

2025

Proyecto Chat Java – Cliente y Servidor con Interfaz Gráfica

Documentación técnica y guía de uso – 2º DAM



FlatLaf



eclipse







Cliente-Servidor

Introducción







¿Te imaginas poder montar tu propio chat, moderno y en red, hecho desde cero en Java?

Este proyecto es justo eso: una aplicación completa de **chat multiusuario**, con un **servidor** que controla todo y varios **clientes** que se conectan, chatean en grupo, mandan privados y hasta disfrutan de una interfaz cuidada con toques actuales.

¿Qué incluye este proyecto?

-  **Servidor de Chat:** controla la sala, gestiona quién entra y sale, muestra el log en tiempo real y te permite monitorizar todo desde una ventana muy visual. Puedes ver todos los usuarios conectados y tienes el control total.
-  **Clientes de Chat:** cualquier usuario puede conectarse, ver quién está online, enviar mensajes a todos o abrir chats privados solo con un doble clic. Todo desde una ventana sencilla y atractiva, ¡nada de consolas aburridas!
-  **Chats privados:** abre tantas ventanas privadas como quieras, se mantienen activas y se reabren si te escriben de nuevo.
-  **Interfaz moderna:** gracias a FlatLaf, todo tiene un toque más profesional. Elige modo claro u oscuro, botones con estilo, áreas de texto cómodas y paneles bien organizados.

Tecnologías utilizadas:

-  **Java SE** (toda la lógica y la programación de hilos)
-  **Swing** (para la interfaz gráfica)
-  **Sockets TCP/IP** (comunicación en red entre clientes y servidor)
-  **Multithreading** (gestión eficiente de varios usuarios a la vez)
-  **FlatLaf** (estética moderna y personalizable)
-  **Eclipse** como entorno principal de desarrollo

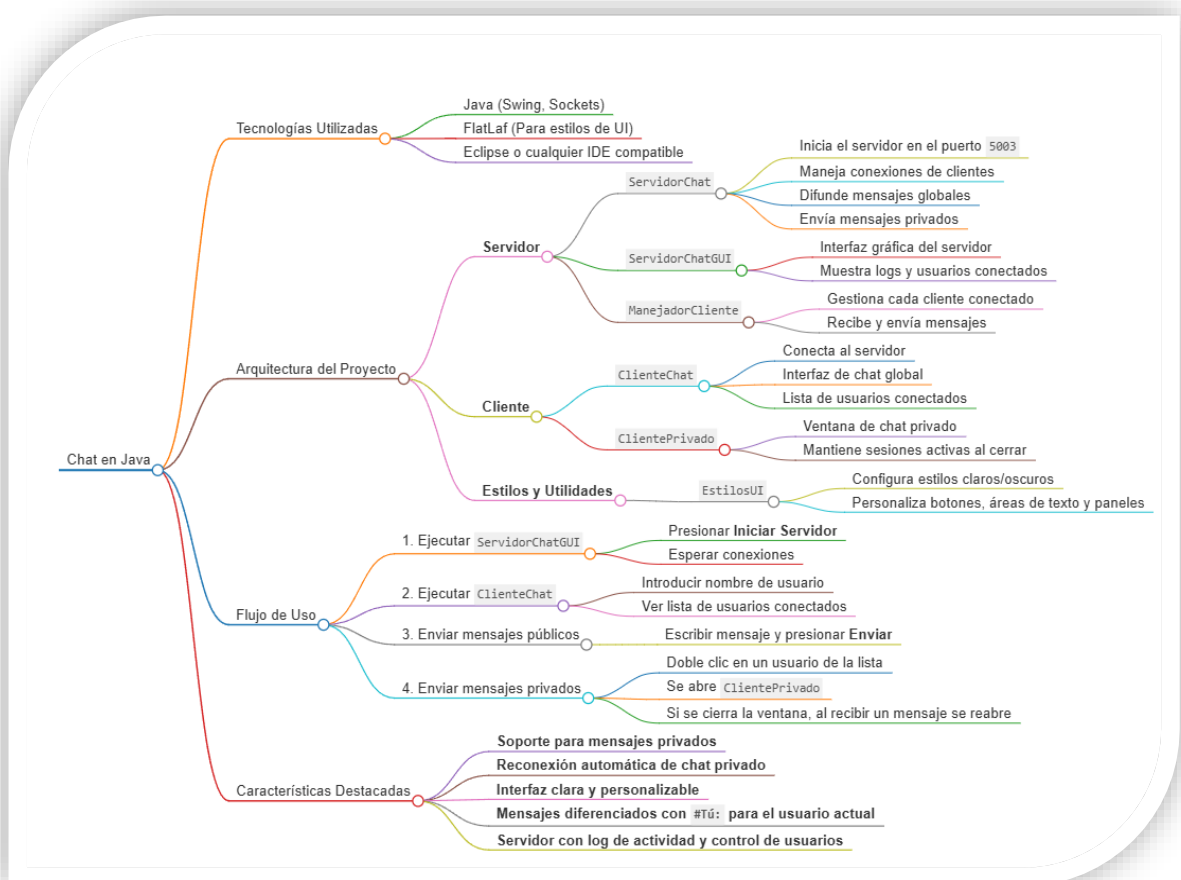
Características destacadas:

