UNIVERSIDAD TECNOLOGICA DE HONDURAS



**Facultad**

Ingeniería en Computación

**Asignatura**

Programación Avanzada II

**Tema**

Proyecto Final

**Nombre**

Jahdiel Oseas López Escoto

**Cuenta**

202410010192

Ing. Louis Zelaya

San Pedro Sula, Cortés 3 de abril de 2025

**PROYECTO HOTEL**

**QUERY DE LA CREACION DE LA BASE DE DATOS Y SUS TABLAS**

CREATE DATABASE proyecto\_hotel;

USE proyecto\_hotel;

CREATE TABLE huespedes (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100) NOT NULL,

apellido VARCHAR(100) NOT NULL,

correo VARCHAR(100) UNIQUE NOT NULL,

telefono VARCHAR(15),

identidad VARCHAR(50) UNIQUE NOT NULL,

activo TINYINT(1) DEFAULT 1

);

CREATE TABLE tipo\_habitaciones (

id INT AUTO\_INCREMENT PRIMARY KEY,

tipo VARCHAR(50) NOT NULL,

capacidad INT NOT NULL,

descripcion VARCHAR(255)

);

CREATE TABLE habitaciones (

id INT AUTO\_INCREMENT PRIMARY KEY,

numero\_habitacion VARCHAR(50) NOT NULL,

tipo\_habitacion\_id INT NOT NULL,

activo TINYINT(1) DEFAULT 1,

FOREIGN KEY (tipo\_habitacion\_id) REFERENCES tipo\_habitaciones(id)

);

CREATE TABLE estados\_reserva (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_estado VARCHAR(50) NOT NULL UNIQUE

);

CREATE TABLE reservas (

id INT AUTO\_INCREMENT PRIMARY KEY,

huesped\_id INT NOT NULL,

habitacion\_id INT NOT NULL,

fecha\_entrada DATE NOT NULL,

fecha\_salida DATE NOT NULL,

estado\_reserva\_id INT NOT NULL,

precio DECIMAL(10,2) NOT NULL,

activo TINYINT(1) DEFAULT 1,

FOREIGN KEY (huesped\_id) REFERENCES huespedes(id),

FOREIGN KEY (habitacion\_id) REFERENCES habitaciones(id),

FOREIGN KEY (estado\_reserva\_id) REFERENCES estados\_reserva(id)

);

CREATE TABLE metodos\_pago (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_metodo VARCHAR(50) NOT NULL UNIQUE

);

CREATE TABLE pagos (

id INT AUTO\_INCREMENT PRIMARY KEY,

reserva\_id INT NOT NULL,

monto DECIMAL(10,2) NOT NULL,

metodo\_pago\_id INT NOT NULL,

fecha\_pago TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

activo TINYINT(1) DEFAULT 1,

FOREIGN KEY (reserva\_id) REFERENCES reservas(id),

FOREIGN KEY (metodo\_pago\_id) REFERENCES metodos\_pago(id)

);

CREATE TABLE cargos\_empleados (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_cargo VARCHAR(50) NOT NULL UNIQUE

);

CREATE TABLE empleados (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100) NOT NULL,

apellido VARCHAR(100) NOT NULL,

cargo\_id INT NOT NULL,

correo VARCHAR(100) UNIQUE NOT NULL,

telefono VARCHAR(15),

salario DECIMAL(10,2) NOT NULL,

activo TINYINT(1) DEFAULT 1,

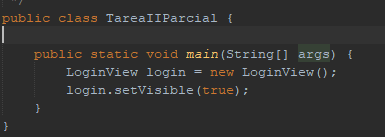
FOREIGN KEY (cargo\_id) REFERENCES cargos\_empleados(id)

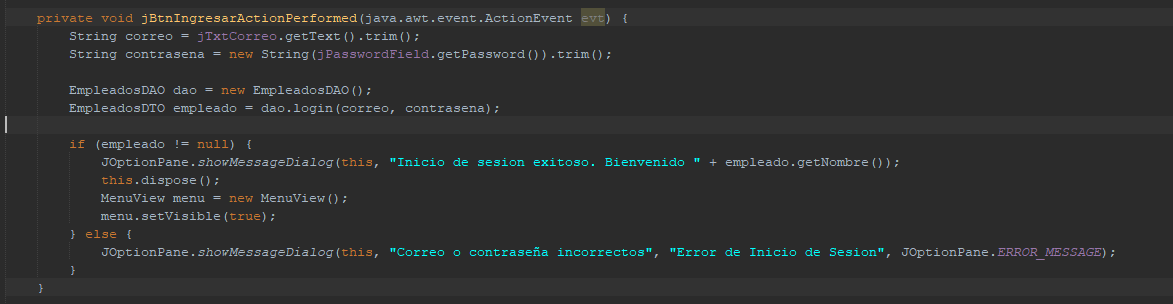
);

ALTER TABLE cargos\_empleados ADD COLUMN activo TINYINT(1) DEFAULT 1;

ALTER TABLE empleados ADD COLUMN contrasena VARCHAR(255) NOT NULL;

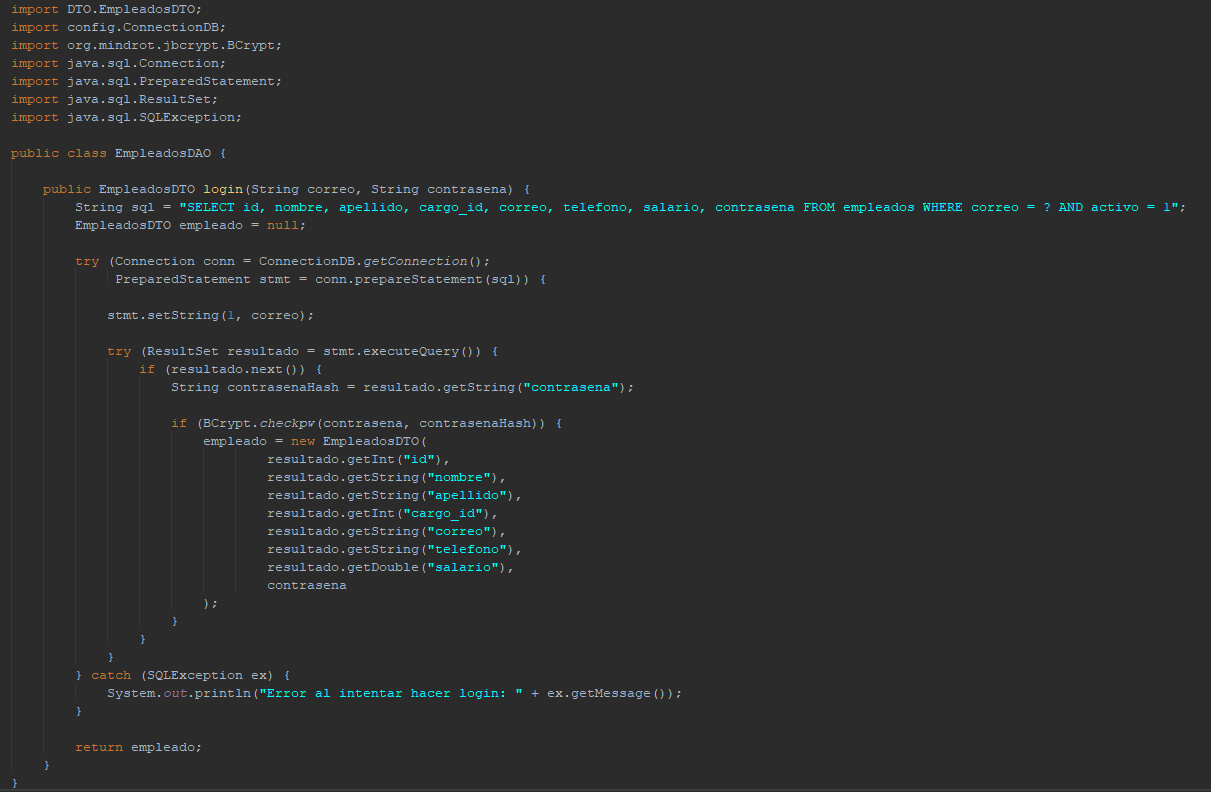
**CODIGO DE FUNCIONALIDAD DEL LOGIN**

****

****

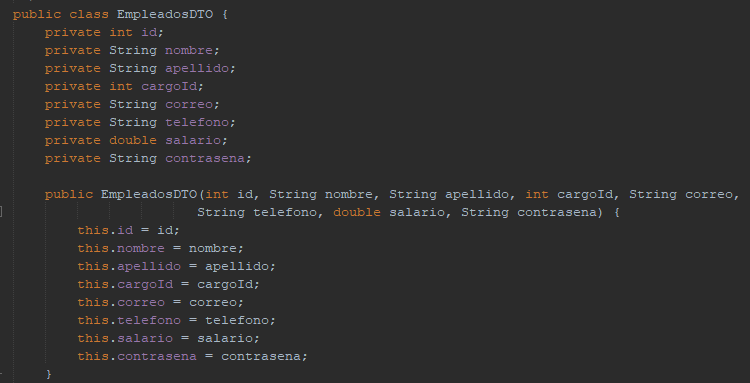
En mi metodo principal main, hago una instancia de mi vista de Login para que sea esta la que se muestra al principio, esta cuenta con 2 campos de texto y un botón para ingresar, en el método del botón obtengo el valor de mi campo correo y de mi contraseña luego haciendo una instancia de mi DAO de empleados hago un nuevo constructor y le envio estos datos al método que tengo en mi DAO.

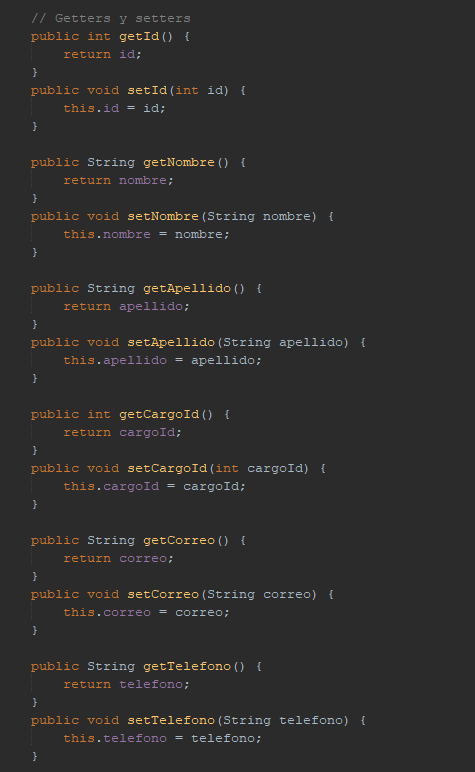
**CODIGO DE FUNCIONALIDAD DEL DAO DE EMPLEADOS**

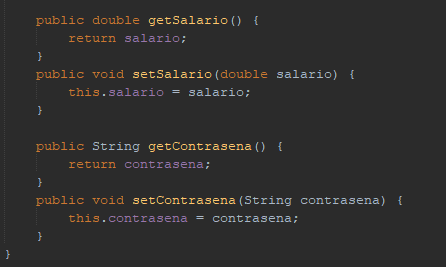


En nuestro caso encriptamos nuestra contraseña usando BCrypt ya que es un método seguro y fácil de implementar, lo primero que hago es una consulta sql para obtener la información del usuario según el correo y si es activo, de ser asi obtengo el hash de la contraseña que esta en nuestra DB, luego usando el método checkpw() de nuestra librería de BCrypt hago la validación de que la contraseña que escribió el usuario sea la misma del hash, de ser asi hago un nuevo DTO con los datos del empleado y lo retorno, de lo contrario muestro un mensaje de error.

