

15 de noviembre de 2024

Examen parcial - 10 puntos (Duración: 2,5h)

Antes de comenzar a resolver el examen debes seguir los pasos indicados en la hoja de instrucciones

El caso que vamos a trabajar en este examen está relacionado con las actividades de un gimnasio. Queremos crear dos diseños para el horario semanal de las actividades (Tareas 1, 2 y 3) y hacer un juego, inspirado en una máquina recreativa de tipo Slot Machine Game, cuyo premio es un abono mensual (Tarea 4). El dominio de la aplicación lo componen 2 clases (**Actividad** y **Sesión**) y 2 enumeraciones (**TipoActividad** y **DiaSemana**).

A continuación, se describen las 4 tareas que componen el examen. Para cada una, se indica su puntuación (sobre 10 puntos), la clase en la que debes añadir el código y una estimación del número de líneas de código necesarias. El código base está organizado en 3 paquetes y en cada uno encontrarás una clase con un método **main()** para ejecutar tu código y ver el resultado: **MainLayout**, (Tarea 1), **MainJTable** (Tareas 2 y 3) y **SlotMachineGame** (Tarea 4).

Por último, dentro de la carpeta **resources/doc** encontrarás imágenes y un vídeo que te ayudarán a ver con mayor detalle el objetivo de las diferentes tareas.

Tarea 1: Layout [2 puntos] [HorarioGimnasioLayout] [70 líneas]

El objetivo de esta tarea es diseñar una vista del horario semanal de las actividades del gimnasio tal y como la que se muestra en la imagen de la página siguiente. Debes analizar la imagen y replicar el diseño de la misma dentro de la clase **HorarioGimnasioLayout**.

Para resolver esta tarea debes procesar la lista de actividades **List<Actividad>** que se crea en la clase **MainLayout** y **NO puedes utilizar ningún componente JTable**. Además, para crear el diseño propuesto, sólo puedes utilizar componentes básicos y los siguientes layouts: **FlowLayout**, **BorderLayout** y **GridLayout**.

Fíjate en que cada instancia de la enumeración **TipoActividad** tiene un atributo de tipo **Color**.

		LUNES	MARTES	MIERCOLES	JUEVES	VIERNES
09:00	YOGA			YOGA		
10:00		PILATES			PILATES	
11:00		ZUMBA			ZUMBA	
12:00	BOXEO					BOXEO
13:00	SPINNING			SPINNING		SPINNING

Tarea 2: Manejo y renderizado de JTable

El objetivo de esta tarea es crear un nuevo diseño para el horario semanal de las actividades del gimnasio. Este nuevo diseño está basado en un componente **JTable**. Debes conseguir que el aspecto de la tabla de actividades sea igual al que se muestra en la siguiente imagen.



Tarea 2.1: Modelo de datos [1,5 puntos] [HorarioGimnasioModel] [60 líneas]

La versión actual de la tabla de actividades se inicializa con datos de tipo String pero queremos inicializar la tabla con la lista de actividades `List<Actividad>` que se crean en la clase `MainJTable`.

El objetivo de esta tarea es crear un modelo de datos personalizado para utilizarlo en la tabla de actividades. Para ello, debes crear una nueva clase llamada `HorarioGimnasioModel`. Además de las particularidades que tienes que identificar directamente al ver la imagen, debes tener en cuenta estas cuestiones adicionales:

- El constructor de la clase recibe como parámetro una lista de actividades `List<Actividad>`.
- Ninguna celda de la tabla es editable.
- Debes pensar muy bien cuál debe ser el retorno del método `getValueAt()` para cada uno de los valores que el modelo debe proporcionar al `JTable`.

Una vez creado el nuevo modelo de datos, debes utilizarlo en la clase `HorarioGimnasioJTable` para inicializar la tabla de actividades `tablaActividades` de manera adecuada.

Tarea 2.2: Diseño y renderizado [1,5 puntos] [HorarioGimnasioJTable] [60 líneas]

El objetivo de esta tarea es lograr el diseño de la tabla de actividades que puedes ver en la imagen de arriba. Fíjate especialmente en lo siguiente:

- Debes impedir que las columnas de la tabla puedan reorganizarse.
- Los textos de las horas y los días tienen tipografía Arial de 18 puntos y negrita.
- La tabla no tiene ninguna línea de cuadrícula.
- Las imágenes para representar cada tipo de actividad están en la carpeta `resources/images/`.
- Se debe mostrar el nombre de la actividad cuando el usuario posiciona el ratón sobre una celda que contenga la información de una actividad.
- Recuerda que cada `TipoActividad` tiene un atributo de tipo `Color`.

Tarea 3: Evento de teclado [1 punto] [HorarioGimnasioJTable] [25 líneas]

La ventana `HorarioGimnasioJTable` no puede cerrarse al pulsar la X de la barra de título y por lo tanto, la aplicación no puede detenerse. El objetivo de esta tarea es permitir cerrar la aplicación. Para lograrlo, debes implementar la funcionalidad necesaria para capturar la pulsación de la combinación de teclas `CTRL + E` y mostrar un cuadro de diálogo de confirmación tal y como se muestra en la imagen que tienes arriba. Si se pulsa `Si`, la aplicación se cierra; de lo contrario, no se hace nada. Para generar el cuadro de diálogo utiliza un diálogo estándar de los que la clase `JOptionPane`.

Tarea 4: Hilos [4 puntos] [SlotMachineGame] [30 líneas]

El objetivo de esta tarea es implementar un juego de azar tipo Slot Machine cuyo premio es un abono mensual de nuestro gimnasio. Para ello debes dotar de dinamismo a 3 JLabel que muestran las imágenes de los diferentes tipos de actividades que ofrece el gimnasio en una secuencia continua y aleatoria.

Debes diseñar una serie de hilos para cambiar la imagen de cada JLabel por otra (seleccionada de forma aleatoria de entre todas las imágenes de las actividades) cada 100 milisegundos en una secuencia continua y totalmente independiente para cada JLabel.

La secuencia se iniciará al pulsar el botón **Iniciar** y se detendrá completamente al pulsar el botón **Parar**. Una vez detenida la secuencia, se comprobará si las 3 imágenes son iguales y se ha obtenido el premio. Se podrá reiniciar la secuencia tantas veces como se quiera, hasta que se haya obtenido el premio. Desde ese momento, hay que mostrar un mensaje indicando que no se puede seguir jugando porque ya se ha obtenido el premio.

En la carpeta **resources/doc** tienes un vídeo que muestra el comportamiento que debes conseguir.