- Mensajes públicos y privados con comandos sencillos
- Lista de usuarios en tiempo real
- Chats privados independientes y automáticos al recibir nuevos mensajes
- Interfaz visual clara, cómoda y fácil de usar
- Log de actividad desde el propio servidor

En definitiva, se trata de un proyecto ideal para aprender y practicar **programación en red, gestión de hilos y diseño de interfaces gráficas en Java**. Tanto si buscas mejorar tus habilidades técnicas como si necesitas un ejemplo real de arquitectura cliente-servidor, aquí tienes un punto de partida sólido y flexible.

Mapa Mental – Arquitectura y Funcionamiento

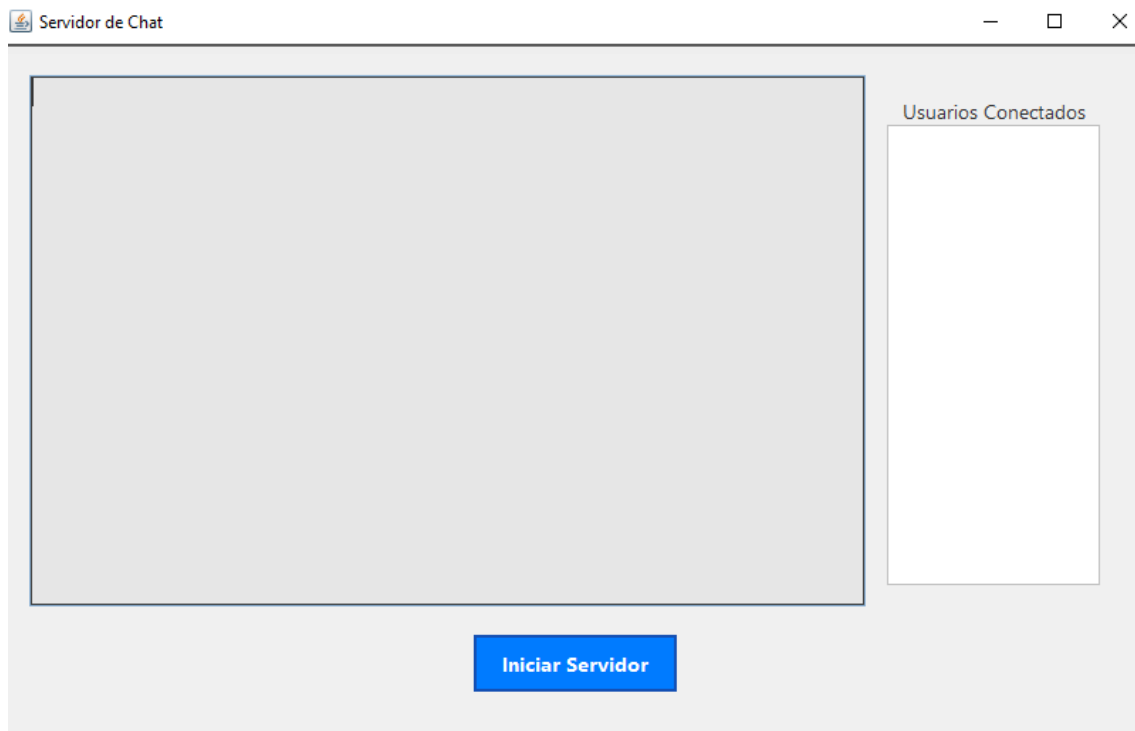
El siguiente mapa mental muestra la estructura general y el funcionamiento del sistema de chat, destacando los principales componentes, tecnologías utilizadas y el flujo de interacción entre servidor y clientes.



