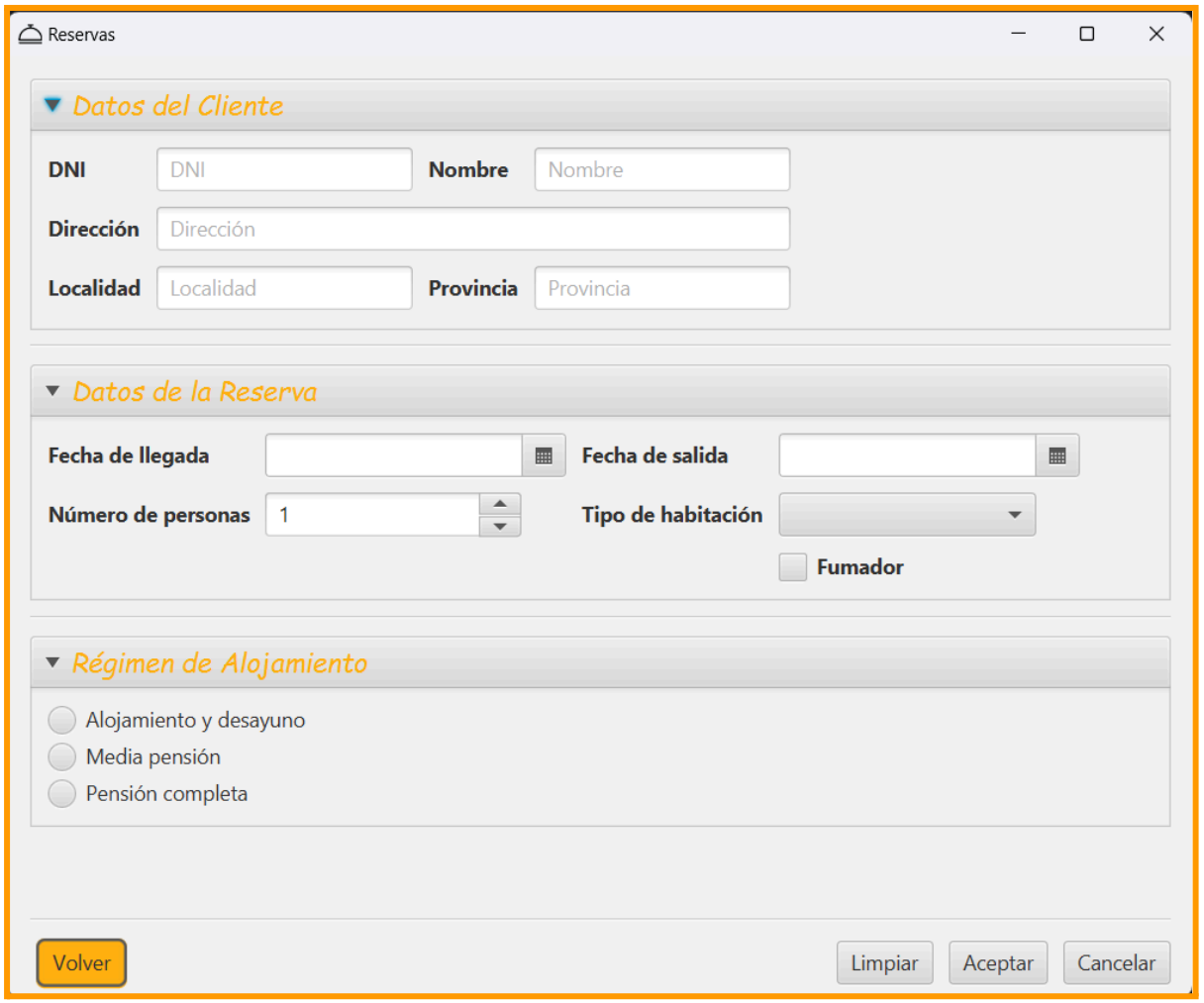
~ Crea la interfaz: Reservas Hotel ~

Ivan David Velazquez Aguilar

Desarrollo de Aplicaciones Multiplataforma

Curso 2024 / 2025

Desarrollo De Interfaces



Índice.

