**Bibliotecas importadas en el Proyecto**

Este Proyecto Java implementa una interfaz gráfica para un juego de ajedrez utilizando los framework de Swing y AWT. A continuación, explicaré brevemente qué hace cada una de las librerías utilizadas en este código:

1. **java.awt.\*:**
   * **Utilidad**: Proporciona clases para crear interfaces gráficas de usuario (GUI) en Java, incluyendo componentes gráficos como botones, ventanas, etc.
   * **Uso en el Proyecto**: Utilizado para crear y gestionar la interfaz gráfica del juego de ajedrez, incluyendo paneles y componentes gráficos.
2. **java.awt.event.\*:**
   * **Utilidad**: Contiene clases y métodos para manejar eventos generados por componentes de la GUI.
   * **Uso en el Proyecto**: Se utiliza para manejar eventos del ratón y del teclado en las casillas del tablero de ajedrez.
3. **java.awt.image.BufferedImage:**
   * **Utilidad**: Permite la manipulación de imágenes en memoria, incluyendo la manipulación de píxeles individualmente.
   * **Uso en el Proyecto**: Se utiliza para cargar y manipular imágenes de las piezas de ajedrez.
4. **java.io.File:**
   * **Utilidad**: Proporciona clases para manejar archivos y directorios en el sistema de archivos.
   * **Uso en el Proyecto**: Especifica la ubicación de los archivos de imagen de las piezas de ajedrez y maneja la lectura de imágenes desde el sistema de archivos.
5. **java.io.IOException:**
   * **Utilidad**: Maneja errores de entrada/salida (E/S) al trabajar con archivos y flujos de datos.
   * **Uso en el Proyecto**: Captura posibles excepciones que pueden ocurrir al leer las imágenes desde los archivos.
6. **java.util.HashMap:**
   * **Utilidad**: Implementa una estructura de datos de mapa basada en tablas de dispersión. Permite asociar claves con valores y proporciona un acceso rápido a los elementos almacenados.
   * **Uso en el Proyecto**: Mantiene un mapa que relaciona los paneles del tablero de ajedrez con los nombres de las piezas en esas casillas.
7. **java.util.Map:**
   * **Utilidad**: Interfaz para asociar claves con valores. HashMap es una de las implementaciones de esta interfaz.
   * **Uso en el Proyecto**: Utilizado junto con HashMap para crear una estructura de datos que almacena las relaciones entre las casillas del tablero y las piezas en esas casillas.
8. **java.io.FileWriter:**
   * **Utilidad**: Se utiliza para escribir caracteres en un archivo.
   * **Uso en el Proyecto**: Utilizado para escribir datos de juego en un archivo de registro.
9. **java.io.PrintWriter:**
   * **Utilidad**: Proporciona métodos para escribir texto en un archivo.
   * **Uso en el Proyecto**: Se utiliza para escribir texto formateado en un flujo de salida.
10. **java.time.LocalDateTime:**
    * **Utilidad**: Permite obtener la fecha y hora actuales.
    * **Uso en el Proyecto**: Registra la fecha y hora en que se realiza cada movimiento.
11. **java.time.format.DateTimeFormatter:**
    * **Utilidad**: Proporciona métodos para formatear la fecha y la hora.
    * **Uso en el Proyecto**: Formatea la fecha y hora para el registro de movimientos.
12. **javax.imageio.ImageIO:**
    * **Utilidad**: Proporciona clases para leer y escribir imágenes en diferentes formatos.
    * **Uso en el Proyecto**: Lee imágenes de los archivos de las piezas de ajedrez y las carga en el programa para su visualización.
13. **javax.swing.\*:**
    * **Utilidad**: Proporciona componentes y clases para la creación de aplicaciones GUI.
    * **Uso en el Proyecto**: Utilizado para crear la interfaz gráfica del juego de ajedrez, incluyendo ventanas, paneles y botones.
14. **javax.swing.SwingUtilities:**
    * **Utilidad**: Proporciona utilidades relacionadas con la interfaz gráfica de usuario (GUI) de Swing.
    * **Uso en el Proyecto**: Utilizado para realizar operaciones en el hilo de eventos de Swing, como invocar la creación de interfaz de usuario desde un hilo no seguro para GUI.
15. **java.sql.Connection:**
    * **Utilidad**: Proporciona métodos para establecer una conexión con una base de datos.
    * **Uso en el Proyecto**: Se utiliza para establecer una conexión con la base de datos de partidas de ajedrez.
16. **java.sql.DriverManager:**
    * **Utilidad**: Implementa el administrador de controladores JDBC para cargar controladores de bases de datos.
    * **Uso en el Proyecto**: Se utiliza para cargar el controlador de la base de datos.
17. **java.sql.PreparedStatement:**
    * **Utilidad**: Permite ejecutar consultas SQL precompiladas en una base de datos.
    * **Uso en el Proyecto**: Se utiliza para ejecutar consultas SQL preparadas en la base de datos.
18. **java.sql.ResultSet:**
    * **Utilidad**: Representa un conjunto de resultados de una consulta SQL.
    * **Uso en el Proyecto**: Se utiliza para almacenar los resultados de las consultas SQL ejecutadas.
19. **java.sql.SQLException:**
    * **Utilidad**: Maneja excepciones específicas de SQL que pueden ocurrir durante la ejecución de operaciones de base de datos.
    * **Uso en el Proyecto**: Se utiliza para capturar errores de SQL que pueden ocurrir durante la ejecución del programa.
20. **java.util.ArrayList:**
    * **Utilidad**: Implementación de la interfaz List que utiliza un arreglo para almacenar elementos.
    * **Uso en el Proyecto**: Se utiliza para almacenar la lista de movimientos de ajedrez.
21. **java.util.List:**
    * **Utilidad**: Interfaz que representa una lista ordenada de elementos.
    * **Uso en el Proyecto**: Se utiliza como una interfaz para la lista de movimientos del juego de ajedrez.