1. Ejecutar el Servidor

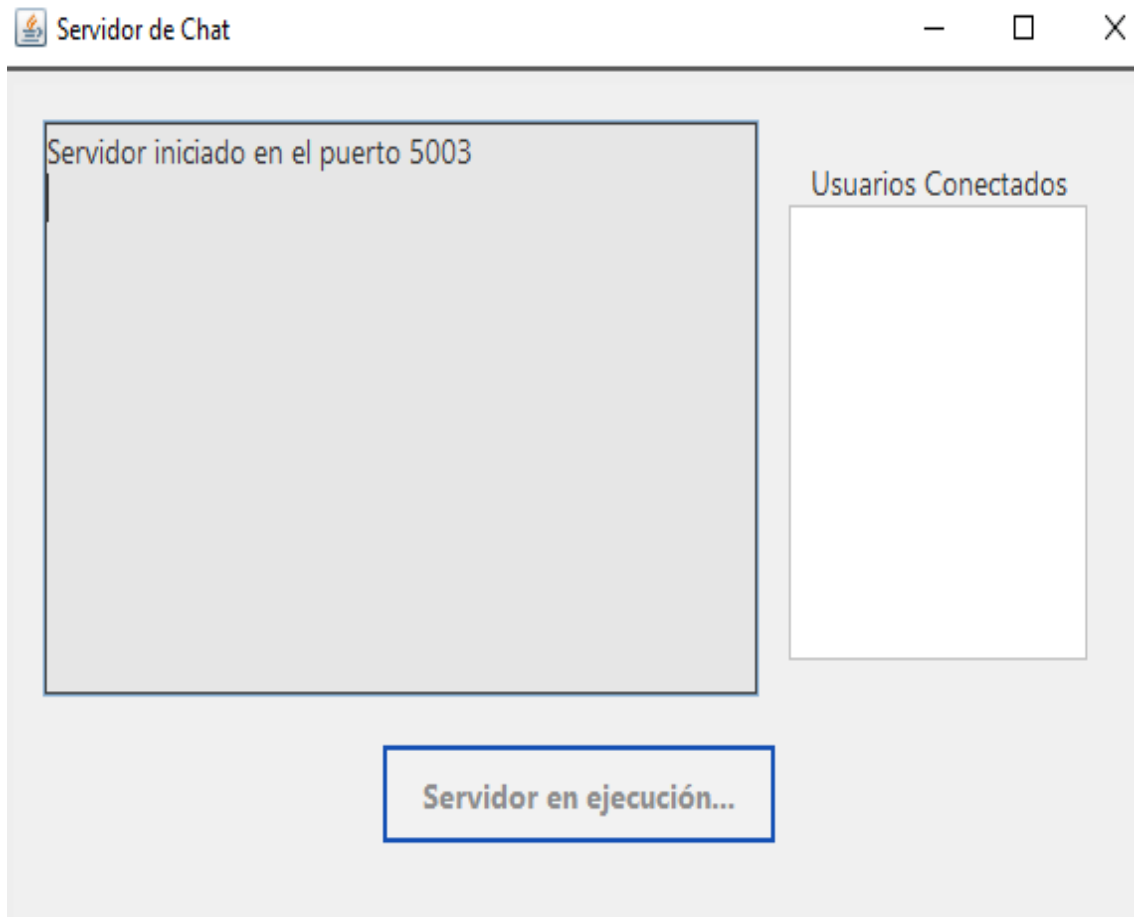
El primer paso para poner en marcha el chat es iniciar el **servidor**, que será el encargado de gestionar las conexiones de los clientes, mantener la lista de usuarios conectados y distribuir los mensajes. Todo esto se realiza desde una sencilla interfaz gráfica creada específicamente para facilitar el proceso.

Cuando abrimos la clase `ServidorChatGUI.java` y ejecutamos su método `main`, aparecerá la siguiente ventana principal del servidor:



En esta pantalla veremos una gran área de log (donde se mostrarán mensajes e incidencias) y un panel lateral para los usuarios conectados. De momento, sólo está disponible el botón azul "**Iniciar Servidor**".

Al pulsar este botón, el servidor se inicializa y comienza a escuchar conexiones en el puerto **5003**. La interfaz se actualiza mostrando que el servidor ya está en ejecución, listo para aceptar clientes:



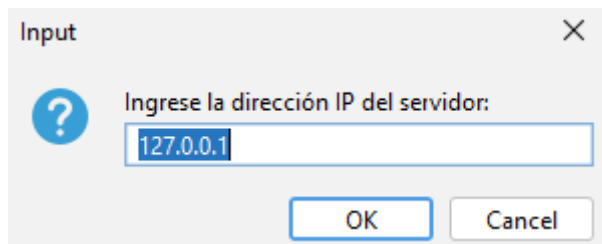
A partir de este momento, la aplicación está preparada para que los clientes se conecten y empiecen a chatear. El área de log se irá llenando con mensajes del sistema y las acciones realizadas, mientras que la lista de usuarios se irá poblando conforme se vayan conectando nuevos participantes.