**CODIGO DE FUNCIONALIDAD DEL DTO DE EMPLEADOS**

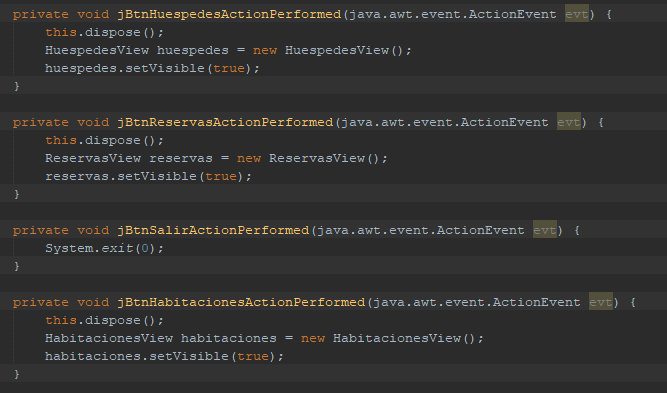






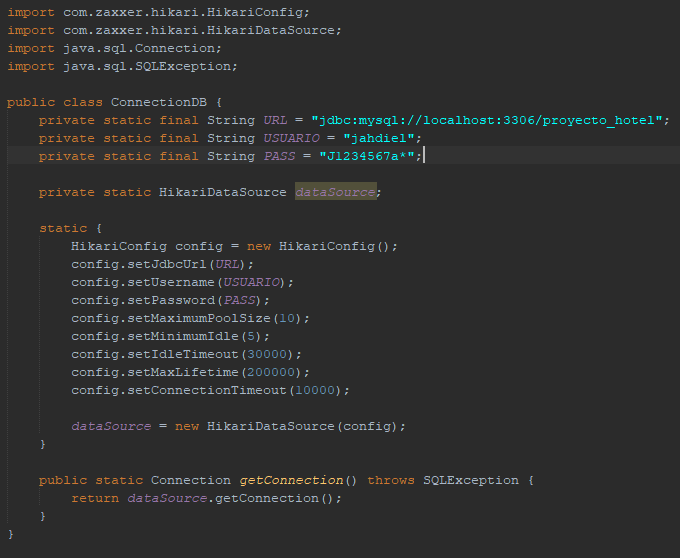
Este es el código del DTO para nuestra tabla de empleados, con cada uno de sus métodos getter y setters.

**CODIGO DE FUNCIONALIDAD DEL MENU DE INICIO**



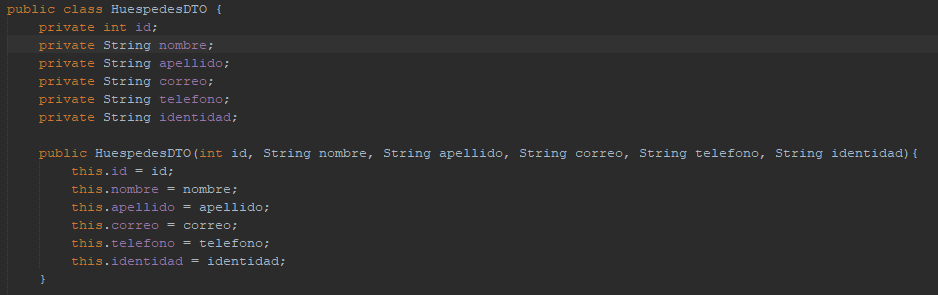
Esta es nuestra vista de Menu, la cual contiene 4 botones, 3 para acceder a nuestras pantallas de huéspedes, habitaciones y reservas y otro para salir del programa

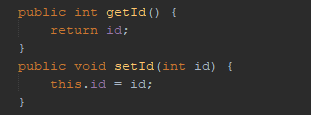
**CODIGO DE FUNCIONALIDAD DE LA CONEXIÓN CON LA BASE DE DATOS**

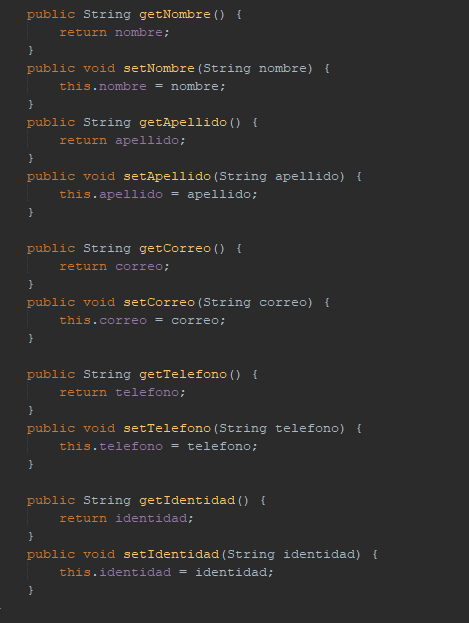


Para conectarnos a nuestra base de datos creamos 3 variables estáticas para la URL de conexión, el usuario y la contraseña. Usamos Hikari para manejar un pool de conexiones y creamos un dataSource con algunas configuraciones como la cantidad de conexiones al mismo tiempo máximas, el tiempo de vida de cada conexión y un tiempo de espera para la conexión.

**CODIGO DE FUNCIONALIDAD PARA DTO DE HUESPEDES**

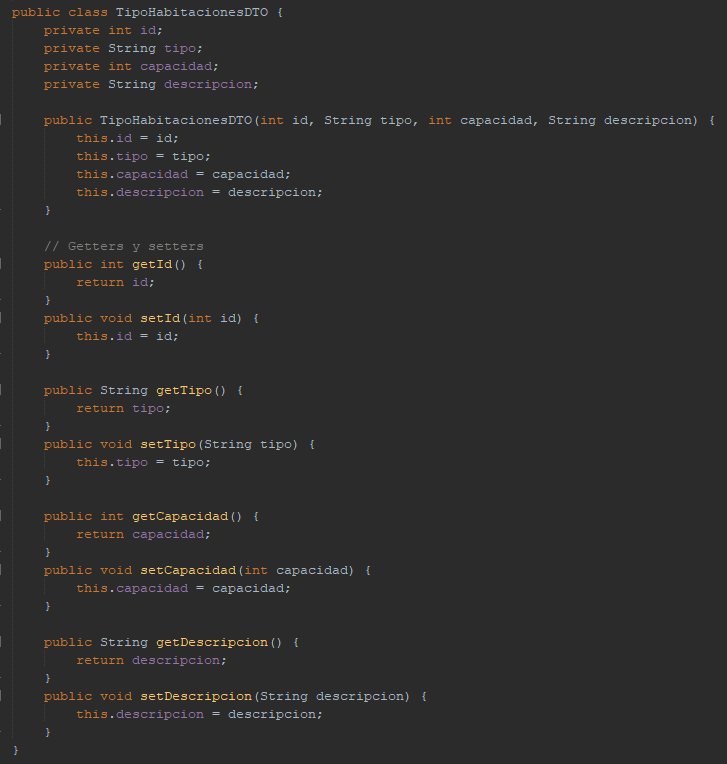






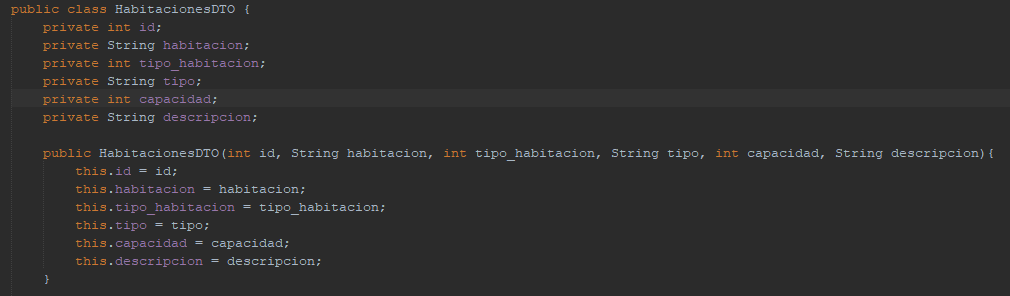
Este es el código que uso para el DTO de mi tabla de huéspedes, en el cual hago un constructor con los campos de la tabla y métodos setter y getter para cada uno de los campos.

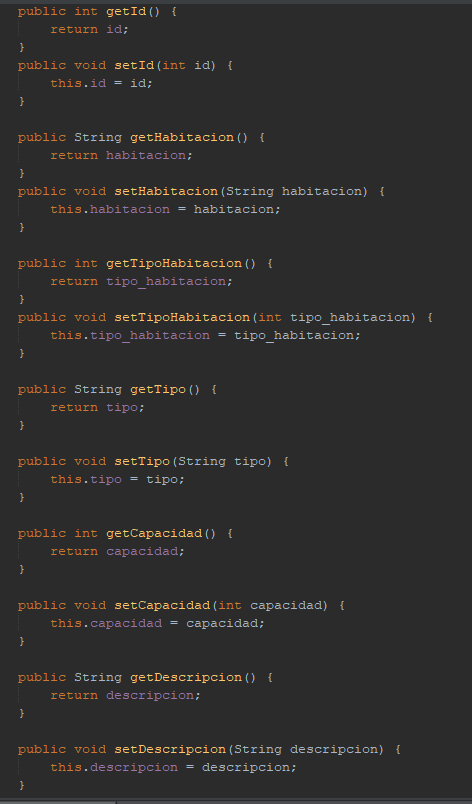
**CODIGO DE FUNCIONALIDAD PARA DTO DE TIPOS DE HABITACIONES**

****

Codigo del DTO para la tabla de Tipos de Habitaciones que se encuentra en nuestra BD.

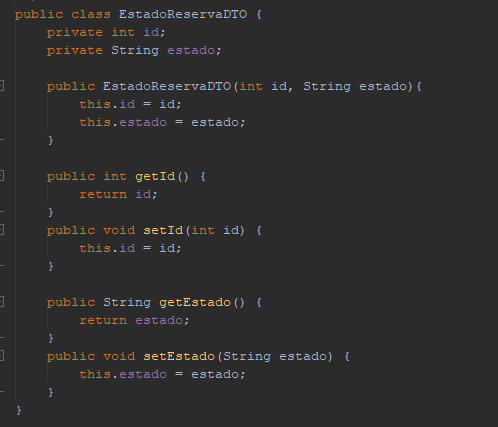
**CODIGO DE FUNCIONALIDAD PARA DTO DE HABITACIONES**





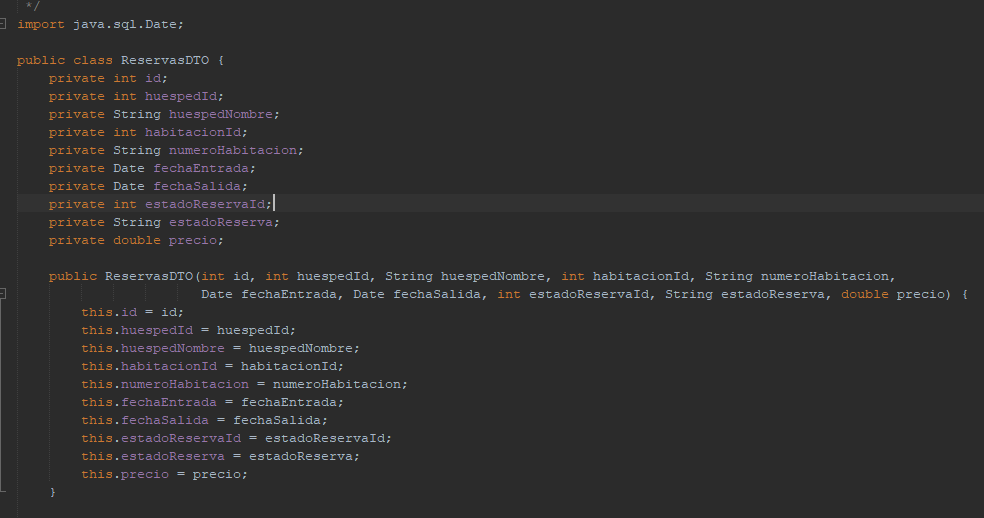
Este DTO lo cree en base a mi tabla de habitaciones, lo cree para poder obtener las habitaciones en mi DAO y poder usarlo para las reservas.