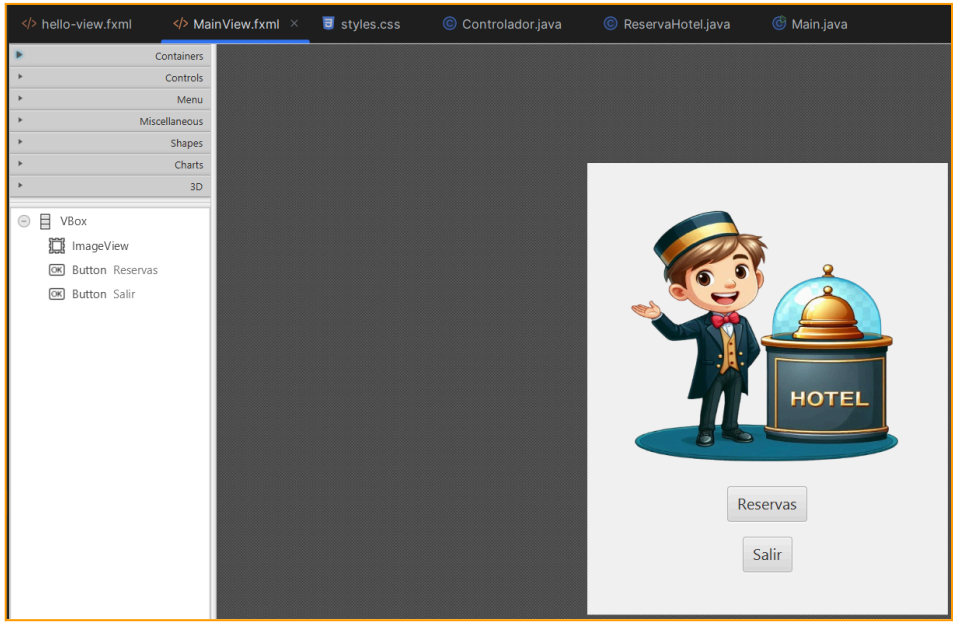
1. Creación de la interfaz.....3
2. Cambio de nombres y ToolTip para los elementos del formulario.....5
3. Distribución de los elementos usando el modo de distribución libre....6
4. Modificación de la fuente para dar vistosidad a la interfaz.....7
5. Código para abrir el diálogo desde el menú principal.....9
6. Gestión de eventos del checkbox de fumadores..... 10
7. Combinaciones de teclas para campos de texto (Alt + tecla)..... 11
8. Cumplimiento del formato de entrega..... 11

1. Creación de la interfaz.

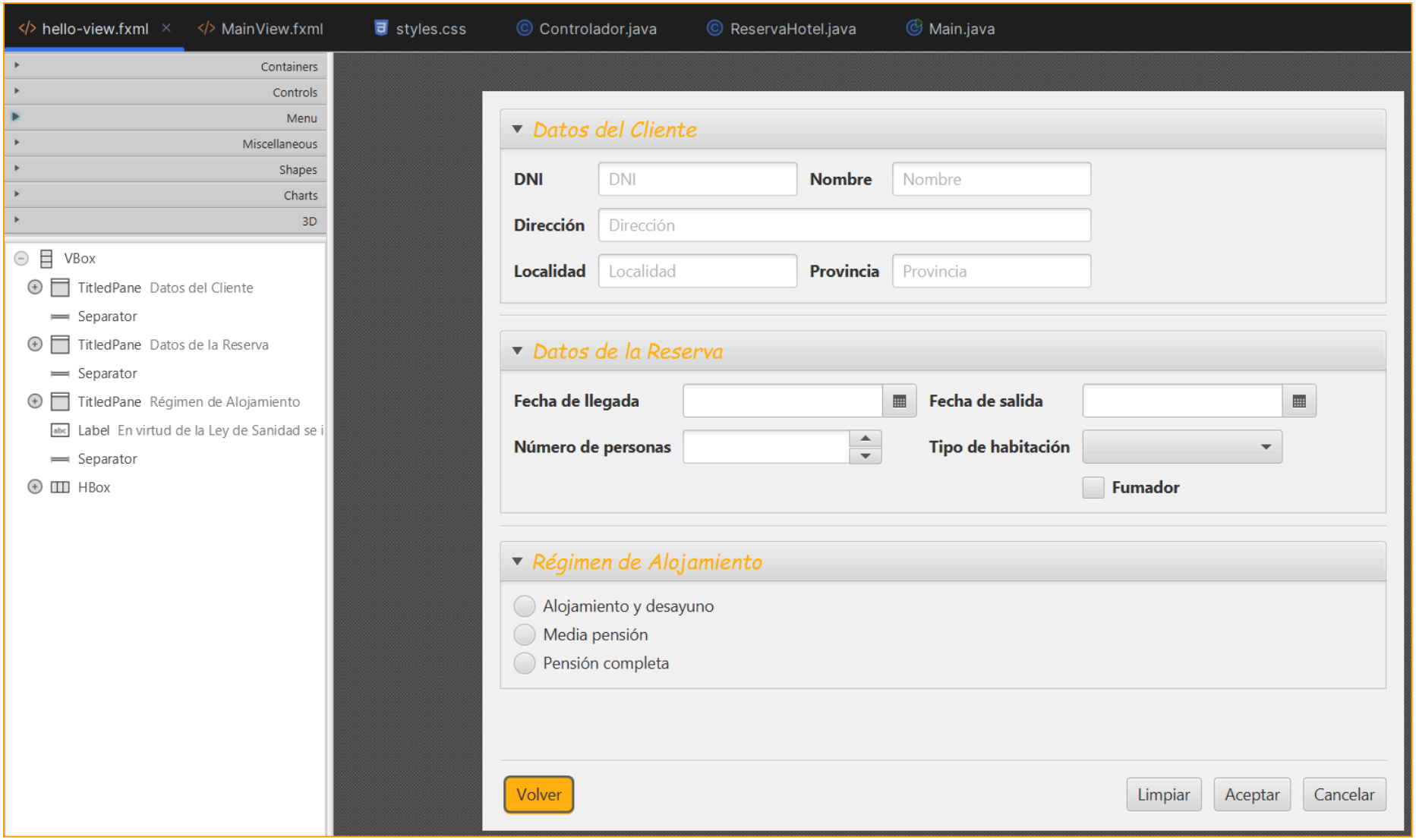
Para la elaboración de la interfaz se emplearon los componentes más adecuados disponibles en JavaFX, incluyendo *VBox*, *GridPane*, *Label*, *TextField*, *ComboBox*, *CheckBox*, y *Button*. Se utilizó *VBox* y *GridPane* para una estructura organizada que facilita la colocación de los elementos, haciendo que la interfaz sea intuitiva y fácil de usar para el usuario final.