2. Ejecutar el Cliente

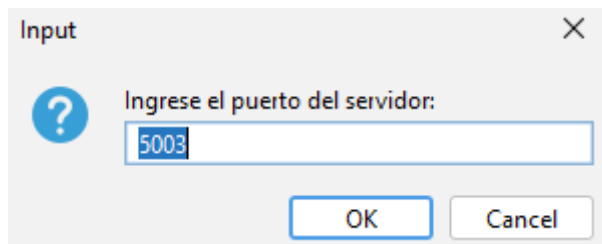
Para conectarte al chat, solo tienes que ejecutar el archivo principal del cliente. Nada más abrirlo, el programa te pedirá los datos necesarios para establecer la conexión con el servidor y configurar tu usuario.

Primero, al iniciar el cliente, verás varias ventanas emergentes:

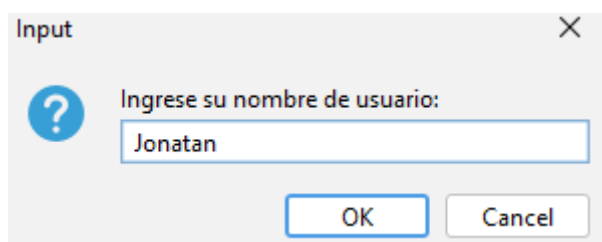
- **Dirección IP del servidor:** introduce la IP donde está corriendo el servidor. Si todo está en el mismo ordenador, puedes dejar el valor por defecto 127.0.0.1.

A screenshot of a Windows-style input dialog box titled "Input" with a close button (X) in the top right corner. On the left is a blue circular icon with a white question mark. The text "Ingrese la dirección IP del servidor:" is displayed. Below it is a text input field containing "127.0.0.1". At the bottom are two buttons: "OK" and "Cancel".

- **Puerto:** a continuación, te pedirá el puerto de conexión.

A screenshot of a Windows-style input dialog box titled "Input" with a close button (X) in the top right corner. On the left is a blue circular icon with a white question mark. The text "Ingrese el puerto del servidor:" is displayed. Below it is a text input field containing "5003". At the bottom are two buttons: "OK" and "Cancel".

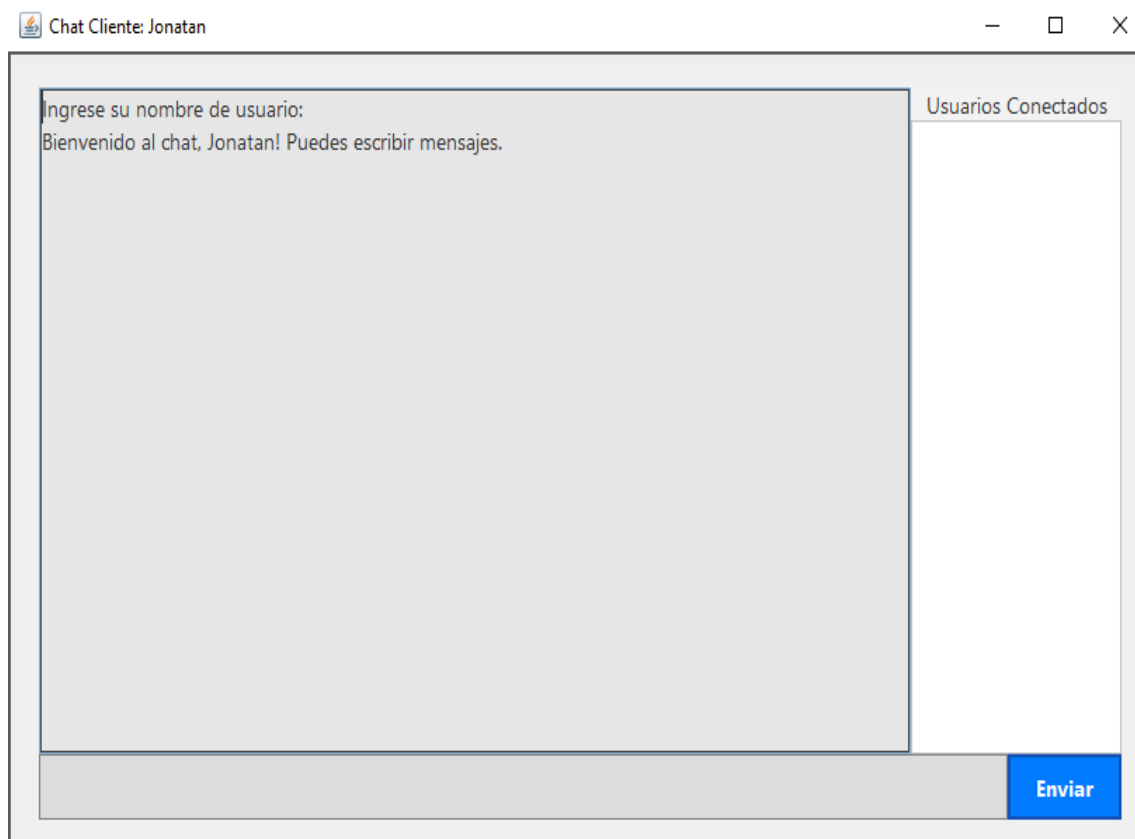
- **Nombre de usuario:** finalmente, introduce el nombre con el que aparecerás en el chat.

