesiones de entrenamiento.

La funcionalidad principal de la aplicación es la de un cronómetro de intervalos. Esta aplicación irá señalando de manera continua los diferentes intervalos de trabajo y de descanso que se deben de realizar a lo largo de una sesión de entrenamiento. Cuando la sesión está iniciada, el cronómetro indicará a todos los usuarios de la sala o bien el tiempo ejecutado, o bien el tiempo restante del intervalo actual (a vuestra elección).

Si el tiempo es de trabajo, se estará realizando un ejercicio. En el caso de ser tiempo descanso, servirá de preparación para el siguiente ejercicio.

La aplicación debe avisar con señales acústicas y/o visuales los inicios o finales de cada uno de estos intervalos. La interfaz de la aplicación se mostrará en una pantalla, visible por todos los participantes de la sesión y será una réplica de la interfaz de la pantalla del ordenador.

Antes de dar comienzo la sesión de entrenamiento, el entrenador o coach tiene que definir el contenido de esta. Una sesión puede constar de un tiempo previo de calentamiento y, a continuación, se estructura en una serie de ejercicios. Estos ejercicios se realizarán durante un tiempo preestablecido que se conoce como el tiempo de trabajo. En nuestro caso este tiempo será el mismo para todos los ejercicios. Al finalizar cada ejercicio, se dispondrá de un tiempo de descanso entre ejercicio (el mismo durante toda la sesión). Una vez finalizado el circuito con los diferentes ejercicios propuestos, se dispondrá de un tiempo de descanso entre circuitos. Al finalizar este tiempo, se repetirá el circuito anterior. El número de repeticiones del circuito es otro parámetro de la sesión.

Una vez que el coach inicia la sesión el gestor de intervalos, tendrá que informar adecuadamente sobre el tiempo transcurrido o restante en cada uno de estos intervalos. Además, indicará el número del ejercicio actual y el número del circuito actual. Es muy importante avisar con señales sonoras y/o luminosas que el tiempo en cuestión está a punto de expirar.

Durante la sesión, el coach podrá detener el cronómetro en cualquiera de los intervalos: durante la realización de un ejercicio, en el de descanso entre ejercicio o en el descanso entre circuitos. Podrá

reanudar el tiempo donde lo paró o podrá avanzar directamente hasta el siguiente intervalo. También podrá reiniciar la sesión, en este caso se volver a comenzar todo el proceso pero se mantendrá la hora en la que se inició la sesión.

Para tener constancia de los tiempos reales de duración de cada sesión, se debe de almacenar el tiempo transcurrido desde el inicio hasta el final de esta. Una sesión termina cuando finaliza su último intervalo de trabajo.

La aplicación se va a complementar de manera que podamos analizar el histórico de actividad realizado en las sesiones. De esta manera en nuestro modelo vamos a tener en cuenta la existencia de *grupos*, de los cuales vamos a guardar:

- un código
- una descripción
- todas las sesiones que realice.

De cada una de estas *sesiones* se almacenará:

- la fecha
- hora de inicio de la sesión
- la duración real de la sesión
- la descripción del trabajo realizado.

Para describir la sesión, se van a definir sesiones tipo, de tal manera que el coach podrá configurar una sesión con uno de los tipos previamente guardados o crear uno nuevo para la ocasión.

Para mantener la integridad de las sesiones ya realizadas, una *sesión tipo* no se podrá borrar ni modificar. Cada sesión tipo tendrá:

- un código
- un tiempo de calentamiento (cero indica que no hay calentamiento)
- un número de ejercicios
- un tiempo de trabajo de ejercicio
- un tiempo de descanso entre ejercicio
- un número repeticiones de circuito
- un tiempo de descanso entre circuitos.

Todos los tiempos se indicarán en segundos.

Para agilizar el trabajo de configuración de la sesión, un grupo tendrá una sesión tipo por defecto que siempre será el tipo de sesión realizado en su última sesión, en el caso de tener por lo menos una sesión realizada.

2. Funciones a desarrollar

Las funciones que se deben de desarrollar son las de:

- Añadir/modificar Grupo
- Añadir una Sesión Tipo
- Realizar Sesión (detener, reanudar, avanzar)
- Mostrar graficas de las sesiones de un grupo.

En las gráficas de las sesiones de un grupo se mostrará un histórico de las sesiones, que permitirá analizar el trabajo de cada grupo. Así, se mostrará una gráfica de líneas con los tiempos de trabajo de cada sesión, tiempos de descanso de cada sesión y tiempo real de la sesión. Estas tres variables,

así como el grupo deben de ser configurables. Sería deseable que se pudiera escoger el número máximo de sesiones a representar en el eje de las X.

El diseño de la interfaz queda abierto al desarrollador, se recomienda hacer un prototipo en papel previamente y utilizar todas las guías y principios de diseño que se hallan podido presentar en clase de teoría. En caso de duda podéis revisar el diseño con vuestro profesor de prácticas.