El *VBox* se utilizó para crear una disposición vertical que permite agrupar elementos relacionados de forma clara y ordenada, mientras que el *GridPane* se empleó para distribuir los campos y sus etiquetas de manera estructurada, asegurando una alineación consistente. Estos componentes garantizan que la interfaz sea flexible y que pueda adaptarse a diferentes tamaños de pantalla sin perder su estética ni su funcionalidad.

MainView.fxml



hello-view.fxml



2. Cambio de nombres y ToolTip para los elementos del formulario.

Cada uno de los componentes de la interfaz fue etiquetado con nombres significativos para mejorar la legibilidad del código y facilitar su mantenimiento. Por ejemplo, en lugar de nombres genéricos como ***textField1***, se utilizaron identificadores descriptivos como *campoDNI* o *campoNombre*, lo cual facilita a cualquier desarrollador entender la función de cada elemento sin necesidad de revisar exhaustivamente el código.

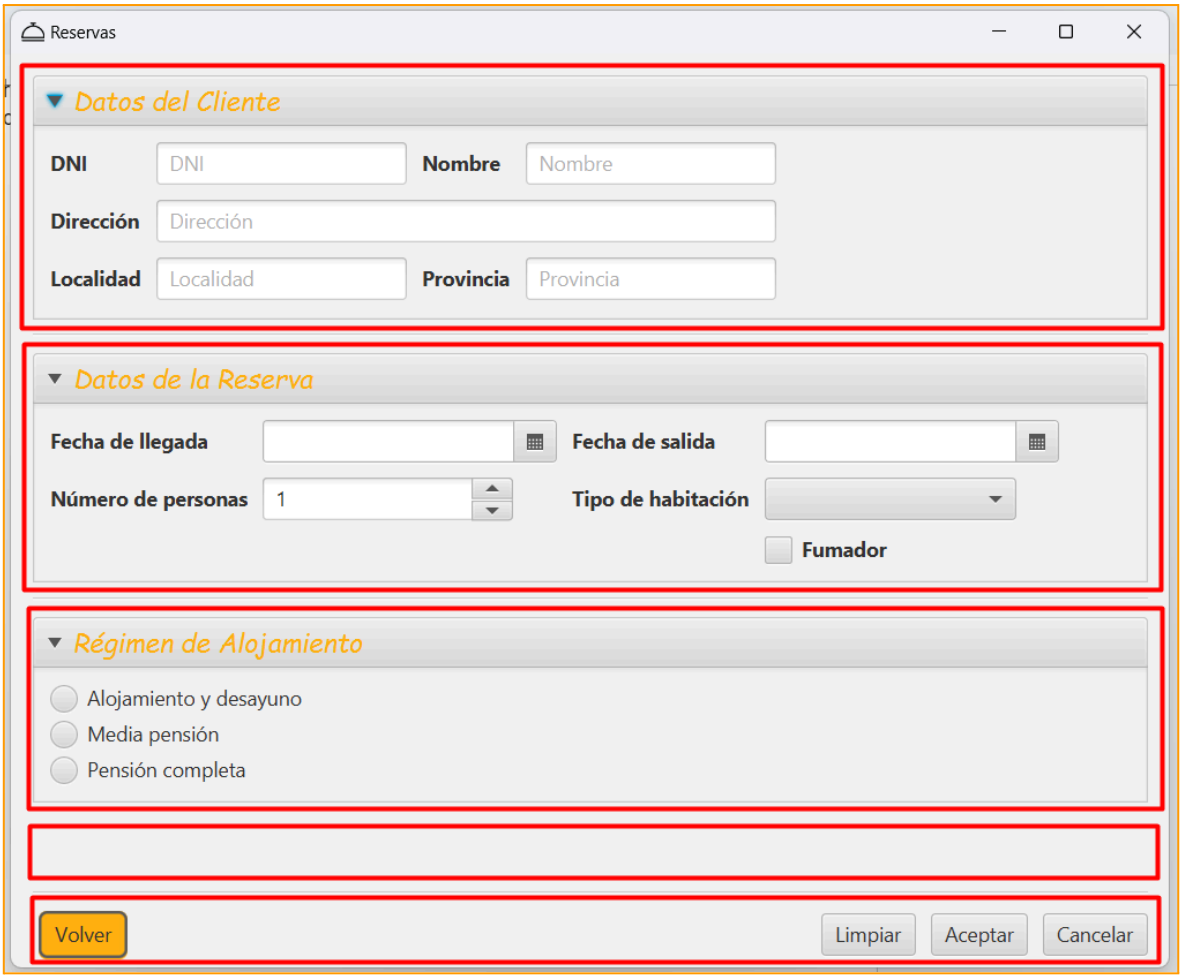
Además, se agregaron **ToolTip** a los campos para proporcionar al usuario información adicional sobre cómo rellenar el formulario. Por ejemplo, el campo *campoDNI* tiene un **ToolTip** que indica al usuario que debe ingresar su DNI con el formato adecuado, y el campo *campoNombre* muestra información sobre cómo debe introducir su nombre completo. Esto hace que la interfaz sea más amigable y reduce la posibilidad de errores por parte del usuario.

```
// Añadir Tooltips a cada componente
campoDNI.setTooltip(new Tooltip( s: "Introduzca el DNI del cliente."));
campoNombre.setTooltip(new Tooltip( s: "Introduzca el nombre completo del cliente."));
campoDireccion.setTooltip(new Tooltip( s: "Introduzca la dirección del cliente."));