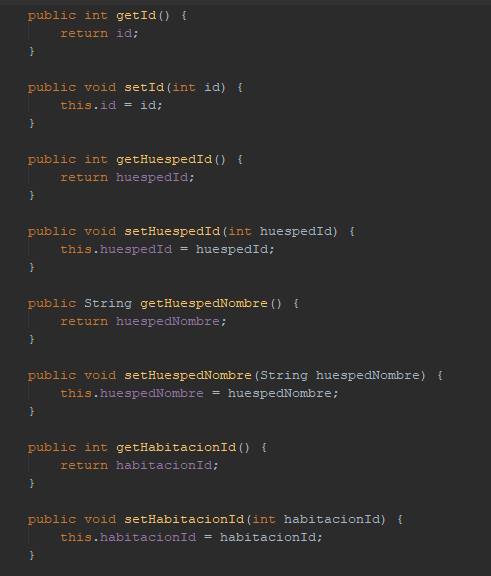
**CODIGO DE FUNCIONALIDAD PARA DTO DE ESTADOS DE RESERVA**



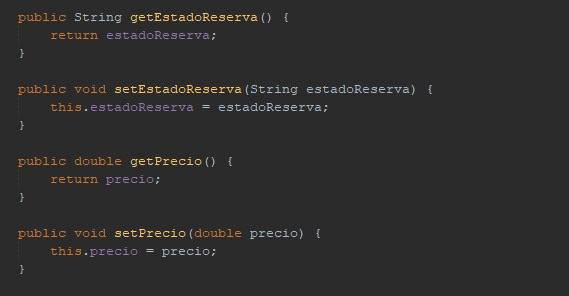
Este DTO también creado en base a su tabla en la BD y que se usa de ayuda para las reservas.

**CODIGO DE FUNCIONALIDAD DE DTO DE RESERVAS**



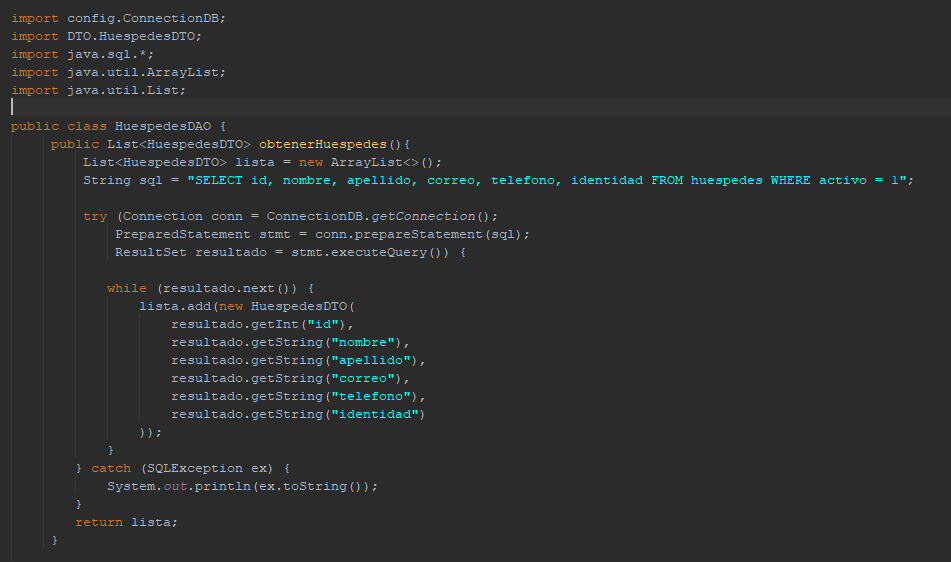




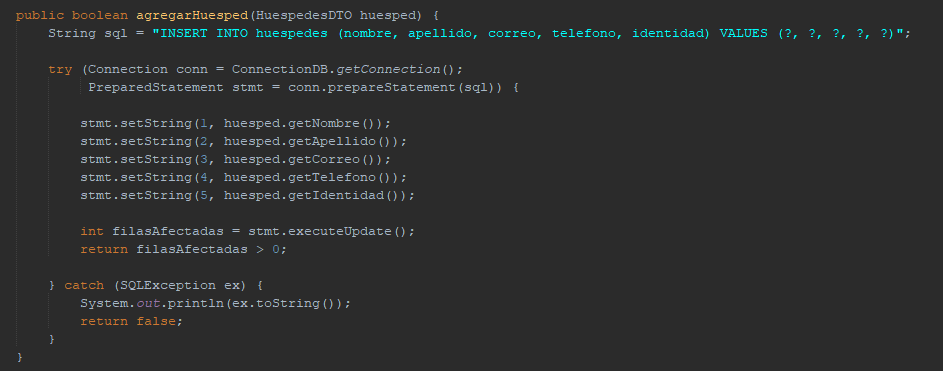


DTO creado para la facilitación de inserción de datos en la tabla de reservas, con sus métodos getters y setters para mejor manejo en las funcionalidades del archivo DAO

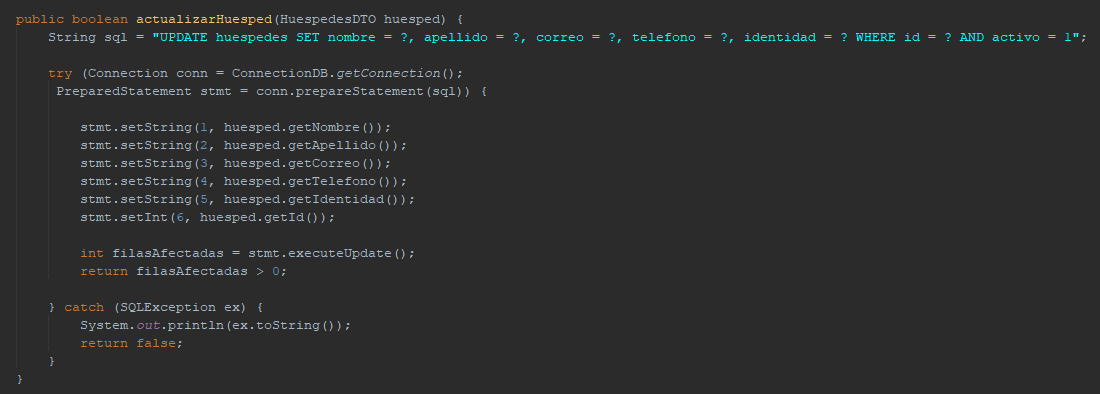
**CODIGO DE FUNCIONALIDAD DEL DAO DE HUESPEDES**



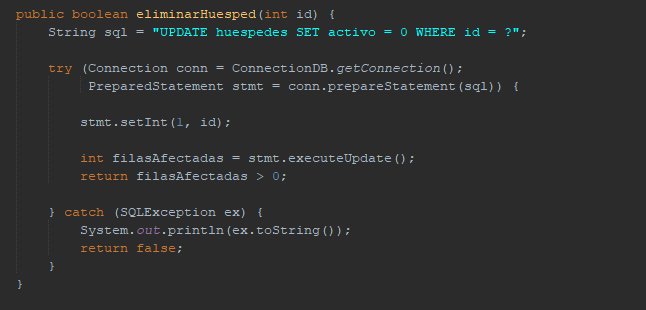
Primero comenzaremos con la función de obtener huéspedes, en esta función creamos una lista vacia que almacenara datos de tipo DTO de huéspedes, luego en una variable guardamos el query que ejecutaremos a través de una instancia de conexión, cuando obtenemos el resultado de cada uno de los datos existentes en la tabla los almacenamos en la lista para luego mostrarlo en la vista de HuespedesView



Para agregar un huésped, primero creamos el query que enviaremos a la BD a través de una instancia de conexión, esta función obtiene de sus parámetros un objeto de datos de tipo HuespedesDTO y accedemos a cada uno de los datos usando las funciones getter y asignamos esos datos al query que ejecutaremos, la función nos retornara un true si se hizo el insert en base de datos

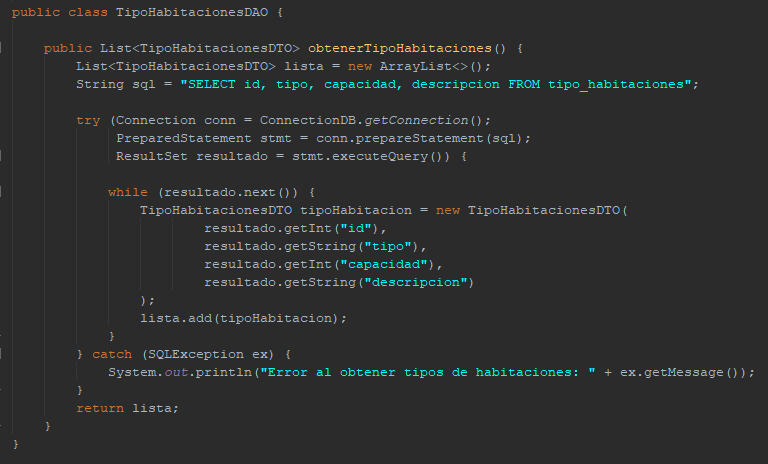


Para actualizar el huésped creamos el query de actualización y lo guardamos en una variable, luego creamos una cadena de conexión, obtenemos los datos del huésped a través de los parámetros y luego asignamos cada uno de esos datos al query para luego ejecutarlo, si se hizo una modificación en alguna de las filas la función devuelve true



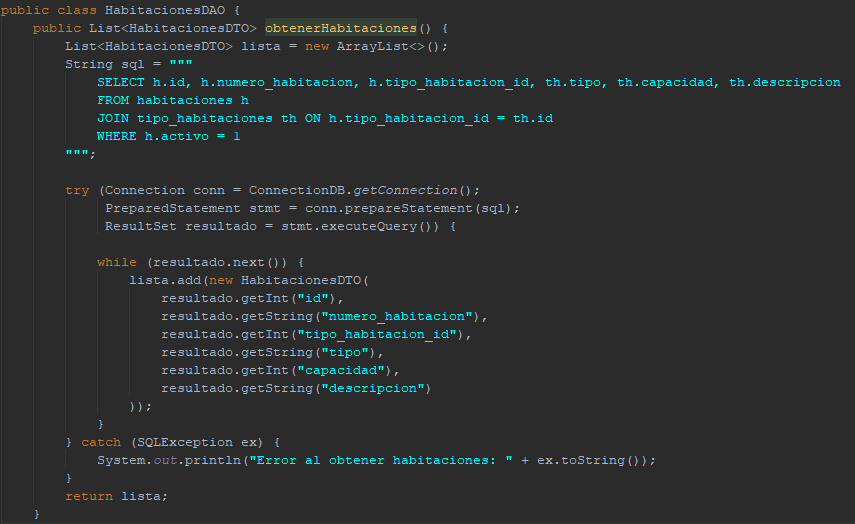
Parecido a la función de actualizar, solo que en este obtenemos solamente el id del huésped que queremos eliminar, realmente lo que se hace es que se cambia el estado de activo a 0 para hacer un borrado lógico y que siempre mantengamos esos datos en nuestra BD

**CODIGO DE FUNCIONALIDAD DEL DAO DE TIPOS DE HABITACIONES**

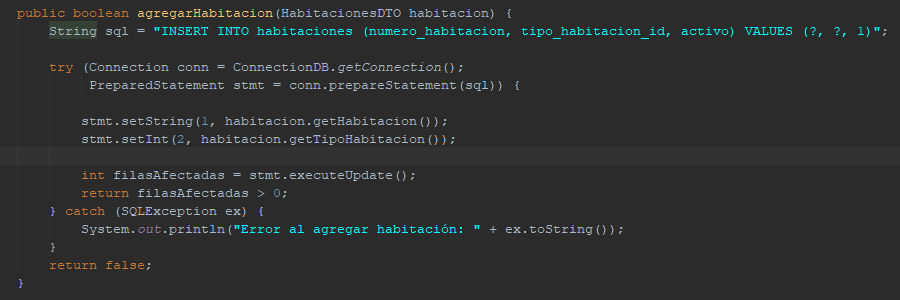


Esta es mi función para obtener los tipos de habitaciones que almaceno en mi BD, obtengo toda esta información ya que la mostrare en un combobox del formulario de Habitaciones.