A screenshot of a Windows-style input dialog box titled "Input" with a close button (X) in the top right corner. On the left is a blue circular icon with a white question mark. The text "Ingrese su nombre de usuario:" is displayed. Below it is a text input field containing "Jonatan". At the bottom are two buttons: "OK" and "Cancel".

Una vez introducidos los datos de conexión, se abrirá la **ventana principal del cliente de chat**.

En la parte central, encontrarás el **área de chat general**, donde aparecerán todos los mensajes enviados por los usuarios conectados al servidor. A la derecha, tienes la lista de **usuarios conectados** en ese momento (en la imagen, por ejemplo, aparece solo un usuario conectado).

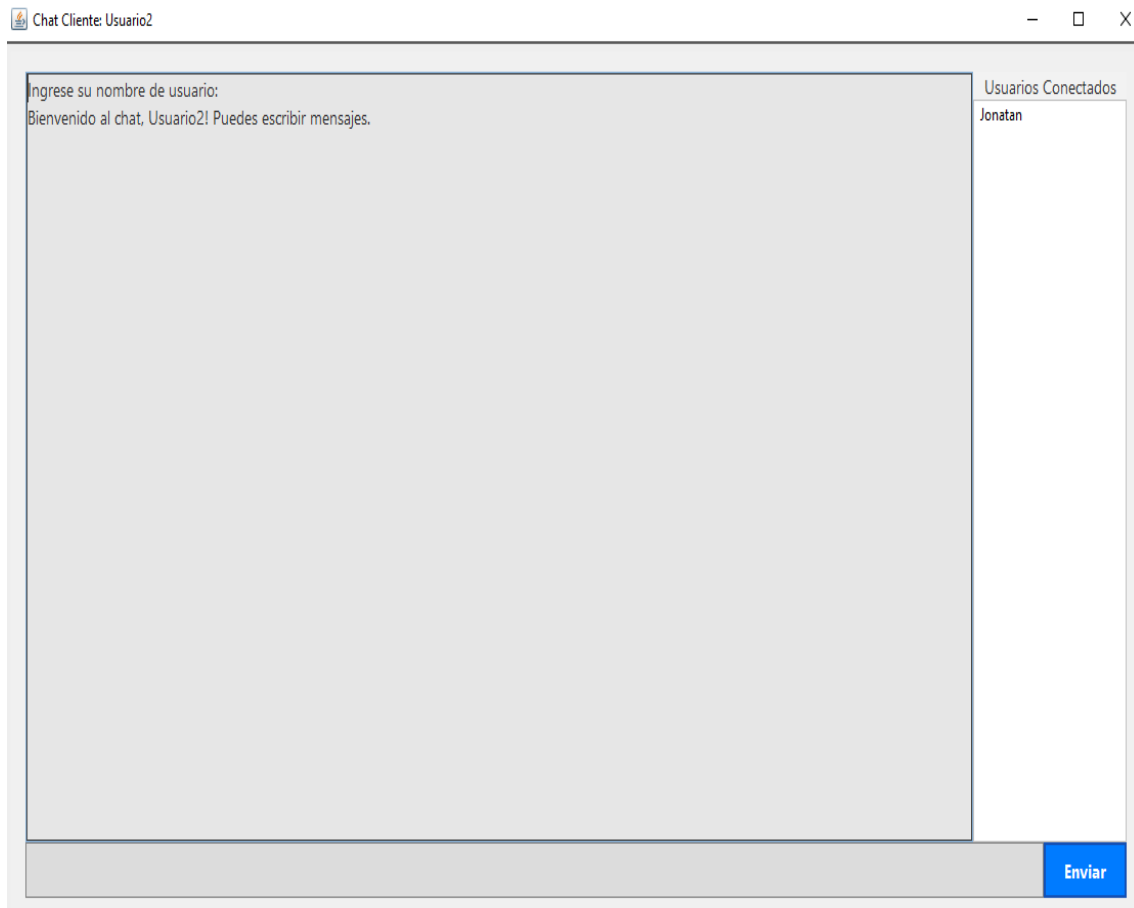
En la parte inferior de la ventana está el **campo de texto** donde puedes escribir tus mensajes. Cuando pulses el botón **Enviar**, el mensaje se enviará a todos los usuarios conectados, como en cualquier chat tradicional.