campoLocalidad.setTooltip(new Tooltip( s: "Introduzca la localidad del cliente."));
campoProvincia.setTooltip(new Tooltip( s: "Introduzca la provincia del cliente."));
fechaLlegada.setTooltip(new Tooltip( s: "Seleccione la fecha de llegada al hotel."));
fechaSalida.setTooltip(new Tooltip( s: "Seleccione la fecha de salida del hotel."));
numeroPersonas.setTooltip(new Tooltip( s: "Seleccione el número de personas que se alojarán."));
tipoHabitacion.setTooltip(new Tooltip( s: "Seleccione el tipo de habitación deseado."));
esFumador.setTooltip(new Tooltip( s: "Marque si la habitación es para fumador."));
desayunoRadio.setTooltip(new Tooltip( s: "Régimen de alojamiento: Alojamiento y desayuno."));
mediaPensionRadio.setTooltip(new Tooltip( s: "Régimen de alojamiento: Media pensión."));
pensionCompletaRadio.setTooltip(new Tooltip( s: "Régimen de alojamiento: Pensión completa."));
advertenciaFumador.setTooltip(new Tooltip( s: "Aviso: Solo se permite fumar en habitaciones designadas para fumadores."));
btnLimpiar.setTooltip(new Tooltip( s: "Limpia todos los campos del formulario."));
btnAceptar.setTooltip(new Tooltip( s: "Acepta y guarda la reserva."));
btnCancelar.setTooltip(new Tooltip( s: "Cancela la reserva actual."));
```

3. Distribución de los elementos usando el modo de distribución libre.

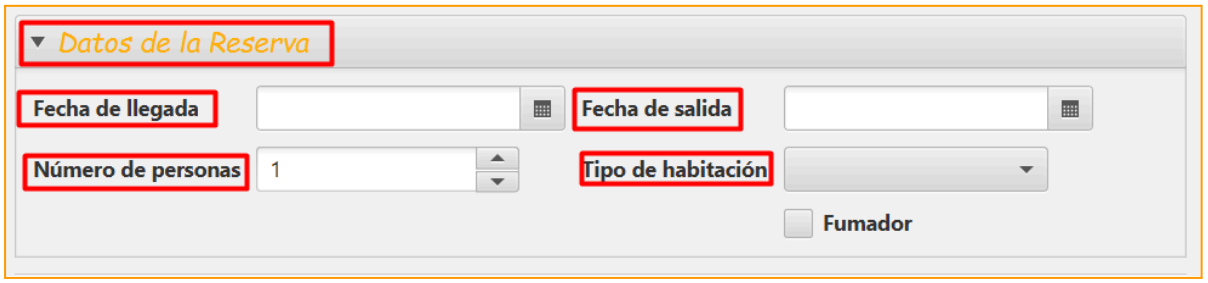
Para la distribución de los elementos, se utilizó *GridPane* dentro de un *VBox*, lo que permite una organización clara y flexible de la información. El uso de *GridPane* permite alinear los componentes en filas y columnas, haciendo que el formulario sea intuitivo y fácil de navegar. Cada etiqueta (`Label`) está alineada con su campo correspondiente (`TextField, ComboBox, etc.`), lo cual hace que el formulario sea coherente y fácil de llenar para el usuario.