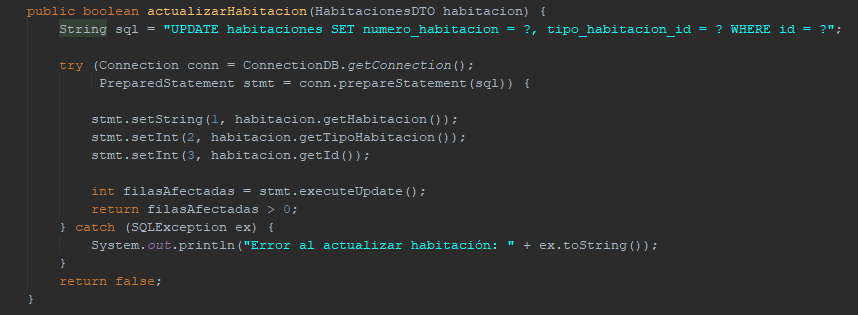
**CODIGO DE FUNCIONALIDAD DEL DAO DE HABITACIONES**



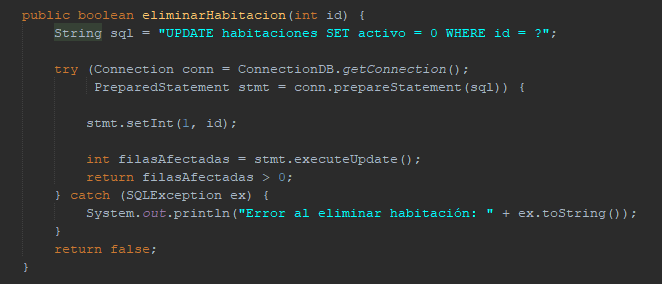
Lo que hacemos en nuestro método es guardar en una variable el query para obtener las habitaciones disponibles (activo = 1) junto con la información de su tipo de habitación gracias a un join y según cada dato que obtenemos los guardamos en una lista de tipo HabitacionesDTO para luego mostrarlo en el formulario de Habitaciones y en un combobox de la tabla de reservas

****

Esta es nuestra función para agregar una habitación, en donde creamos el query para el insert y los datos que obtenemos del DTO que esta como parámetro, lo pasamos al query, luego si la condición en donde las filas afectadas son mayores de 0 se cumple, es porque se inserto la habitación correctamente.

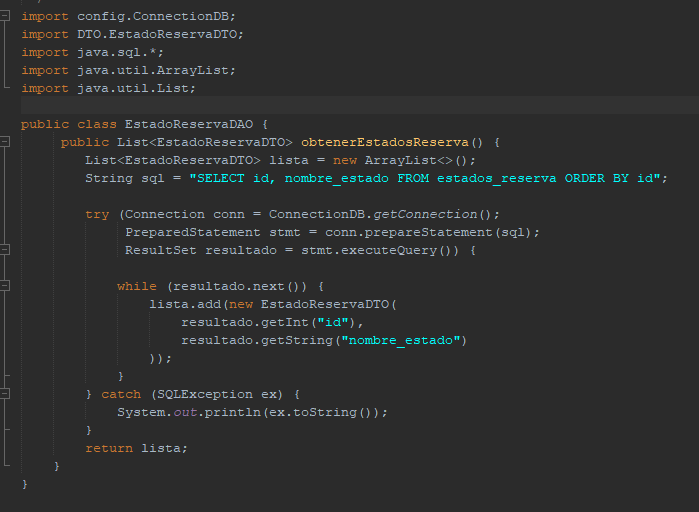


Esta función para actualizar las habitaciones lo que hacemos es guardar nuestro query de update en una variable a la cual le pasamos de parámetro el id de la habitación que se va a cambiar y su información, luego si hay alguna fila afectada es porque se actualizo correctamente nuestra habitación



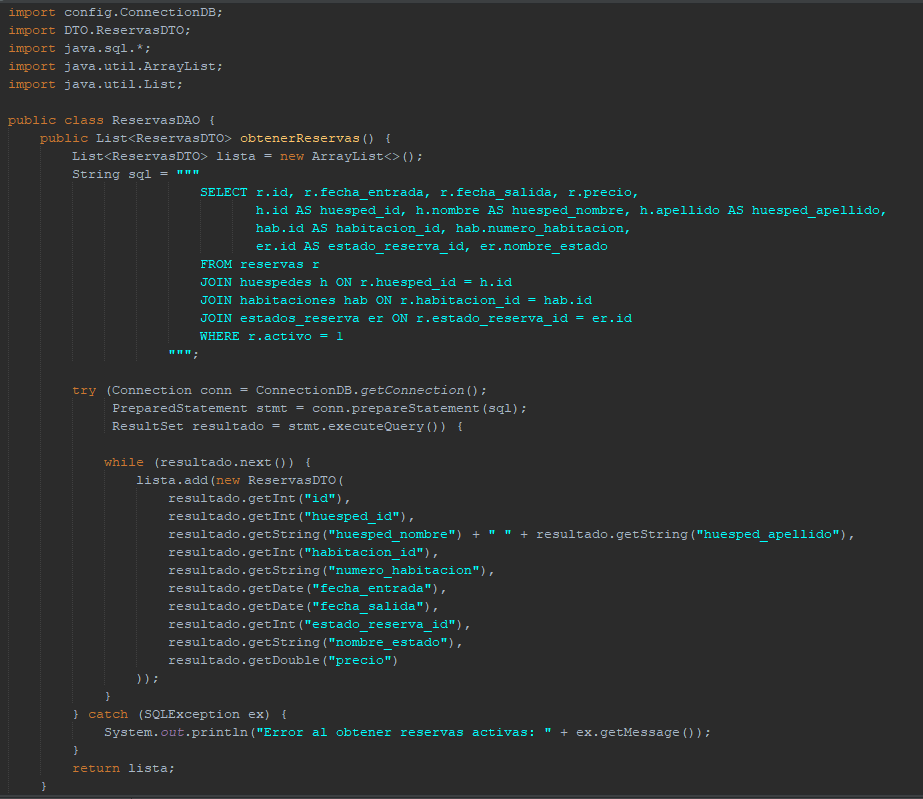
Nuestra función de eliminar donde obtenemos como parámetro el id de la habitación que deseamos eliminar, luego a nuestra consulta que es un update para borrado lógico le enviamos el id que recibimos, igualmente si hay filas afectadas es porque todo se ejecuto correctamente.

**CODIGO DE FUNCIONALIDAD DEL DAO DE ESTADOS DE RESERVA**

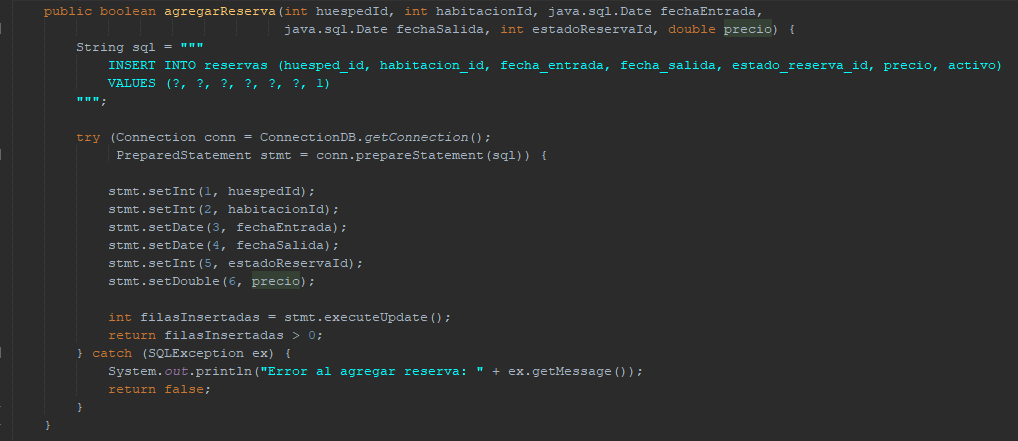


En este código aplicamos lo mismo que hicimos en el de obtener habitaciones y también lo usamos para un ComboBox en la pantalla de ReservasView

**CODIGO DE FUNCIONALIDAD DEL DAO DE RESERVAS**



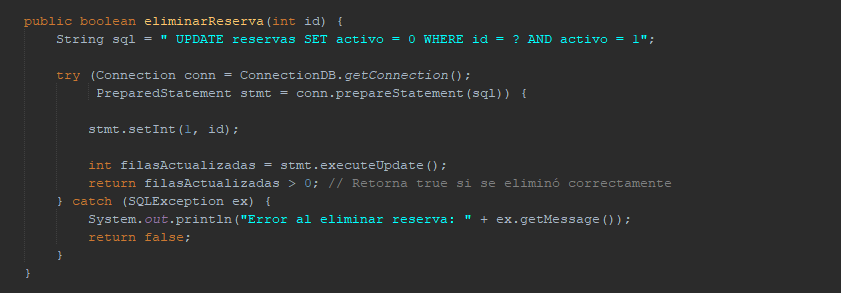
Lo que hago en mi método de obtenerReservas() es que en una variable guardo un query que es para obtener todos los datos de la reserva, incluyendo datos que vienen de otras tablas que se relacionan con la misma, luego hago una instancia de conexión y ejecuto ese query, los datos que me regresa los voy guardando en una lista tipo ReservasDTO



Para agregar una reserva lo hice de una manera distinta, para mostrar diferentes opciones de como se puede hacer, la única diferencia aquí es que envio todos los parámetros directamente al método, luego se crea el query y agrego cada uno de los datos que obtengo de los parámetros, si todo funciona retorna un true