Para facilitar el desarrollo de la entrega, vamos a aportar las clases que representan el modelo, así como las funciones necesarias para poder guardar y recuperar la información los grupos y sus sesiones en ficheros XML. La persistencia de los datos en ficheros XML está descrita en los enunciados de prácticas

3. Gestión del tiempo

Existen diferentes manera de gestionar el tiempo en java, en nuestro caso se recomienda hacer uso de las nociones de procesamiento en segundo plano de javafx que hemos visto en el laboratorio. Para obtener valores precisos del reloj del sistema podemos utilizar la función:

```
long nowTime = System.currentTimeMillis();
```

De esta manera podremos obtener con precisión los milisegundos transcurridos entre dos instantes restando el tiempo obtenido en cada uno de ellos. Para trabajas con intervalos tenemos la clase Duration:

```
Duration duration = Duration.ofMillis(totalTime);
final Long minutos = duration.toMinutes();
final Long segundos = duration.minusMinutes(minutos).getSeconds();
```

En este código se muestra como obtener los minutos y segundos contenidos en un intervalo de tiempo expresado en milisegundos.

4. Clases del modelo

Las clases que componen el modelo de esta aplicación son: Gym, Grupo, SesionTipo, Sesion. Estas clases ya están incluidas en una librería que facilitamos. Para poder utilizar estas clases será necesario incluir la librería en el proyecto como se indica posteriormente.

Clase Gym

```
public class Gym {

    private ArrayList<Grupo> grupos = new ArrayList<>();
    private ArrayList<SesionTipo> tiposSesion;

    public Gym() {...2 lines }
```

Clase Grupo

```
public class Grupo {

    private String codigo;
    private String descripcion;
    private ArrayList<Sesion> sesiones;
    private SesionTipo defaultTipoSesion;
```

Clase SesionTipo

```
public class SesionTipo {
    private int t_calentamiento;

    private int num_ejercicios;
    private int t_ejercicio;
    private int d_ejercicio;

    private int num_circuitos;
    private int d_circuito;
```

Clase Sesion

```
public class Sesion {

    private LocalDateTime fecha;
    private SesionTipo tipo;
    private Duration duracion;
```

5. Clases para almacenar y recuperar los datos del modelo en XML

Para almacenar y recuperar los datos de la aplicación se ha creado la clase AccesoBD que implementa el patrón singleton. De esta manera siempre que invoquemos al método estático getInstance() nos retornara una única instancia de la clase AccesoBD.

La clase implementa los métodos:

```
public Gym getGym() {...18 lines }
```

Este método retorna un objeto del tipo Gym inicializado con los datos almacenados en el fichero XML. Es responsabilidad de la aplicación utilizar este objeto para añadir Grupos, Sesiones y SesionTipo.

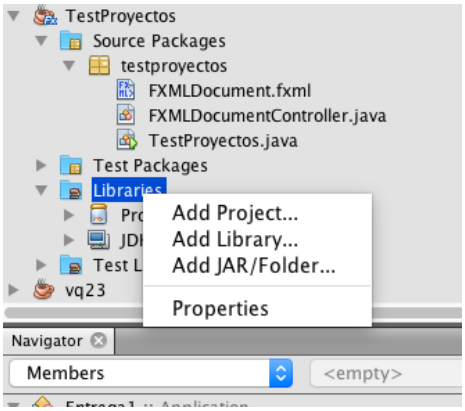
```
public void salvar() {...18 lines }
```

Al invocar a este método se guardaran en el fichero XML todos los cambios realizados sobre el objeto Gym retornado al invocar al método getGym();

6. Como añadir un jar a un proyecto Java

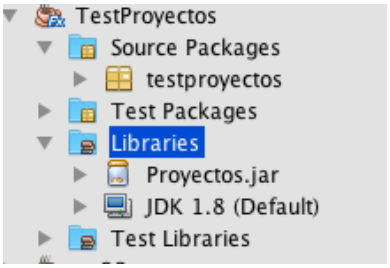
Para añadir una librería o un fichero de clases JAR a un proyecto de java recomendamos en primer lugar dejar la librería dentro del directorio del proyecto, de esta manera las referencias a la librería serán mediante un path relativo y se podrá ejecutar la aplicación sin errores. Para ello recomendamos crear una carpeta en la raíz del proyecto que se puede llamar por ejemplo "lib". Deberemos de copiar dentro de esta carpeta la librería que en nuestro caso se llama AccesoBD.jar. Además, la librería se complementa con el javadoc y las clases para que podáis acceder al código y a la documentación, copiad también los ficheros AccesoBD -source.zip y AccesoBD-javadoc.zip.

Para añadir una librería al proyecto lo podremos hacer desde las propiedades del proyecto o simplemente desde el explorador del proyecto en la carpeta *Libraries* Si nos situamos sobre esta y pulsamos el botón derecho nos aparece la opción de añadir la librería



Ahora pulsamos sobre *Add JAR/Folder* y escogemos el fichero *Proyectos.jar* que se encuentra en lib.

El resultado final será este:



7. Instrucciones de entrega

Exporta el proyecto NetBeans en un ZIP (https://media.upv.es/player/?id=a0bfd620-021e-11e6-851a-656f7e06a374) 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.

8. Evaluación:

- Aquellos proyectos que no compilen o que no muestren la pantalla principal al arrancar se calificarán con un cero.
- 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.
- El diseño y las dimensiones de la interfaz deben de ser adecuadas al entorno de utilización de esta aplicación