El uso de un diseño de distribución libre también facilita la extensión futura de la interfaz, ya que se pueden añadir más elementos sin romper la estructura existente. Además, los elementos fueron organizados de forma lógica, agrupando la información relacionada (*como los datos del cliente y los datos de la reserva*), lo cual mejora la experiencia del usuario al completar el formulario.



```
<!-- Panel de datos del cliente -->
<TitledPane expanded="true" text="Datos del Cliente" textFill="#ffb013">
    <VBox spacing="10">
        <GridPane hgap="10" vgap="10">
            <Label fx:id="labelDNI" mnemonicParsing="true" style="-fx-for
            <TextField fx:id="campoDNI" promptText="DNI" GridPane.columnI

            <Label fx:id="labelNombre" mnemonicParsing="true" style="-fx-
            <TextField fx:id="campoNombre" promptText="Nombre" GridPane.c

            <Label fx:id="labelDireccion" mnemonicParsing="true" style="-
            <TextField fx:id="campoDireccion" promptText="Dirección" Gric

            <Label fx:id="labelLocalidad" mnemonicParsing="true" style="-
            <TextField fx:id="campoLocalidad" promptText="Localidad" Gric

            <Label fx:id="labelProvincia" mnemonicParsing="true" style="-
            <TextField fx:id="campoProvincia" promptText="Provincia" Gric

        </GridPane>
    </VBox>
  <font>
     <Font name="Comic Sans MS Italic" size="18.0" />
  </font>
</TitledPane>
```

4. Modificación de la fuente para dar vistosidad a la interfaz.



Para mejorar la apariencia visual de la interfaz, se modificó la fuente de las etiquetas principales para hacerlas más vistosas. Se utilizó un tamaño de fuente de 14px con un estilo en negrita (`-fx-font-weight: bold`), lo cual hace que los títulos y las etiquetas sean más fáciles de identificar y resaltan sobre los campos de entrada.

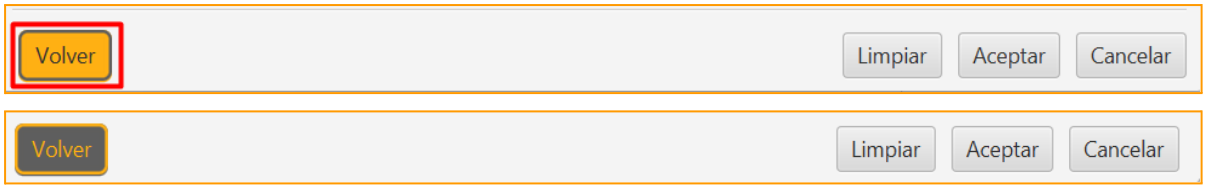
Esta modificación ayuda a guiar al usuario a través del formulario, destacando las secciones importantes y mejorando la legibilidad general. La elección de un tamaño de fuente y un peso específico también contribuye a la accesibilidad de la aplicación, asegurando que la información sea visible para usuarios con diferentes capacidades visuales.

```
<CheckBox fx:id="esFumador" mnemonicParsing="true" style="-fx-font-weight: bold;" text="_Fumador"
```

>> También el título del *TitledPane* ha sido modificado para resaltar y visualizar mejor el título de capa *GridPane*.

```
<!-- Panel de datos de la reserva -->
<TitledPane expanded="true" text="Datos de la Reserva" textFill="#ffb013">
    <font>
        <Font name="Comic Sans MS Italic" size="18.0" />
    </font>
```

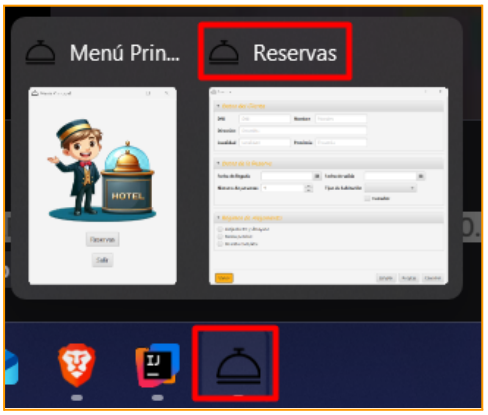
>> Adicionalmente he metido efecto en el botón de "**Volver**", que cambia su estilo por default y para cuando se pasa el ratón por encima. Se puede ver en el archivo styles.css las modificaciones al elemento.