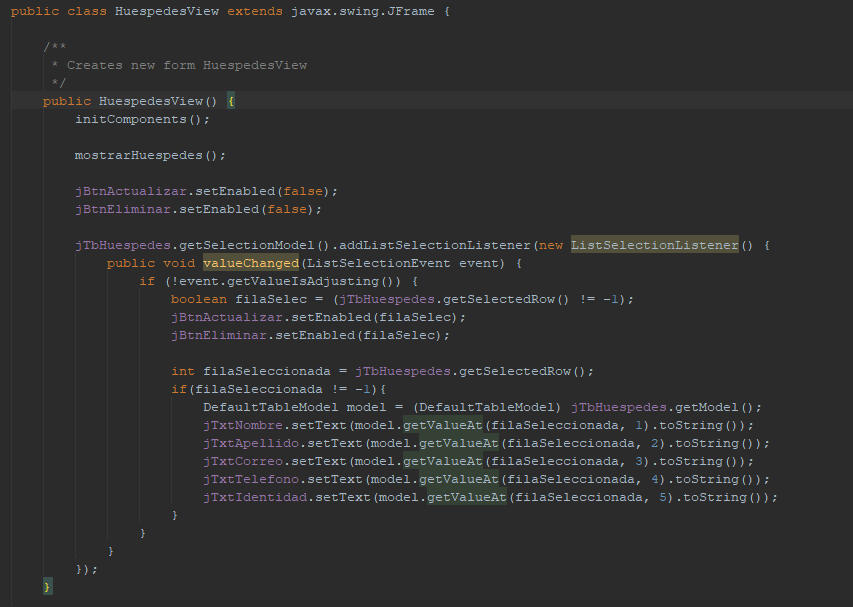


Para actualizar la reserva hago prácticamente lo mismo, lo único que cambia es el query y también que envio un dato mas que es el Id de la reserva que quiero actualizar

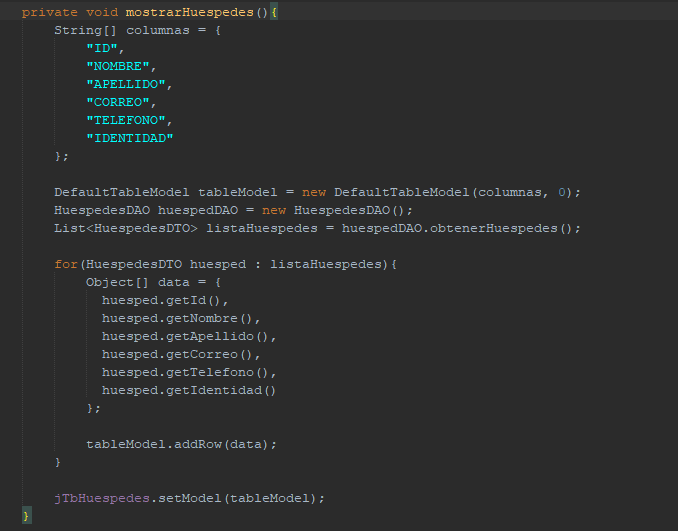


Para eliminar la reserva hago lo mismo que en actualizar, solo que aquí envio como parámetro el Id de la reserva y actualizo el campo de activo a 0 para que sea un borrado lógico y que el registro se siga manteniendo pero no mostrando

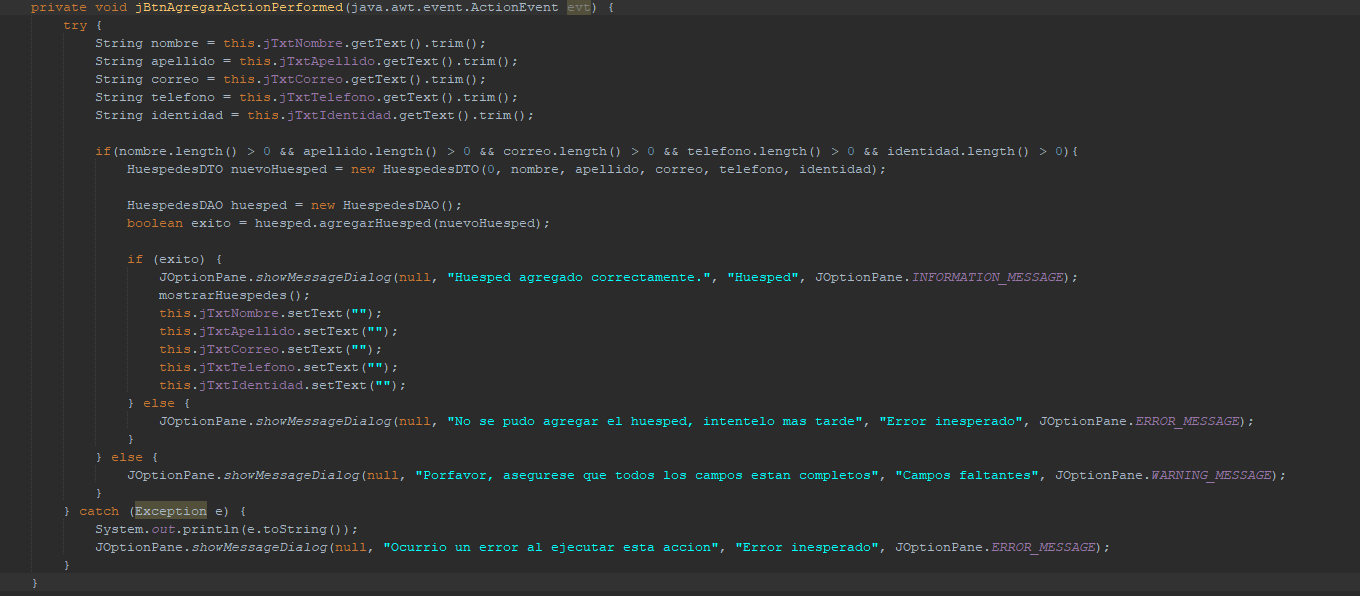
**CODIGO DE FUNCIONALIDAD DE HUESPEDESVIEW**



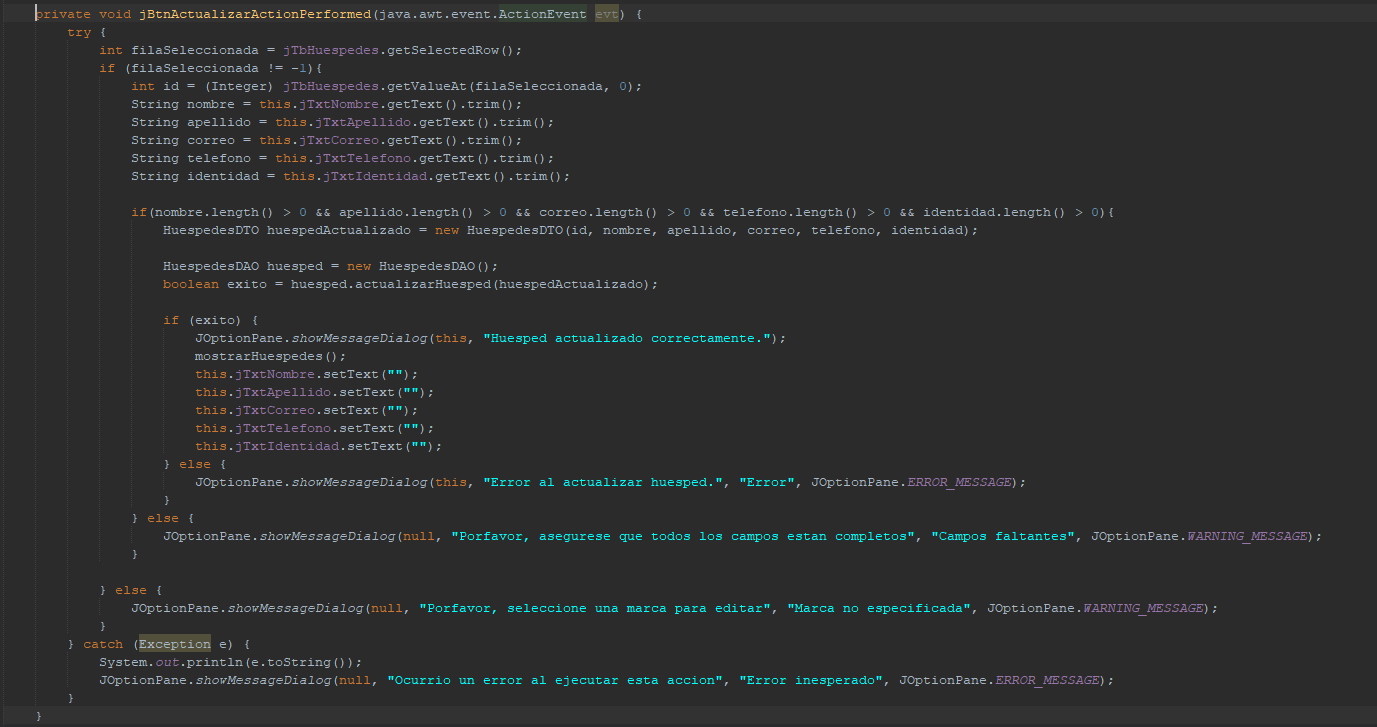
Primero, al momento en que se inicializa la vista de Huespedes, mandamos a llamar un método llamado mostrarHuespedes() y los botones que usamos para actualizar y eliminar están deshabilitados. Tambien tenemos un addListSelectionListener el cual funciona para rellenar cada uno de los campos del formulario cuando seleccionamos un fila con datos de nuestro JTable de huéspedes.



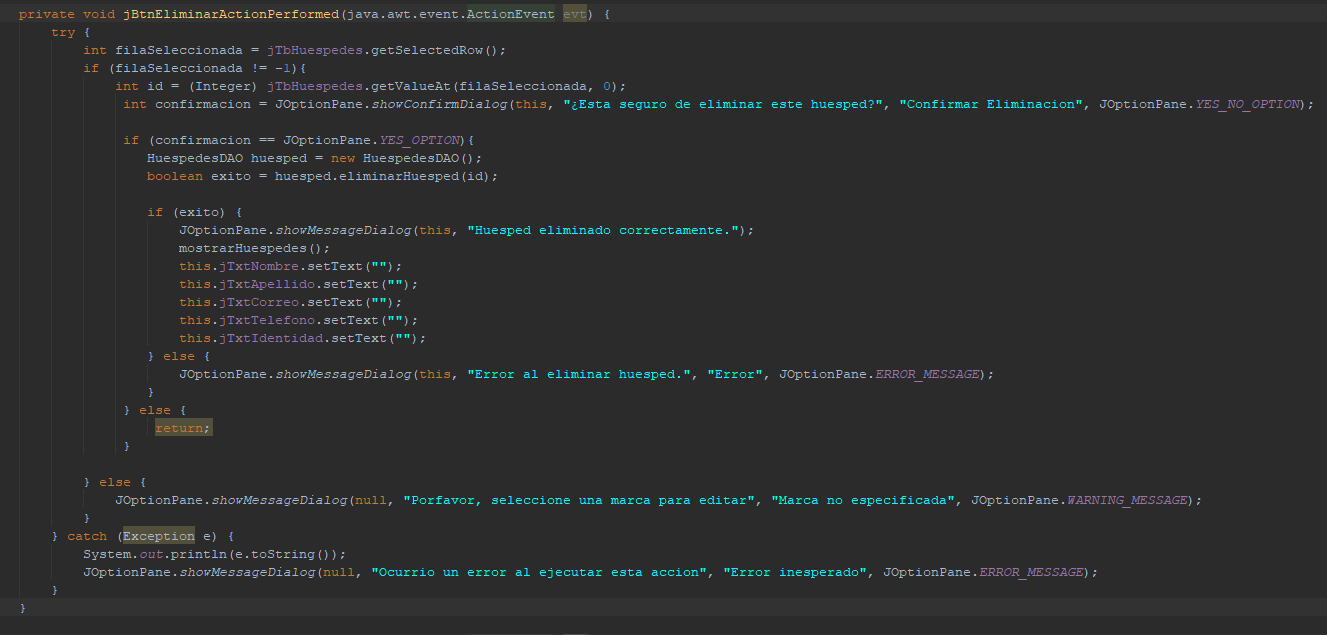
Nuestro método de mostrarHuespedes() lo que hace es que crea un modelo de tabla conforme a la cantidad de columnas que obtenemos de nuestro método de obtenerHuespedes() de nuestro DAO y luego setea la tabla existente en el JFrame con las columnas y datos que obtuvimos



