Calidad de los Sistemas Informáticos, 2018-2019
Práctica 4 – Interfaz de detalle

Objetivo

El objetivo de esta práctica es la ampliación del proyecto creado en las prácticas anteriores para dotarlo de una interfaz de detalle que permita insertar, ver, editar, guardar y eliminar un objeto a través de la clase desarrollada previamente. Para esta práctica no se usarán pruebas unitarias: al ser interfaz de usuario, son los criterios de accesibilidad y usabilidad los que deben tenerse en cuenta.

Requisitos previos

Para la realización de esta práctica se requieren los recursos y resultados de las prácticas anteriores. Es asimismo necesario tener WindowBuilder en Eclipse en las condiciones que se explicaron en la práctica de instalación.

Criterios sintácticos

- `Courier`: código fuente, nombres de archivos, paquetes, entidades, atributos, tablas y campos.
- *Cursiva*: términos en otro idioma.
- Negrita: contenido resaltado.
- **$**Entre dólar**$**: contenido no textual, generalmente a decidir por el desarrollador.

Instrucciones previas

- En aquellas partes del código en la que se consideren adecuadas precondiciones (generalmente, al inicio de cada método), que deberán indicarse mediante el comentario hasta el último punto de esta práctica, en el que se propone la implementación de las mismas.

```
// TODO: Preconditions
```

- En esta fase ya es requerido avisar al usuario en caso de error o excepción. Para ello, todas las excepciones y errores deben mostrar un mensaje en pantalla mediante el uso de `JOptionPane`, de la forma en la que se muestra, garantizando que el mensaje de error es suficientemente descriptivo como para garantizar cierto grado de fiabilidad de la aplicación.

```
JOptionPane.showMessageDialog(null,
    "mensaje", "Error", JOptionPane.ERROR_MESSAGE);
```

Pablo de la Torre Moreno
Departamento de Ingeniería Informática
5 de noviembre de 2018

Procedimiento

1. Creación de esqueleto y ventana principal

1. Crear en `src` un paquete `es.uca.gii.csi18.$equipo$.gui`, donde se crearán las clases que gestionarán la interfaz de usuario.
2. Crear en dicho paquete un formulario principal llamado `FrmMain` (clase `FrmMain`) en dicho paquete, mediante [New... > Other... > WindowBuilder > Swing Designer > Application Window].
3. Ejecutar dicha clase (botón derecho, *Run As*) como aplicación Java y observar que se abre la ventana.

2. Configuración y creación del menú

1. Con el código de `FrmMain.java` visible, pulsar en la pestaña [Design] (generalmente, abajo) para mostrar el diseñador. Pulsar sobre el borde del boceto de la ventana para activar el diseñador de ésta.
2. En la sección [Properties], buscar `title` y dar un título a la ventana principal, suficientemente descriptivo del proyecto.
3. En la sección [Palette > Menu], pulsar `JMenuBar` y luego pulsar otra vez sobre el boceto de la ventana, a la derecha. El menú deberá quedar colocado con el mensaje "(Add items here)".
4. En la sección [Palette > Menu], pulsar `JMenu` y luego pulsar sobre el menú previamente ubicado. En la caja de texto, poner "Nuevo". En la sección [Properties], cambiar el nombre de `Variable` por `mitNew` o `mitNuevo`.
5. Repetir la operación del punto anterior con "Buscar", asignando a `Variable` "mitSearch" (o "mitBuscar"). “Nuevo” y “Buscar” deben quedar uno junto al otro, en la barra de menú.
6. Observar la sección [Components] para ver cómo está quedando el árbol de estructura de la ventana.
7. Pulsar sobre “Nuevo”. En la sección [Palette > Menu], pulsar `JMenuItem` y luego pulsar otra vez sobre "Nuevo", en la sección [Components]. Escribir el nombre del tipo de elemento del que se haya realizado la clase de mapeo ("Usuario", "Persona", "Vehículo"...). En la sección [Properties], cambiar el nombre de la variable por `mitNew$Entity$` o `mitNuevo$Entidad$` (por ejemplo, `mitNewUser` o `mitNuevoUsuario`).
8. Repetir lo mismo sobre "Buscar", con `mitSearch$Entity$` o `mitBuscar$Entity$`.
9. En el código fuente de `FrmMain`, arriba, refactorizar el nombre de la variable privada de tipo `JFrame` renombrándola como `frame` (con el botón derecho sobre el nombre).
10. Ejecutar como aplicación Java y observar el resultado.
11. Opcionalmente, dar aspecto de aplicación del sistema operativo en uso mediante incluir, al principio del constructor de `FrmMain.java` justo antes de la llamada a `initialize()`:

```
UIManager.setLookAndFeel(
    UIManager.getSystemLookAndFeelClassName());
```

12. De igual manera, se puede cambiar la posición inicial y el tamaño modificando los parámetros del método `setBounds` (posición X, posición Y, ancho y alto, respectivamente).
13. Ejecutar como aplicación Java y observar el resultado.