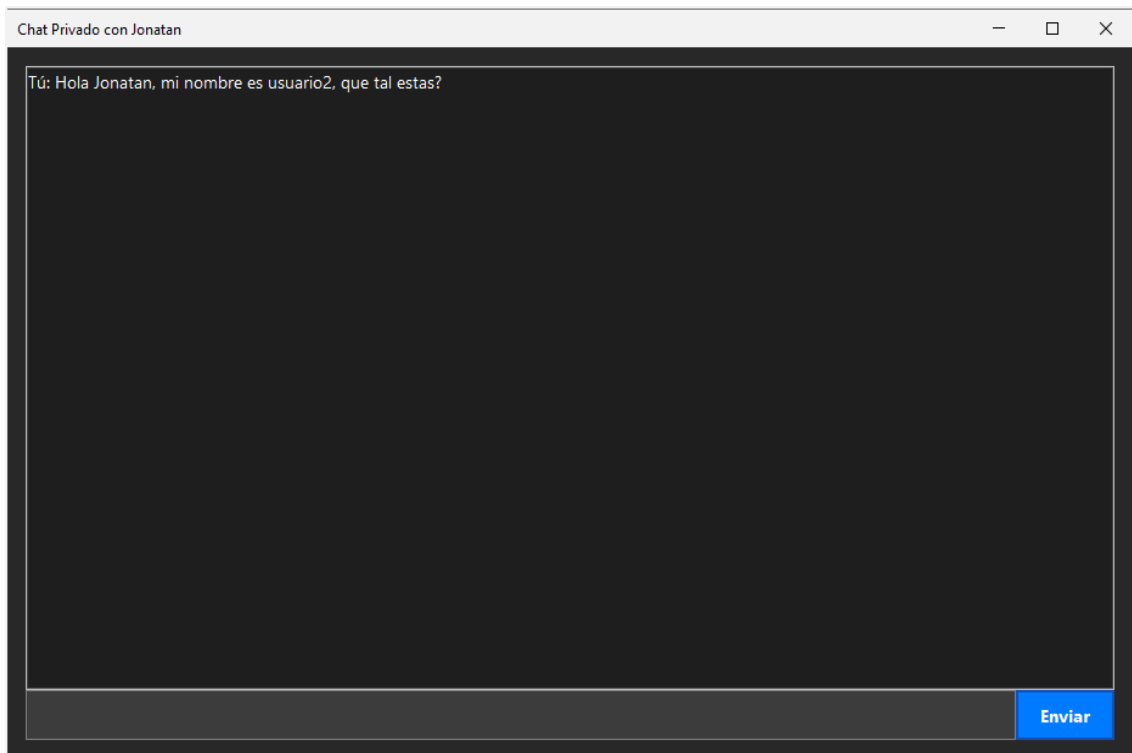
Esta interfaz te permite mantener conversaciones públicas en el chat general y, como veremos más adelante, también podrás iniciar conversaciones privadas con cualquier usuario conectado haciendo doble clic en su nombre.

Ahora, cuando un segundo usuario se conecta, el chat pasa a ser realmente **multichat**: cada cliente puede ver la lista de usuarios conectados y comunicarse con ellos de diferentes formas.