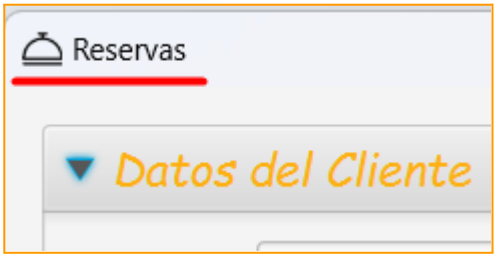


styles.css × hello-view.fxml Main\

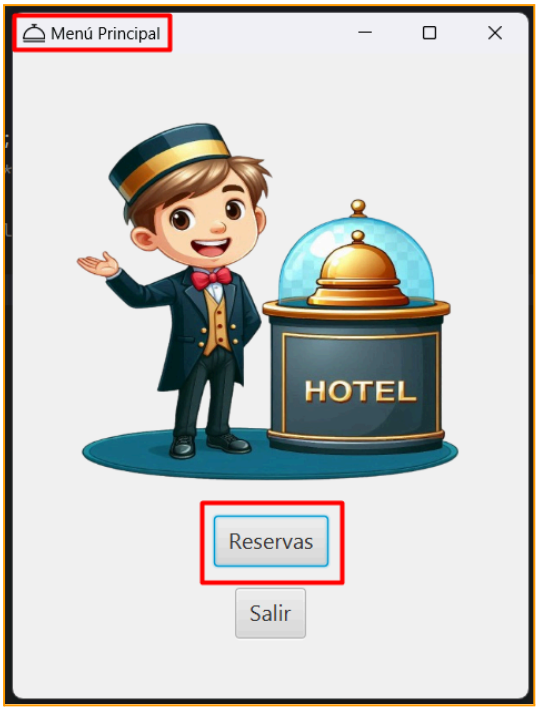
```
#btnVolver {
    -fx-background-color: #ffb013;
    -fx-border-color: #5e5e5e;
    -fx-border-width: 2px;
    -fx-border-radius: 5px;
    -fx-background-radius: 6px;
}

#btnVolver:hover {
    -fx-background-color: #5e5e5e;
    -fx-border-color: #ffb013;  /*
    -fx-text-fill: #ffb013;
    -fx-cursor: hand; /* Cambia el
```

>> Y como extra he metido un icono en las ventanas, todo esto ayuda a que la interfaz sea más amena para el usuario.





5. Código para abrir el diálogo desde el menú principal.



Para abrir el diálogo desde el menú principal, se añadió un botón que gestiona el evento mediante un controlador. El botón "*Reservas*" en *MainView.fxml* está configurado con un `onAction` que llama al método `abrirReservas()`, el cual se encarga de cargar el archivo FXML correspondiente (*hello-view.fxml*) y abrirlo en una nueva ventana.

El método `abrirReservas()` utiliza la clase **FXMLLoader** para cargar la vista y crear un nuevo **Stage** donde se muestra la interfaz de reservas. Esta implementación asegura una separación clara entre el menú principal y la interfaz de reservas, lo cual mejora la modularidad del proyecto. Cada ventana funciona de manera independiente, proporcionando una experiencia de usuario coherente y organizada.

MainView.fxml

```
<Button fx:id="btnReservas" onAction="#abrirReservas" style="-fx-font-size: 16px;" text="Reservas" />
<Button fx:id="btnSalirAplicacion" onAction="#salirAplicacion" style="-fx-font-size: 16px;" text="Salir" />
```

