Comunicación Persona Máquina

Universidad de Oviedo

Grado en Ingeniería Informática del Software

Segundo curso

Documento explicativo del módulo de prácticas

Iván Álvarez López – 71741444M

- **Índice:**
  - **Introducción**: Página 3.
  - **Prototipo**: Página 3.
  - **Interfaz**: Páginas 4 a 7.
  - **Lógica**: Página 7.

• Introducción

En este módulo de prácticas se nos ha propuesto realizar una aplicación visual en la que un usuario cualquiera organice una fiesta, escogiendo artículos varios y locales para su posterior compra.

La aplicación se encuentra **internacionalizada** a los idiomas en los que tengo fluidez, que son el **castellano**, el **inglés** y el **gallego**, por lo tanto, aplico la opción para subir nota. Una vez internacionalizado, es relativamente sencillo internacionalizar para otros idiomas (digo relativamente por cosas como el tipo de divisa que se use, o el formato horario, detalles susceptibles de ser pasados por alto).

• Prototipo

El prototipo inicial que hicimos en seminarios fue bastante menos inmersivo que el final, entraba mal por los ojos, por lo que decidí cambiar de primeras el *Look&Feel* de la aplicación, además de incluirle imágenes personalizadas como icono y para la pantalla de inicio de sesión.

Por simplicidad, decidí combinar las pantallas de registro y de inicio de sesión en una, así reducimos el engorro de hacer que el usuario esté atento a varias ventanas que se le puedan estar abriendo.

Me pareció absurdo poner el número de asistentes en una ventana a parte, por lo que lo incrusté directamente en la pantalla principal. También retiré el cuadro de búsqueda y sustituí su función por un *JTabbedPane* autogenerado, cuyos paneles fuesen los tipos de los productos.

Para mejor manejo del registro de cuentas, decidí retirar la opción de registrarse o de iniciar sesión en la propia pantalla principal.

El panel de productos lo decidí establecer mediante tablas generadas automáticamente, que mostrasen todos los atributos del producto directamente en ellas.

- # Interfaz

Se nos propone crear un sistema de registro y acceso al programa mediante el registro de unas credenciales típicas de nombre de usuario y contraseña, lo cual solvento con una **ventana inicial** de *login*, la cual viene conformada por la clase de interfaz gráfica de usuario llamada ***WelcomeWindow***, la cual ha de ser la que se arranque al iniciar el programa.