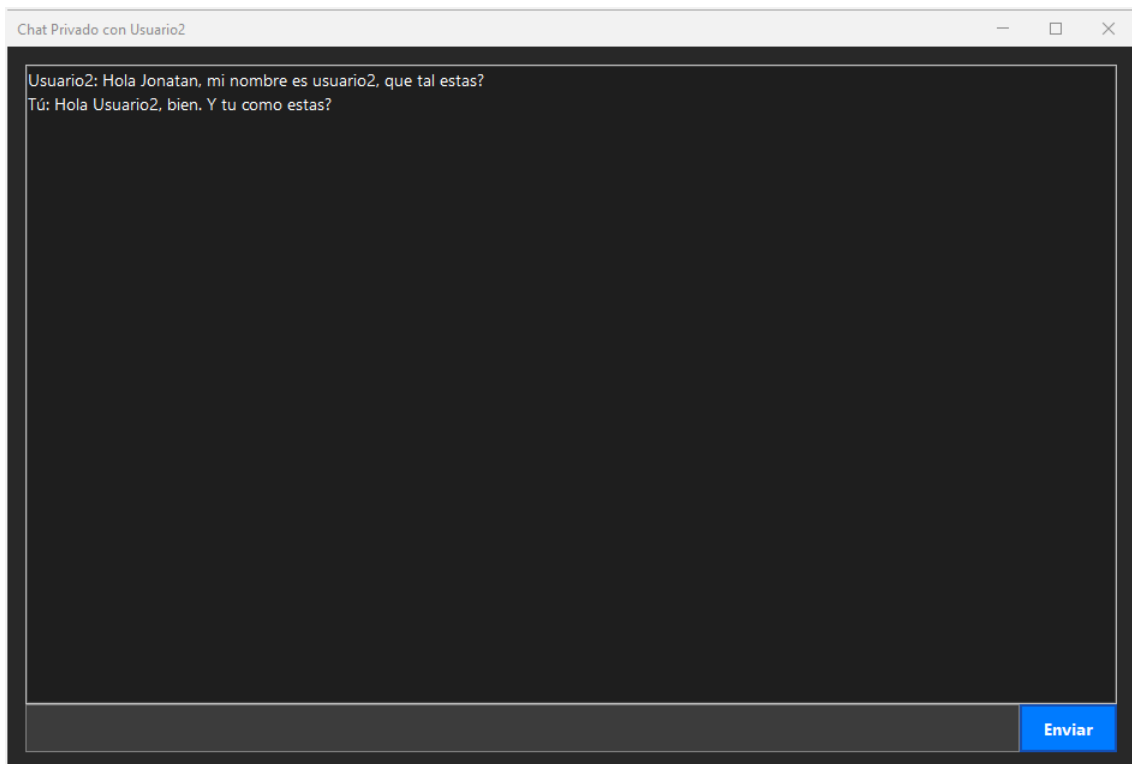
Para iniciar una conversación privada, **basta con hacer doble clic** sobre el nombre de otro usuario (por ejemplo, "Jonatan") en la lista de la derecha. Al hacerlo, se abrirá una **nueva ventana** con un estilo visual diferente (generalmente en modo oscuro), lo que te indica que estás en un chat privado.



En la parte superior de la ventana de chat privado, verás claramente el nombre del usuario con el que estás conversando (en este caso, Jonatan), y cada mensaje aparecerá en el área de conversación. Los mensajes que tú envíes estarán marcados como “Tú”, para que puedas distinguirlos fácilmente.

