PROGRAMACIÓN AVANZADA

PRÁCTICA 3: PROGRAMACIÓN DE INTERFACES GRÁFICAS

1. Objetivo

Hemos llegado a la última práctica.

En la primera práctica has utilizado POO para programar una aplicación de facturación para una empresa proveedora de telefonía móvil. En la segunda práctica has utilizado los patrones de diseño Factory Method y Decorator para ampliar la funcionalidad de tu aplicación.

En esta práctica final vas a programar una interfaz gráfica de usuario para tu aplicación de facturación.

El diseño de la interfaz lo decides tú, la única directriz es que la aplicación final sea lo más «usable» posible.

2. Enunciado

A partir del código de la primera y segunda practicas, vas a crear una interfaz gráfica de usuario que permita realizar todas las tareas que ahora presentas como menús en consola.

Para ello vas a utilizar el (meta)patrón de diseño Modelo/Vista/Controlador (MVC). Recuerda que según este patrón de diseño, las clases de tu aplicación sólo pueden jugar uno de los tres roles, una clase puede formar parte sólo del Modelo, sólo de la Vista, o sólo del Controlador. Un buen modo de empezar es creando paquetes con esos nombres, aunque esto, evidentemente, no te garantiza seguir el patrón MVC.

También debes decidir cual es el aspecto de la interfaz gráfica de usuario. Puedes construirla con los componentes que hemos visto en clase de teoría, o puedes, si te apetece y te animas, incluir nuevos componentes que no hemos visto en clase. En este último caso revisa los enlaces que tienes en la sección **Fuentes de información**.

Metodología

Esta práctica la vas a desarrollar en tres sesiones. Como en las prácticas anteriores, en cada una de las sesiones te vamos a proponer que te concentres en un aspecto particular, pero, de nuevo, siéntete libre de marcar tú tus propios ritmos.

Sesión 1

En esta primera sesión vas a estructurar tu aplicación para que siga el patrón de diseño Modelo/Vista/Controlador, y vas a construir una primera versión de la interfaz gráfica de usuario.

Estructura tus clases según los roles que van a tomar. Para ello puedes ayudarte creando tres paquetes cuyos nombres sean: modelo, vista, controlador. Una clase sólo pertenecerá a uno de estos paquetes. Una vez que hayas realizado estos cambios, cuando tengas creada la Vista Swing, te resultará muy sencillo «desconectar» la vista basada en consola/teclado para conectar la nueva Vista basada en Swing.

Diseña, en una hoja de papel por ejemplo, el aspecto de la interfaz. Presta atención a que todas las opciones que se pueden llevar a cabo mediante las opciones de menú de la versión consola, se puedan llevar a cabo mediante la interfaz gráfica.

Crea también los flujos de la aplicación, ¿qué ocurre con cada interacción del usuario?

Decide cuáles van a ser los componentes de la interfaz gráfica que vas a utilizar, para cada uno de ellos, cuáles son los eventos que te interesa escuchar y cómo programarlos.

Decide cual es la interface pública (métodos) que va a necesitar la Vista del Controlador. Por ejemplo, si un botón se ha pulsado en la Vista, de qué modo se va a informar al Controlador, si es que hay que informarle. Decide el método de notificación push o pull.

Programa la interfaz gráfica con los componentes Swing. Programa también todos los escuchadores que necesites para detectar la interacción del usuario. Enlaza los escuchadores con el Controlador. Para ello, crea un controlador sencillo que sólo muestre texto por consola para indicar que le ha llegado la notificación.

Sesión 2

En este segunda sesión vas a programar el Controlador.

En la sesión anterior, has decidido cual es la interface pública del Controlador y has programado uno para comprobar que le llegan las notificaciones de la Vista.

Ahora vas a programar el comportamiento real del Controlador. Por ejemplo, cuando un usuario pulsa sobre un botón Añadir Cliente, la Vista informa al Controlador de lo sucedido y éste debe obtener los datos del nuevo cliente y añadirlos al modelo. Cómo debe obtener los datos el Controlador depende de si has elegido método push o pull de notificación.

Decide cual es la interface pública (métodos) que debe tener la Vista para que el Controlador la pueda utilizar.

El Controlador es el responsable de actualizar el Modelo. Tu Modelo ya lo tienes construido, es parte de la aplicación que ya tienes programada (los datos y cómo actualizarlos). También tienes que programar cómo interacciona el Controlador con el Modelo, para ello, de nuevo, define la interface pública del Modelo para que el Controlador la use.

Sesión 3

Finalmente, vas a programar la actualización de la Vista antes los cambios en el Modelo.

Ya sabes, que si el Modelo cambia, notifica a la Vista de los cambios para que ésta tenga la oportunidad de actualizarse. Por ejemplo, si la Vista está mostrando el número actual de clientes en la cartera de la empresa de telefonía, y se añade un nuevo cliente al Modelo, éste debe notificar el cambio (un cliente más) a la Vista para que ésta se actualice, si lo considera oportuno.

Con esta última conexión, tendrás tu aplicación de facturación con interfaz de usuario programada con el patrón MVC, y perfectamente funcional.

Existen frameworks específicos para probar la interfaz gráfica de usuario, pero no las vamos a utilizar. Haz las pruebas que consideres necesarias para probar el funcionamiento de tu aplicación.

3. Fuentes de información

- Big Java Capítulo 17.
- Head first Java Capítulo 12.

- Head first design Patterns Capítulos 12.
- Patrones de diseño Capítulos 3 y 5.
- Desarrollo de proyectos informáticos con tecnología Java Capítulo 11.
- Swing en la página web de Oracle.