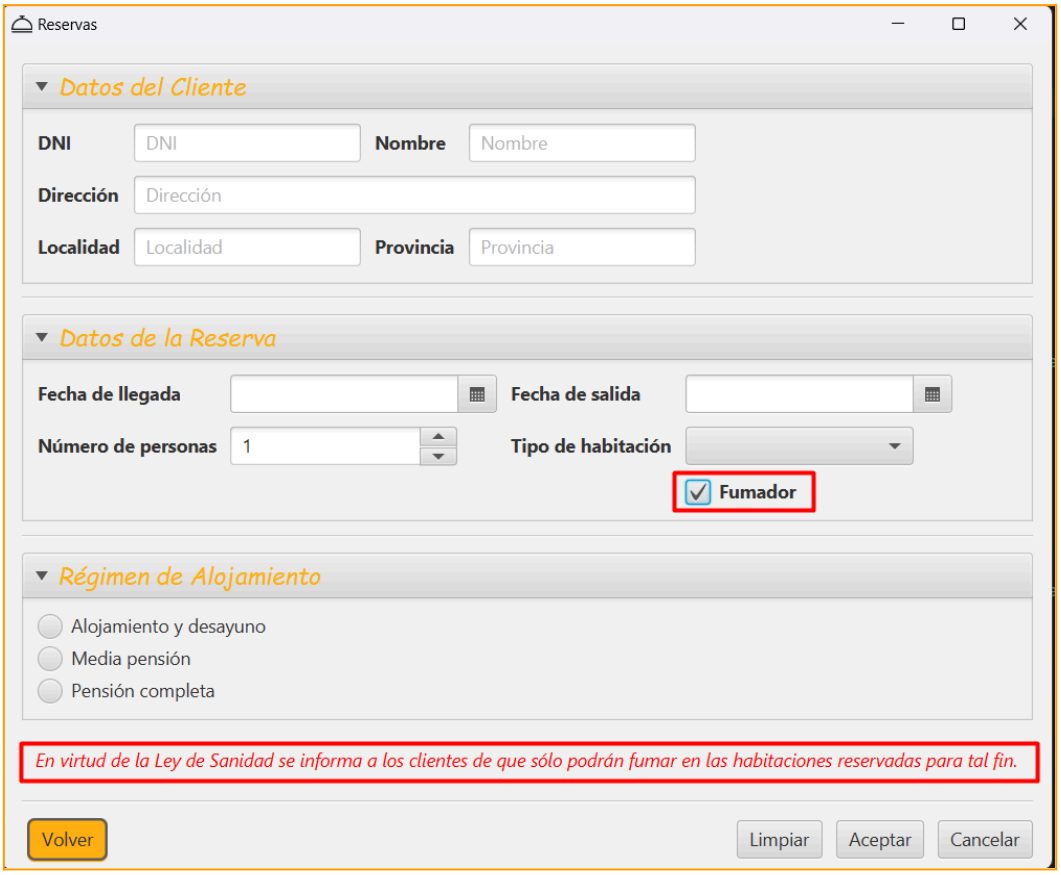
ReservaHotel.java *(Metodo que abre, la ventana de Reservas)*

```
@FXML
private void abrirReservas() {
    try {
        // Cargar el archivo FXML de la vista de reservas
        FXMLLoader loader = new FXMLLoader(getClass().getResource( name: "/com/example/interfaz/hello-view.fxml"));
        Parent root = loader.load();

        // Crear una nueva ventana para la vista de reservas
        Stage reservaStage = new Stage();
        reservaStage.setTitle("Reservas");

        // Configurar el icono de la ventana
        Image icon = new Image(Objects.requireNonNull(getClass().getResourceAsStream( name: "/com/example/interfaz/icono.png")));
        reservaStage.getIcons().add(icon);

        reservaStage.setScene(new Scene(root));
        reservaStage.show();
```

6. Gestión de eventos del checkbox de fumadores.

Para el checkbox "Fumador", se implementó un Listener que detecta cambios en su estado. Si el checkbox está seleccionado, se muestra un mensaje de advertencia en un Label indicando las restricciones sobre fumar. Este mensaje se oculta si el checkbox no está activo, cumpliendo con el requerimiento de la gestión de eventos.

Este enfoque asegura que el usuario esté siempre informado sobre las políticas de fumar, evitando posibles malentendidos. Además, la implementación con un Listener proporciona una respuesta inmediata en la interfaz, lo cual mejora la experiencia del usuario al darle retroalimentación visual instantánea basada en sus acciones.



>> También se han hecho cambios en el texto, para dar una mejor interfaz. El código para poder realizarlo es el siguiente:

Controlador.java

```
// Listener para mostrar u ocultar la advertencia para fumadores
esFumador.selectedProperty().addListener((ObservableValue<extends Boolean> observable, Boolean oldValue, Boolean newValue) -> {
    advertenciaFumador.setVisible(newValue);
});
```

hello-view.fxml

```
<!-- Mensaje de advertencia para fumadores -->
<Label fx:id="advertenciaFumador" style="-fx-text-fill: red; -fx-font-style: italic; -fx-padding: 10;"
       text="En virtud de la Ley de Sanidad se informa a los clientes de que sólo podrán fumar en las habitaciones reservadas para tal fin."
       visible="false" wrapText="true" />
```

7. Combinaciones de teclas para campos de texto (Alt + tecla)

Se añadieron combinaciones de teclas usando **mnemonicParsing** para cada etiqueta. Esto permite que el usuario pueda acceder rápidamente a cada campo usando combinaciones de teclado, como Alt + N para el campo de "Nombre". Estas combinaciones de teclas mejoran la usabilidad, especialmente para usuarios avanzados que prefieren utilizar el teclado sobre el ratón.