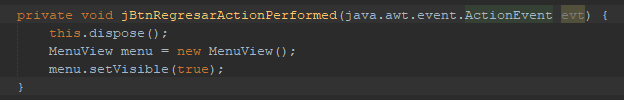
Para agregar el botón lo que hago es obtener todos los datos de cada uno de los campos y con trim() elimino espacios adicionales que existan, luego valido que todos los campos esten llenos, de no ser asi muestro una advertencia de que se tienen que llenar los campos faltantes, si están todos llenos hago una instancia de mi DTO donde le paso todos los datos que obtuve de los campos de texto, luego hago una instancia de mi DAO para enviarle ese objeto de tipo DTO de huéspedes, valido si el método me retorno un true, de ser asi muestro un mensaje de que el huésped fue agregado correctamente y limpio los campos del formulario



Para actualizar primero revisamos que se haya seleccionado una fila del JTable seleccionada, de ser asi obtenemos todos los datos de los TextBox, validamos que esten llenos, creamos un objeto de tipo HuespedesDTO y se lo enviamos al método de actualizarHuespedes de nuestro DAO, si todo fue exitoso mostramos un mensaje que nos dice que se actualizo el huésped correctamente, de caso contrario mostramos un mensaje de error, igual si no se selecciono una fila del JTable, también mostramos un mensaje de error

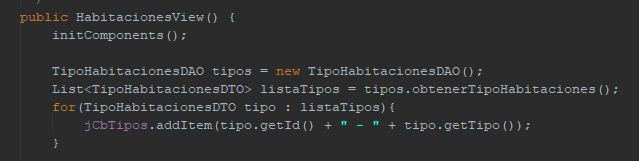


Para eliminar hacemos lo mismo que en actualizar, con la única diferencia que aquí solo obtenemos el ID del huésped que se seleccionó en el JTable, antes de eliminar primero le preguntamos al usuario si esta seguro de eliminar, de ser asi enviamos el id al método DAO para eliminar, si todo fue correcto nos muestra un mensaje de que fue eliminado correctamente



En este botón lo único que hacemos es cerrar la vista de Huespedes y mostrar el Menu nuevamente

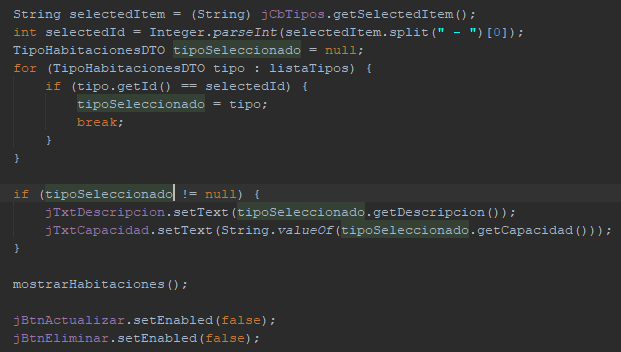
**CODIGO DE FUNCIONALIDAD DE HABITACIONESVIEW**



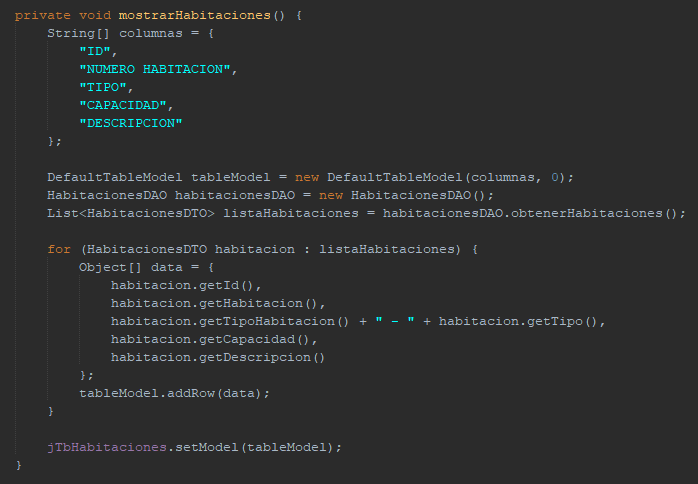
Cuando se ejecuta nuestra vista, lo primero que hacemos es rellenar un ComboBox que es el que contiene los tipos de habitaciones, llenamos cada item con el id y tipo de habitación, concatenándolo en un mismo string



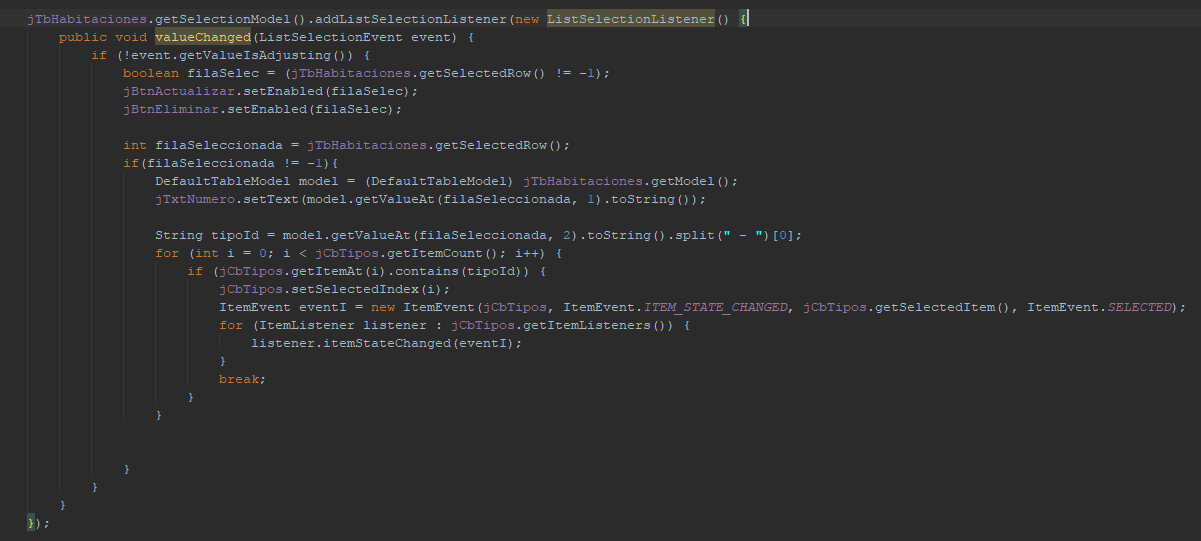
Luego creamos un ItemListener para este ComboBox, lo que hacemos es que cuando una opción de nuestro ComboBox cambie, también cambien los textos de nuestros jTextField para Cantidad y Descripcion, haciendo de esta manera que la información sea dinámica



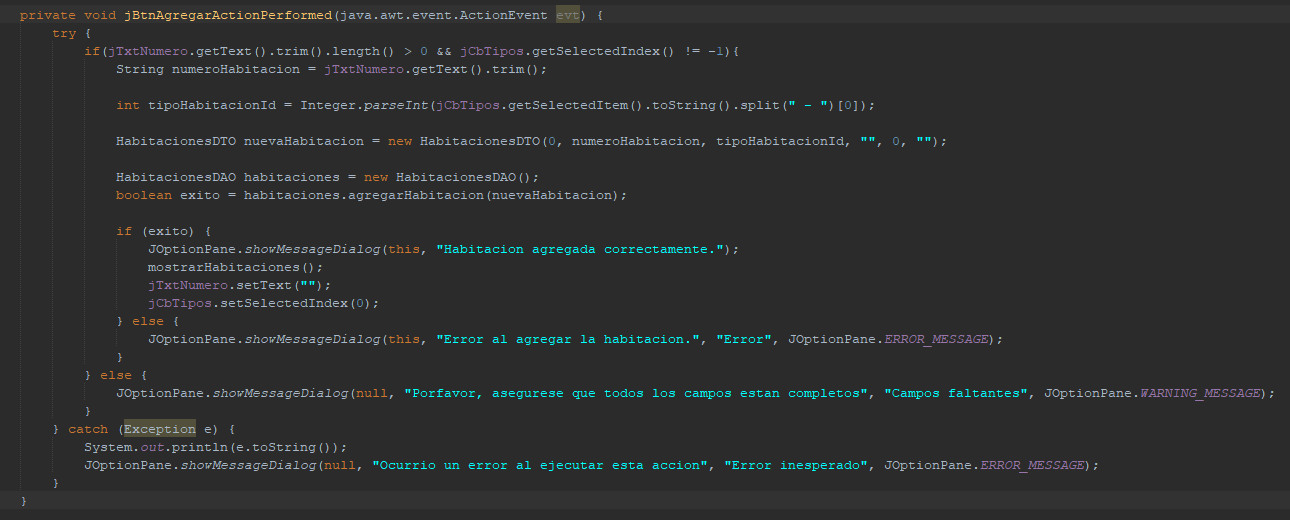
Hacemos lo que hace nuestro ItemListener una vez cuando se ejecuta la vista, esto ayuda a que la primera vez que se abra nuestro formulario ya estén los jTextField llenos y no tenga que esperar a un cambio del ComboBox para llenarse, así mismo mando a llamar una función que me permite mostrar la tabla con las habitaciones y establezco que mis botones de actualizar y eliminar no estén activos.



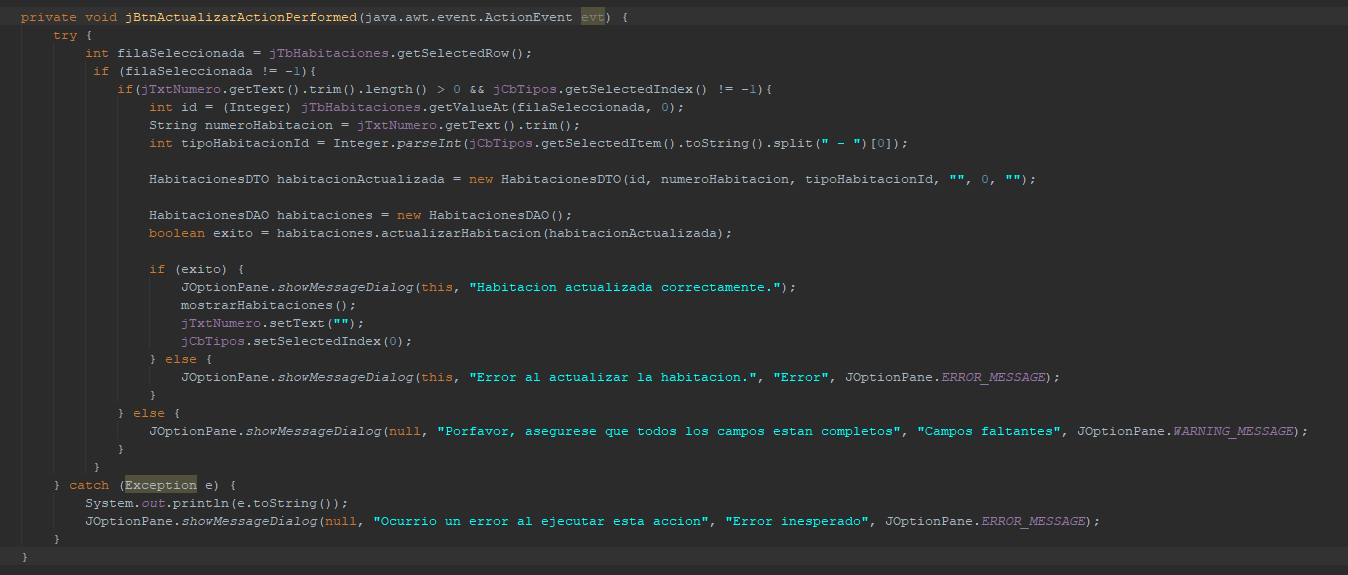
Este es mi método para mostrar Habitaciones, en donde creo un modelo nuevo de la tabla y asi la relleno con la información que obtengo de la función obtenerHabitaciones de mi DAO.