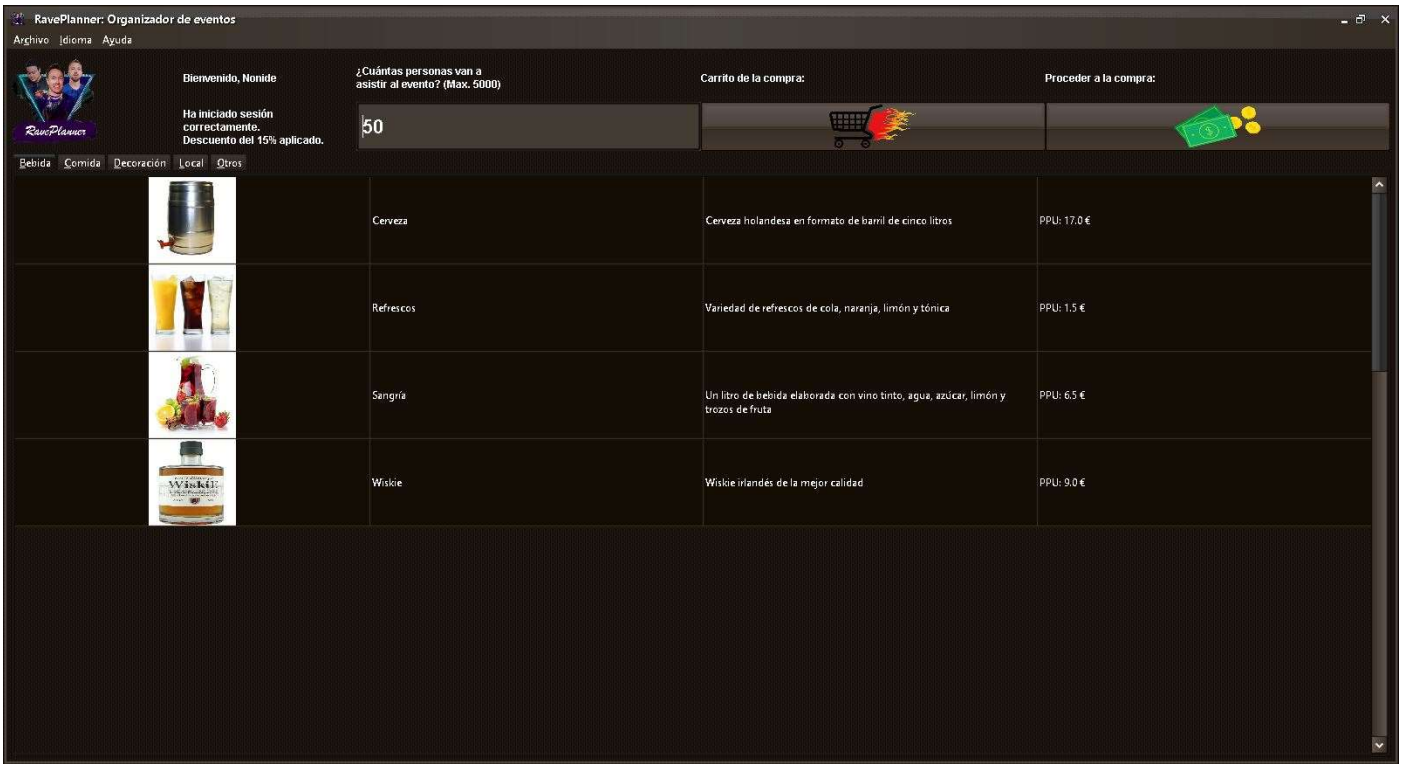
He tenido en cuenta el concepto de *ToolTip* y lo he aplicado a todo elemento interaccionable que he podido, así como el de mnemónico. Éstos también cambian con el idioma seleccionado.

Para lograr consistencia en la aplicación, he utilizado el mismo icono para todas las ventanas de la aplicación, y también he puesto el título principal de la aplicación en el título de cada ventana, seguido de la función de ésta. Además, he intentado usar siempre las mismas fuentes, con el mismo color y el mismo tamaño (según fuese pertinente).

Esta ventana tiene la capacidad de registrar en la base de datos unas credenciales escritas, y el nombre de usuario no ha de estar previamente tomado por otro usuario. También puedes dar de baja tus credenciales de la base de datos.

Para acceder a la pantalla principal de la aplicación, tienes la alternativa de escribir unas credenciales registradas y acceder, o bien puedes acceder anónimamente al sistema, lo cual generaría un nombre de usuario aleatorio y no registrado previamente en el sistema. Esto lo he hecho tomando como ejemplo sistemas de inicio de sesión anónimo de ciertas páginas web de compra de artículos en línea, las cuales te generan un nombre de usuario aleatorio, para registrar en su base de datos a la fecha en la que iniciaste sesión junto con tu ubicación y varios datos más, aunque yo no hice un sistema tan complejo por obvias razones.

Para localizar la aplicación a otro idioma desde esta ventana, basta con irte a la barra de menús, al apartado “Idioma”, y seleccionar el idioma que desees. Dicha barra también otorga acceso a la ayuda, al “Acerca de…”, y a una opción de salida total del sistema.



La **ventana principal** (***MainWindow***) ofrece las funcionalidades principales de la aplicación.

En ésta, se te informa acerca del nombre de usuario con el cual has iniciado sesión, además del porcentaje de descuento aplicado (mostrado solamente si no has iniciado sesión anónimamente). Podemos especificar también cuántas personas tenemos pensado que asistirán a nuestro evento, aunque tiene un límite (el autoimpuesto por el valor de un *int*, ya que en el trabajo no se especificaba cuál podría ser el límite). Podemos proceder a la compra, sí y sólo si el carrito de la compra no está vacío, mediante el botón situado más a la derecha.