El uso de **mnemonicParsing** hace que la navegación dentro del formulario sea más eficiente, permitiendo a los usuarios alternar entre los diferentes campos de manera fluida. Esta característica es particularmente útil para aquellos usuarios que necesitan ingresar muchos datos de forma rápida, ya que reduce significativamente el uso del ratón y mejora la accesibilidad general del formulario.

Importante que el "**text**" lleve un "_" porque la letra que le sigue será la que se va a seleccionar con el alt.

```
<Label fx:id="labelDNI" mnemonicParsing="true" style="-fx-font-weight: bold;" text="_DNI" GridP
<TextField fx:id="campoDNI" promptText="DNI" GridPane.columnIndex="1" GridPane.rowIndex="0" />
```

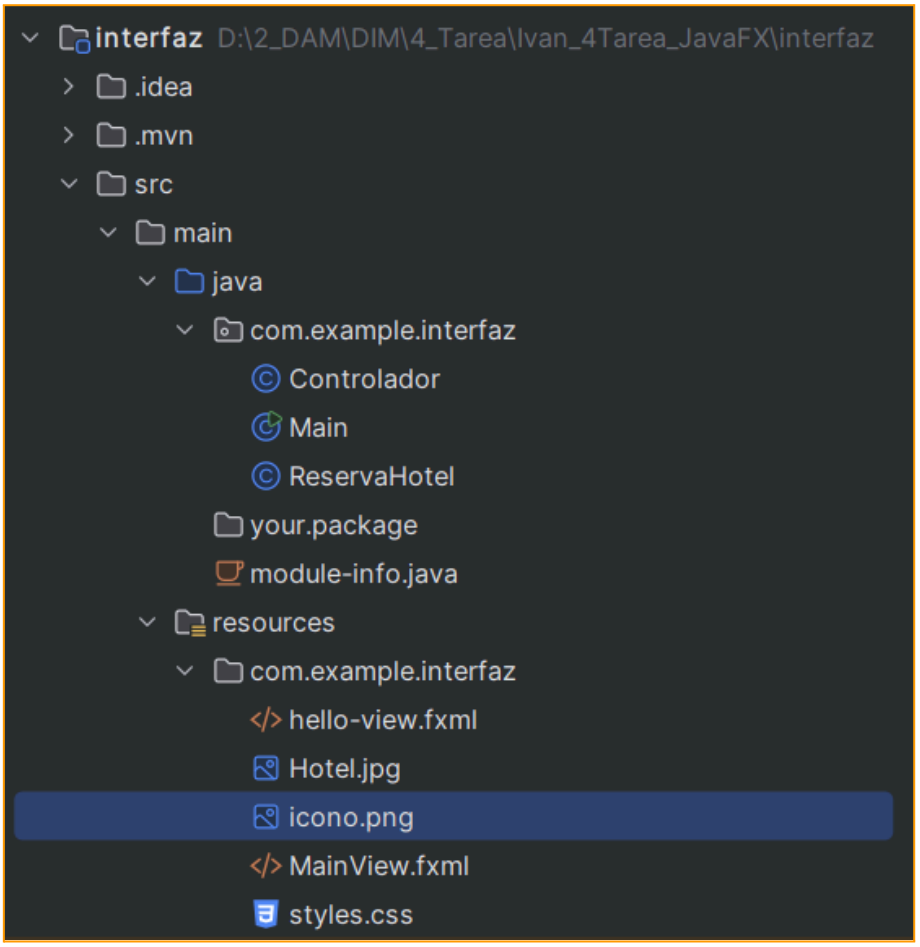
>> El efecto obviamente se ve mejor cuando se ejecuta el programa.

8. Cumplimiento del formato de entrega.

La entrega de la práctica se realizó siguiendo el formato especificado en el apartado "Indicaciones de entrega", asegurándose de que todos los puntos requeridos estuvieran cubiertos y que el código estuviera bien documentado. Se proporcionaron comentarios explicativos para cada método clave, facilitando la revisión y mantenimiento del proyecto.

Cada sección del proyecto se organizó siguiendo las mejores prácticas de desarrollo, incluyendo una adecuada separación de responsabilidades mediante controladores y vistas bien definidas. Esto facilita la escalabilidad del proyecto y permite a futuros desarrolladores entender rápidamente la estructura y lógica de la

aplicación. Además, se utilizó un enfoque modular que garantiza que cada parte del sistema pueda ser mejorada o modificada sin afectar negativamente otras partes del código.



>> Github con el código.

https://github.com/Ivan0522/InterfazReservaHotel.git

~ FIN ~