Documento de Sustentación del Código: `FrmLogin`

Introducción

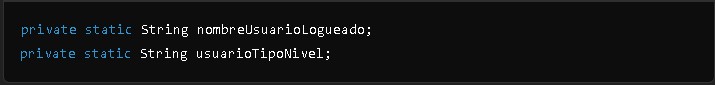
Este documento detalla el funcionamiento y propósito del código de la clase `FrmLogin`, una ventana de autenticación en una aplicación de escritorio Java. La clase extiende `javax.swing.JFrame` para ofrecer una interfaz gráfica que permite a los usuarios iniciar sesión en el sistema. Además, incluye características visuales y funcionales como placeholders, ocultamiento de contraseña, e integración con la base de datos para autenticar al usuario.

Objetivo

El objetivo principal de esta clase es ofrecer una interfaz de usuario intuitiva para el inicio de sesión, validar las credenciales del usuario, y permitir o denegar el acceso según su estado y nivel de permisos.

Estructura del Código

1. Variables estáticas



- nombreUsuarioLogueado: Almacena el nombre del usuario autenticado.

- usuarioTipoNivel: Almacena el nivel de permisos del usuario autenticado.

Estas variables permiten que otras partes de la aplicación accedan a la información del usuario que ha iniciado sesión, lo que puede ser útil para gestionar permisos y funciones de manera dinámica.

2. Constructor: `FrmLogin()`



- setUndecorated(true): Quita los bordes decorativos de la ventana.

- initComponents(): Inicializa los componentes de la interfaz, generados automáticamente por el IDE.

- setSize(400, 550): Establece el tamaño de la ventana.

- setResizable(false): Impide que la ventana sea redimensionada.

- setLocationRelativeTo(null): Centra la ventana en la pantalla.

- setShape(...): Redondea las esquinas de la ventana para un diseño más moderno.

El constructor también define los fondos y placeholders para los campos de usuario y contraseña, mejorando la apariencia del formulario de login.

3. Métodos Principales

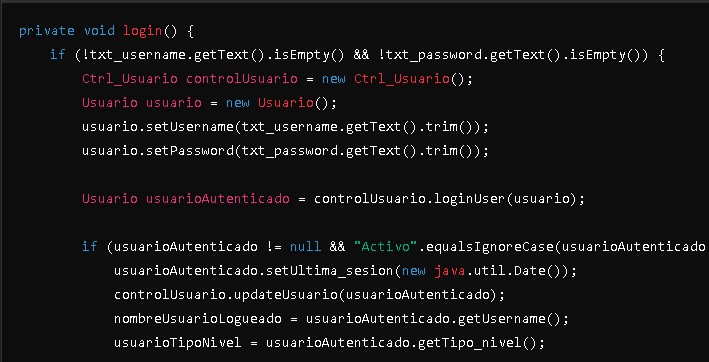
a) Icono personalizado del JFrame



Este método sobrescribe el icono predeterminado del JFrame y lo reemplaza por uno personalizado (`favicon.png`).

b) Método `login()`

Este método gestiona el proceso de autenticación del usuario.



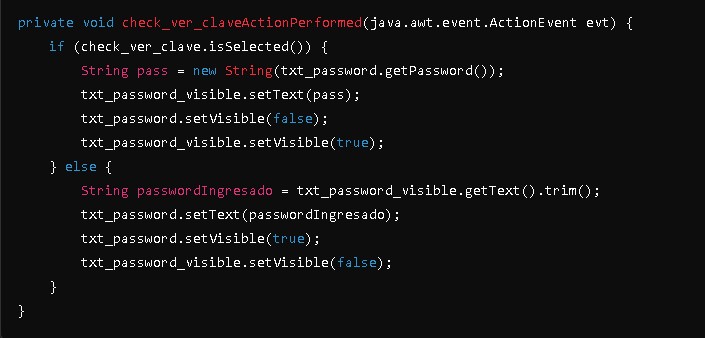
- Verificación de campos vacíos: Se asegura de que el usuario y la contraseña no estén vacíos.

- Autenticación: Utiliza el controlador `Ctrl\_Usuario` para verificar las credenciales del usuario contra la base de datos.

- Actualización de sesión: Actualiza la fecha y hora de la última sesión del usuario autenticado.

- Control de acceso: Redirige al usuario a la interfaz de administrador (`FrmAdmin`) si la autenticación es exitosa, o muestra una alerta personalizada en caso de error o cuenta inactiva.

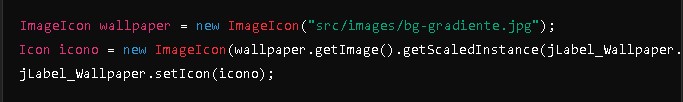
c) Mostrar/Ocultar contraseña



Este método permite alternar entre mostrar la contraseña como texto claro o como caracteres ocultos.

4. Diseño de la Interfaz

- Fondos e imágenes:



La clase utiliza imágenes de fondo y logos para mejorar la apariencia visual de la interfaz.

- Placeholders: 

Los placeholders son sugerencias que se muestran dentro de los campos de texto, proporcionando una mejor experiencia de usuario.

5. Métodos estáticos

- `getNombreUsuarioLogueado()`: Devuelve el nombre del usuario autenticado.

- `getUsuarioTipoNivel()`: Devuelve el tipo de nivel del usuario autenticado.

Estos métodos son importantes para acceder a la información del usuario logueado en otras partes del sistema.

Consideraciones Técnicas

1. Diseño y Responsividad: La ventana está diseñada para ser compacta y centrada, con un estilo moderno gracias a los bordes redondeados y la eliminación de decoraciones predeterminadas.

2. Interacción con la base de datos: El controlador `Ctrl\_Usuario` maneja la autenticación y actualización de los datos del usuario, como la fecha de la última sesión.

3. Manejo de Errores: El código incluye ventanas de alerta personalizadas (`AlertaCuentaInactiva`, `AlertaCamposVacios`) en lugar de los diálogos estándar de Java, ofreciendo una interfaz más consistente y profesional.

-

Conclusión

La clase `FrmLogin` proporciona una solución completa para el inicio de sesión en un sistema basado en Java Swing, con autenticación basada en base de datos, personalización de la interfaz y control de errores. Además, incluye características avanzadas como la alternancia de visibilidad de la contraseña y placeholders, que mejoran la experiencia del usuario final.