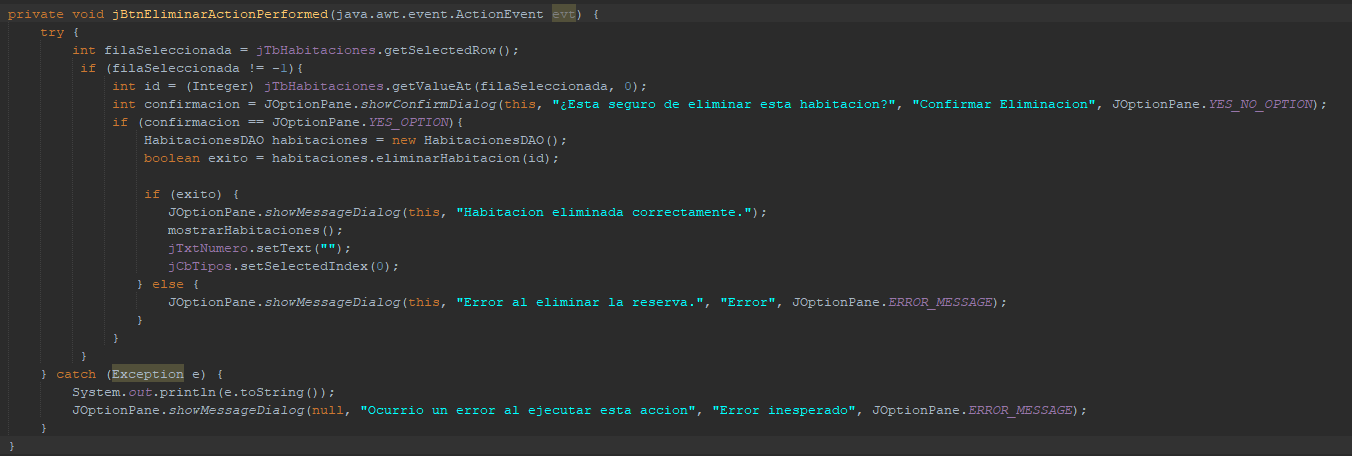
Lo que hago después es un método que obtiene la información de alguna fila que se selecciono del formulario y lleno los campos, en el caso de mi ComboBox lo que hago es primero obtener el id del tipo de habitación que se selecciono, luego creo todo un evento pasarle este Id a mi ComboBox, cambiando el estado con el item que se selecciono y mandando a llamar al ItemListener notificándole que sucedió un cambio.



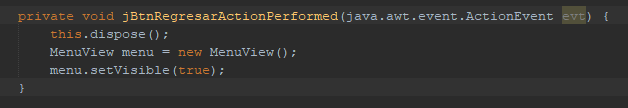
Mi botón para agregar Habitaciones valido que tanto mi jTextField para el numero de habitación y mi combobox no esten vacíos, luego obtengo el id del tipo de habitación y lo paso a un nuevo constructor del DTO de habitaciones y mando a llamar y ejecutar mi función para agregar habitaciones, si todo salió bien muestro un mensaje de éxito y limpio los campos del formulario y recargo la tabla, de lo contrario muestro mensajes de error.



Mi botón para actualizar, el cual se activa cuando se selecciona una fila de mi tabla primero valida que los campos esten llenos, luego de eso obtiene el ID de la fila seleccionada junto con los otros campos, los pasa a un constructor del DTO de habitaciones y ejecutamos el método para actualizar habitaciones, si todo sale bien mostramos un mensaje de éxito y recargamos la tabla, de lo contrario mostramos mensajes de error.

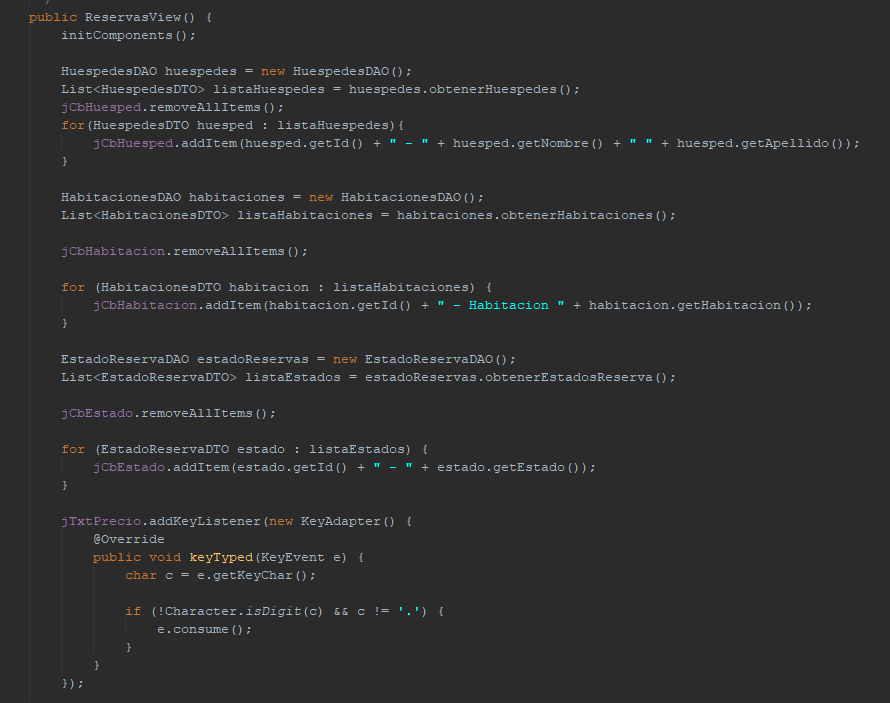


En nuestro método para el botón de eliminar lo que hacemos es obtener el ID de la fila que se selecciono, después a través de un jDialog hacemos la pregunta al usuario si de verdad quiere eliminar ese registro, de ser asi llamamos a nuestro método de eliminar habitación, si el borrado lógico se ejecuta bien mostramos un mensaje de éxito, de lo contrario un mensaje de error.



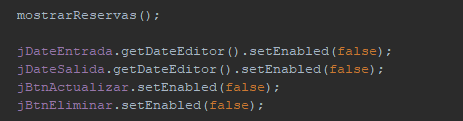
Nuestro botón de regreso, que nos permite regresar a nuestro en menú en caso que lo deseemos.

**CODIGO DE FUNCIONALIDAD DE RESERVASVIEW**

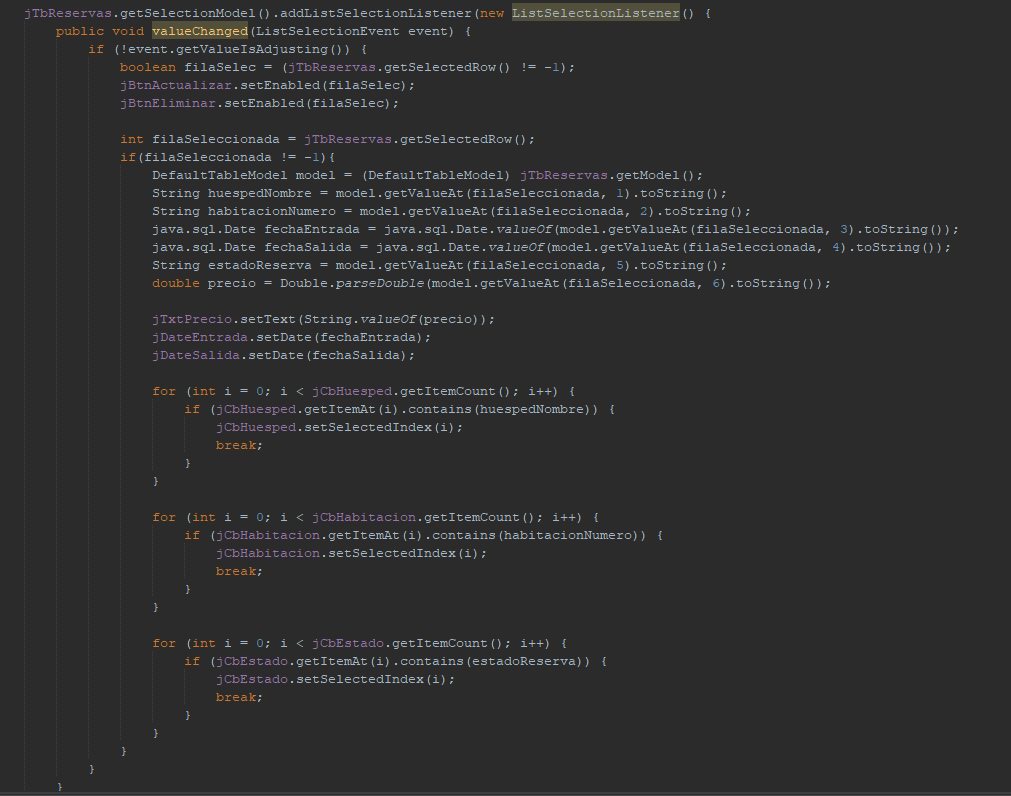


Lo que hago primero es que lleno todos los ComboBox que hay en mi formulario, en mi caso son 3, huéspedes, habitaciones y estado de reserva, asi que hago una instancia del DAO de cada uno y obtengo los datos en una lista, luego recorro esa lista para agregar cada item al ComboBox.