El panel principal se genera automáticamente en base al archivo en el que vienen implícitos los datos de los artículos, dividiendo las categorías en paneles. Para comprar cierta cantidad de un artículo, el usuario pulsará dos veces en el artículo deseado, e introducirá por teclado la cantidad de unidades de ese artículo que desea, excepto para artículos grupales, que sólo se podrá introducir uno y no más a la selección de artículos.

Dentro de la **ventana del carrito de la compra** (***ShoppingCartWindow***) podremos eliminar las unidades que deseemos de un artículo mediante una ventana auxiliar que se nos mostrará al clicar dos veces sobre el artículo a eliminar. En el caso de artículos grupales, no hará falta especificar las unidades a eliminar.

Habiendo accedido al botón del proceso de compra, se nos mostrará la **ventana de formulario** (***FormWindow***), la cual solicitará el nombre del usuario, sus apellidos, su identificación, la fecha en la que se llevará a cabo el evento y, opcionalmente, unas observaciones que quiera aportar. Se comprueba que los cuatro primeros campos mencionados estén escritos, de lo contrario se le informará por pantalla al usuario de que no ha rellenado apropiadamente el formulario. También verifico con un *JFormattedTextField* que la fecha se encuentre en un formato apropiado, así me aseguro de que el archivo final de la factura sea del estilo: "N…NDDMMYYYY.txt" ("N…NMMDDYYYY.txt" si se ha elegido el idioma inglés), quitándole cualquier carácter que no sea un número o una letra, para que el sistema no interprete subcarpetas en la ruta de escritura de la factura.

La última ventana que veremos será la **ventana de facturación** (***InvoiceWindow***), que mostrará al usuario una previsualización de su factura, y si éste acepta que todo está correcto, generará una factura (en la subcarpeta *invoice* de la carpeta *files* en la que están la base de datos de los clientes y los datos de los artículos).

Una vez concluido el negocio, el programa se reiniciará a su estado inicial, volviendo a la ventana de inicio de sesión.

• Lógica

Para la capa de lógica de negocio ideé cuatro clases, las cuales realizan diversas labores:

- *Registrator*: Se encarga de manipular la base de datos de los usuarios, para lo que utiliza métodos estáticos. Puede registrar y dar de baja a un usuario dado su nombre de usuario y su contraseña, corrobora la existencia del nombre de usuario en la base de datos y comprueba que su contraseña sea correcta, y también puede generar un nombre de usuario aleatorio que no exista ya en la base de datos (para el inicio de sesión anónimo).

- *Article*: Representa un objeto "artículo", cuyos atributos son su código, su tipo, su nombre, su descripción, su precio como artículo unitario y su precio como artículo grupal. Comprueba que el código tiene 5 dígitos, y a su tipo se le capitaliza su letra inicial, y se le pasan a minúsculas el resto, por cuestiones formales.

- *Collator*: Organiza todos los artículos en una lista, así como sus categorías. También guarda el número de personas que asistirán al evento.

- *Invoice*: Se presenta como la clase que genera e imprime la factura, podría ser conceptualmente: "la caja registradora". Guarda los artículos que se van metiendo al carrito de la compra en un *HashMap*, en el cual cada entrada tiene como clave el artículo, y como valor la cantidad escogida. En cuanto a la selección de artículos, tiene un método *add* que añade cierta cantidad de un artículo al carrito de la compra y otro método *remove* que elimina cierta cantidad de un artículo del carrito. Sobre la factura, tiene un método *generateInvoice* que genera la factura en un solo valor cadena de caracteres y se la asigna a un atributo de la clase, y tiene otro método *typeInvoice* que finalmente genera la factura en la subcarpeta *invoice* de la carpeta *files* en la que están la base de datos de los clientes y los datos de los artículos.