3. Creación del formulario de detalle

1. En la sección [Components], seleccionar `getContentPane()` y, en la sección [Properties], seleccionar `Layout` como “Absolute Layout".
2. Crear un formulario (en el mismo paquete), a través de WindowBuilder mediante [New... > Other... > WindowBuilder > Swing Designer > JInternalFrame]. Se denominará `Ifr$Entidad$`, en español o inglés (por ejemplo, `IfrUser` o `IfrLicencia`).
3. Eliminar el método `main` y su comentario.
4. Acceder a [Design].
5. Poner como título a la ventana el tipo de elemento (por ejemplo, "Usuario" o "Licencia"), seleccionando en [Components] la raíz del árbol y buscando `title` en la sección [Properties]. En la misma sección, poner `closable` y `resizable` a `true` (de lo contrario, no será posible cerrar ni redimensionar la ventana en ejecución).
6. En la sección, [Palette > Layouts], pulsar `AbsoluteLayout` y luego pulsar sobre el centro del boceto de la ventana.
7. Diseñar el formulario mediante tantos elementos de [Palette > Components] (o los que se requiera) de tipo `JLabel` y `JTextField`, así como `JCheckBox` u otros controles, si fuera necesario, sin olvidar el criterio de denominación de controles (prefijos lbl, txt, chk...; por ejemplo, `lblDni` para el texto “DNI”, `txtDni` para su contenido, etcétera[1]).
8. Incluir un botón (`JButton`) "Guardar" (`butSave` o `butGuardar`).
9. En el código del formulario, crear arriba una variable privada de la clase del tipo de elemento e igualarla a `null` (`private User _user = null;` por ejemplo). Será necesario importar el paquete `data` pertinente. Dicha variable empieza siendo `null`, pero podrá no serlo si el formulario actual es invocado para un elemento ya existente (lo que se verá en la próxima práctica) o tras guardar un elemento nuevo (que se verá a continuación).
10. Eliminar posibles *warnings* (salvo el generado por la variable privada recién creada).

4. Añadir funcionalidad al formulario de detalle

1. En el diseño, sobre el botón "Guardar" pulsar con el botón derecho y seleccionar [Add event handler > action > actionPerformed]. De esta forma, se le creará un código que responda al evento de pulsación del botón.
2. En el método automáticamente creado, asignar la variable del tipo de elemento a lo siguiente[2]:
    a. Si la variable privada creada en el punto 3.9 es `null` (por defecto lo es, pero en el futuro no lo será siempre), asignar dicha variable al resultado de llamar al método estático `Create` de su clase (creado en la práctica segunda)

[1] El contenido textual de un `JLabel` se cambia en la sección [Properties], variable `text`.
[2] El código requerirá `try/catch`: incluirlo todo en él, no un `try/catch` dentro de cada `if/else`.

incluyendo como parámetros los valores de las cajas de texto, según sean necesarios[3], (por ejemplo; `txtDni.getText()`sería un parámetro, en su caso))[4].

b. En caso contrario (si no es `null`), invocar para dicha variable tantos métodos `set` como campos hay en el formulario, respectivamente; posteriormente, ejecutar su método `Update()`.

5. Invocar al formulario detalle y hacer pruebas

1. En `FrmMain.java`, sobre el menú [Nuevo > **$**Entidad**$**] del boceto de la ventana pulsar con el botón derecho y seleccionar [Add event handler > action > actionPerformed].
2. En el código, incluir algo similar a esto (cambiar los límites en `setBounds`, si no se ajusta bien a la pantalla):

```
Ifr$Entidad$ ifr$Entidad$ = new Ifr$Entidad$();
ifr$Entidad$.setBounds(10, 27, 244, 192);
frame.getContentPane().add(ifr$Entidad$);
ifr$Entidad$.setVisible(true);
```

3. Compilar, ejecutar y hacer pruebas, comprobando que: a) se crea en la base de datos el primer elemento guardado; y b) posteriores pulsaciones a "Guardar" modifica dicho elemento en la base datos (por ejemplo, mirando directamente las filas en phpMyAdmin). Esto deberá producirse por cada [Nuevo > $Entidad$]. Si no funciona adecuadamente, se requerirá revisar los métodos de prueba de prácticas anteriores.
4. Comprobar y arreglar errores como dejar campos vacíos o escribir letras en lugar de números en los campos que requieran esto último.
5. Ajustar código (variables, espacios, tabulaciones) a la sintaxis y los criterios de la asignatura antes de entregar.

[3] En caso de haber alguna fecha, no se aconseja perder el tiempo con ella durante la práctica. En lugar de ello, se enviará `new Date()` al crear y nada al actualizar, y se pospondrá para trabajo de casa o si la práctica se finaliza con tiempo.

[4] Puede que sea necesario trasladar la declaración de algunos controles fuera del constructor.