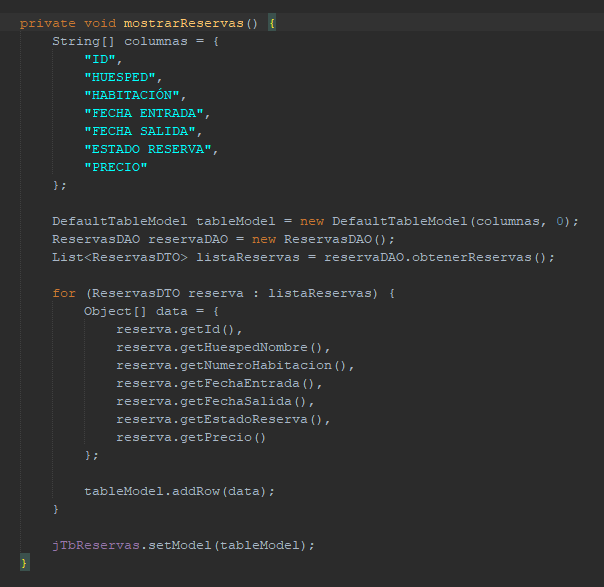
También agrego un KeyListener a mi Textbox de precio, para que solo acepte números y un punto decimal, para no tener problemas a la hora de ingresarlo en la BD



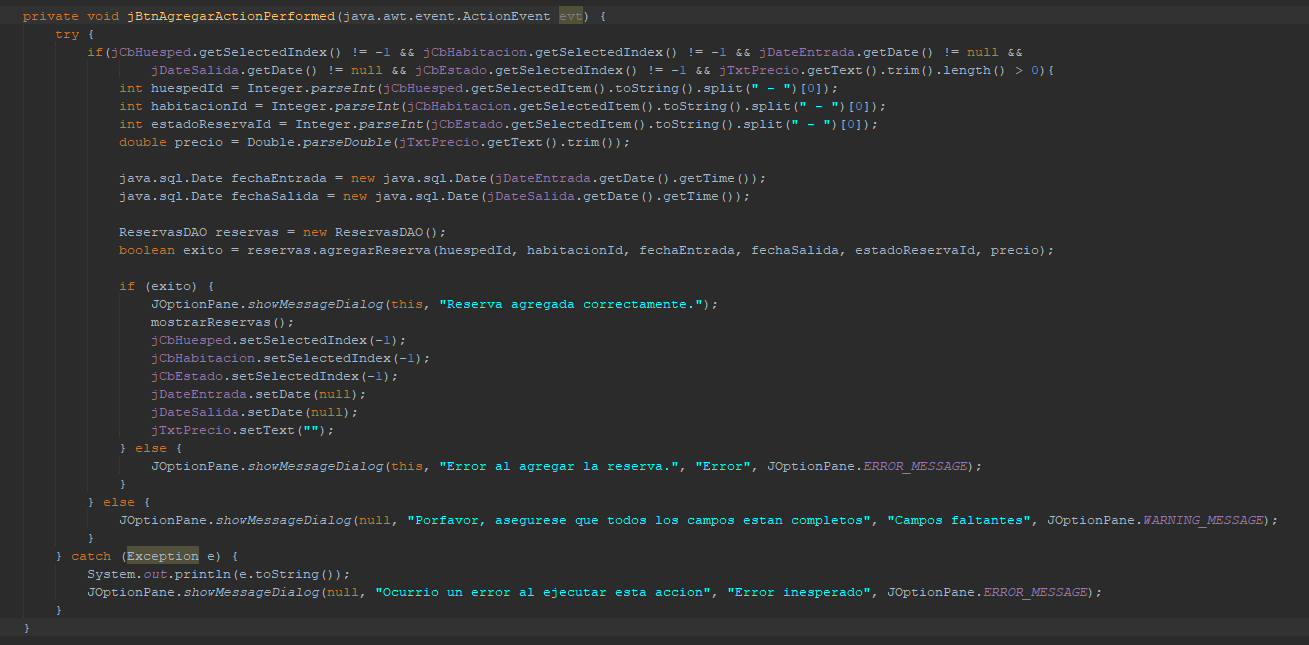
Luego mando a llamar una función para mostrar las reservas y también deshabilito los textbox para las fechas, para que no se ingresen manual, sino solamente desde el calendario que traen, dejo deshabilitado también los botones de actualiza y eliminar a la hora de que carga el formulario



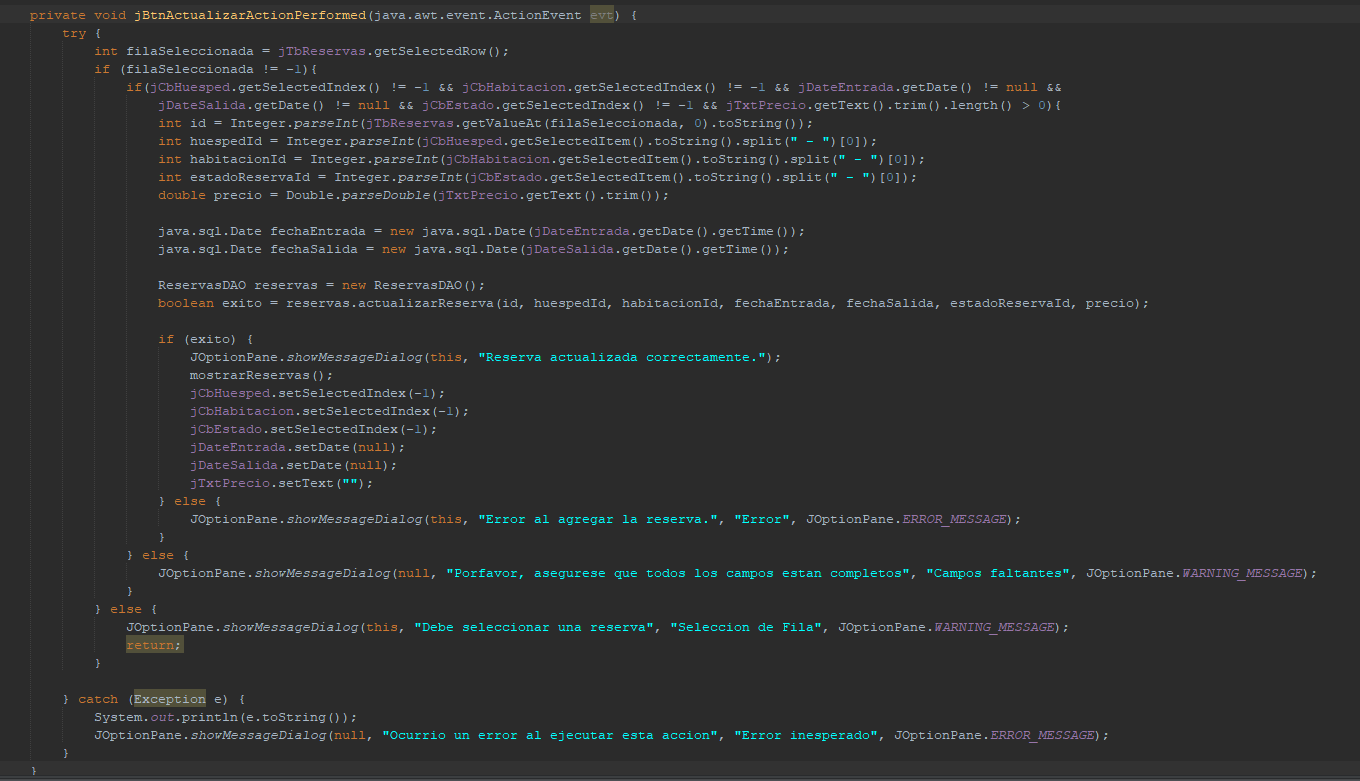
Agrego un ListSelectionListener asi como en la tabla de Huespedes para llenar el formulario con los datos del JTable y de esta manera poder actualizar o eliminar el registro que yo deseo.



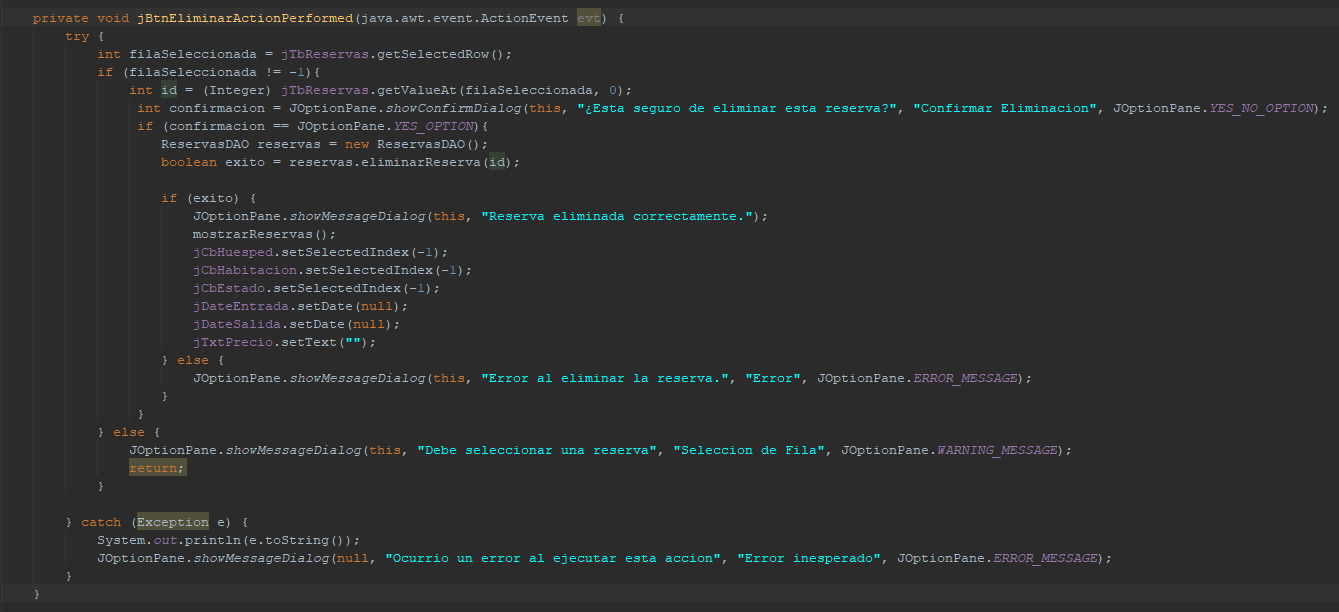
La función de mostrarReservas() con la que obtengo todas las reservas y creo un modelo de tabla según las columnas que obtengo



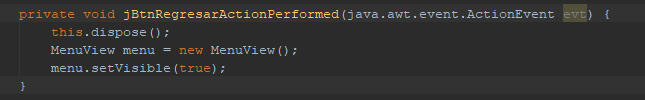
Boton para agregar reservas, en donde valido que cada uno de los campos no contenga datos vacíos o nulos, luego de eso, hago un Split y un seteo a los datos de los ComboBox para obtener el ID del item que se selecciono, también hago un seteo a los textbox de fechas y por ultimo envio todos estos datos como parámetros a mi método en el DAO para insertar el nuevo registro, si todo sale bien muestro una alerta que dice que se registro la reserva correctamente



Para actualizar hacemos casi lo mismo, con la única diferencia de que validamos que hayamos seleccionado una fila y que también obtenemos el id del registro que se selecciono, luego todo eso lo mandamos al método para actualizar y si se completó mostramos una alerta, de lo contrario mostramos un error, todo siempre usando los DialogMessage.

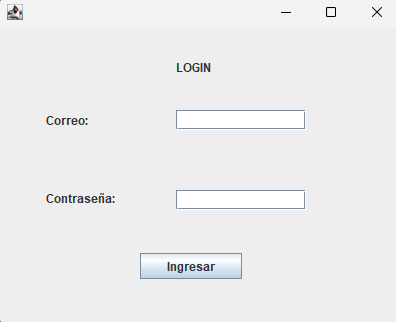


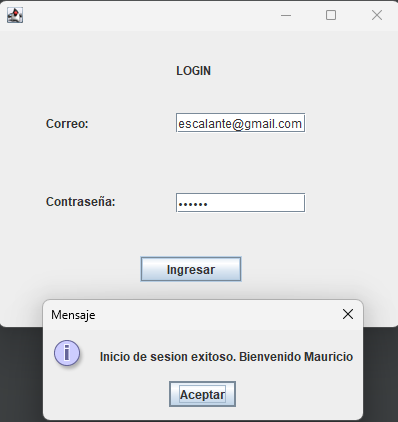
Para eliminar la reserva hacemos lo mismo que en actualizar, solo que aquí solo enviamos el id de la reserva que queremos eliminar, si todo salió correctamente mostramos un mensaje de alerta, sino un mensaje de error



Boton de regresar que hace lo mismo que lo de Huespedes, cierra el JFrame actual y luego muestra el de Menu

**CAPTURAS DE PANTALLA DEL PROGRAMA**



****