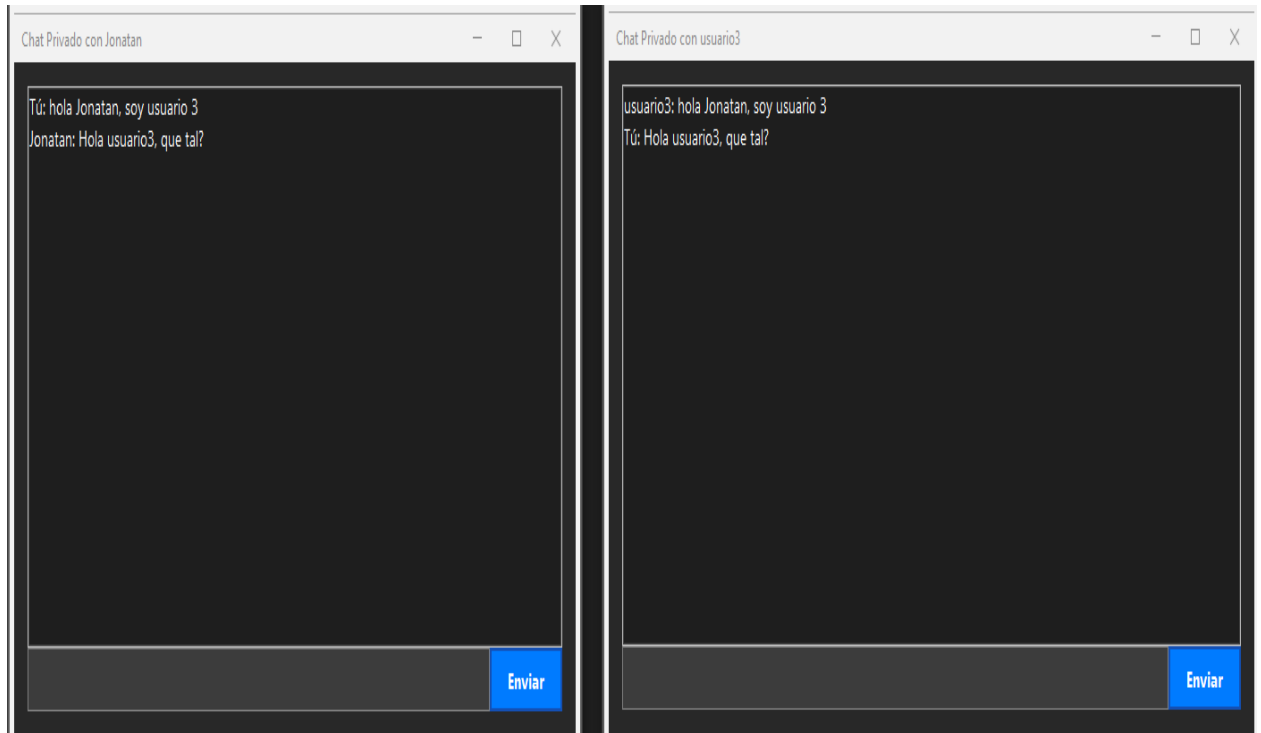


Ventana Jonatan



Esto permite mantener varias conversaciones privadas abiertas al mismo tiempo, sin perder de vista el chat general. Además, si cierras una ventana privada y te vuelven a escribir, la conversación se reabre automáticamente.

Ventanas de chats privados del segundo usuario, con Jonatan y usuario3:

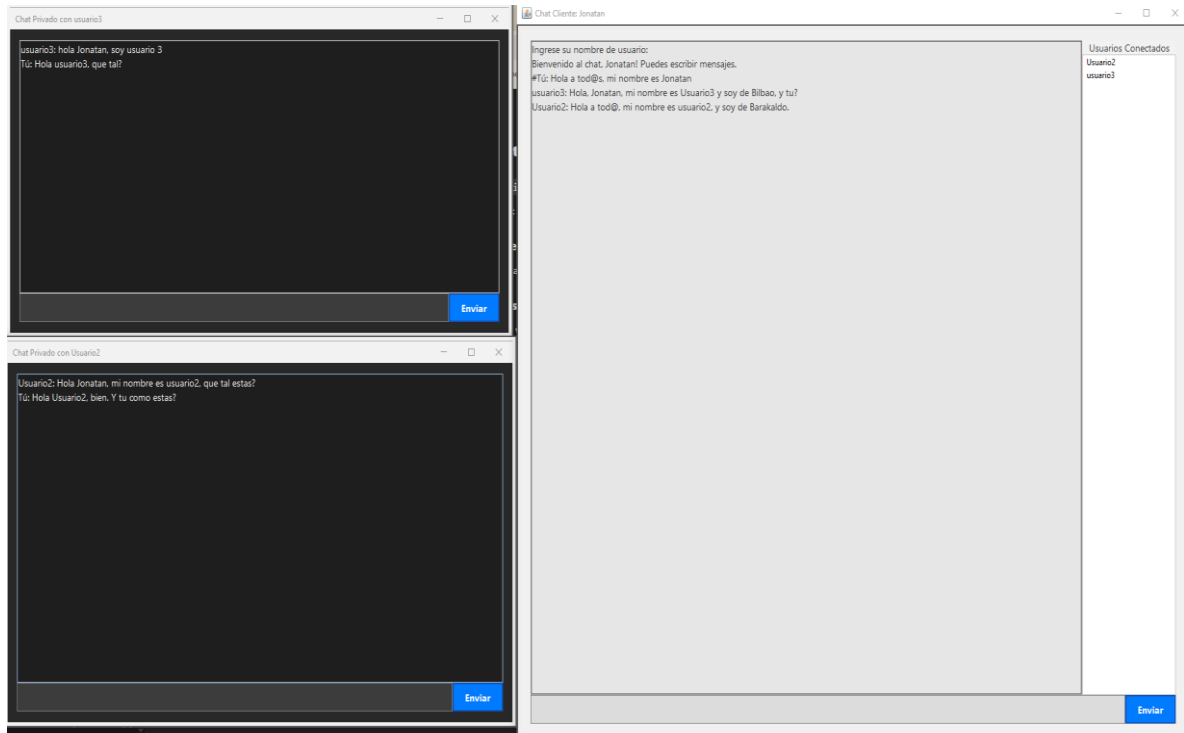


De esta forma, la aplicación de chat no solo permite mensajes globales a todos los usuarios, sino que también ofrece una experiencia más completa gracias a los **chats privados** al estilo de cualquier chat profesional.

Cuando hay varios usuarios conectados, cada uno puede tener abiertas varias ventanas de chat a la vez: una ventana principal para el chat general y ventanas adicionales para conversaciones privadas con otros usuarios. Así, la aplicación permite mantener varias conversaciones al mismo tiempo, igual que en los chats modernos.

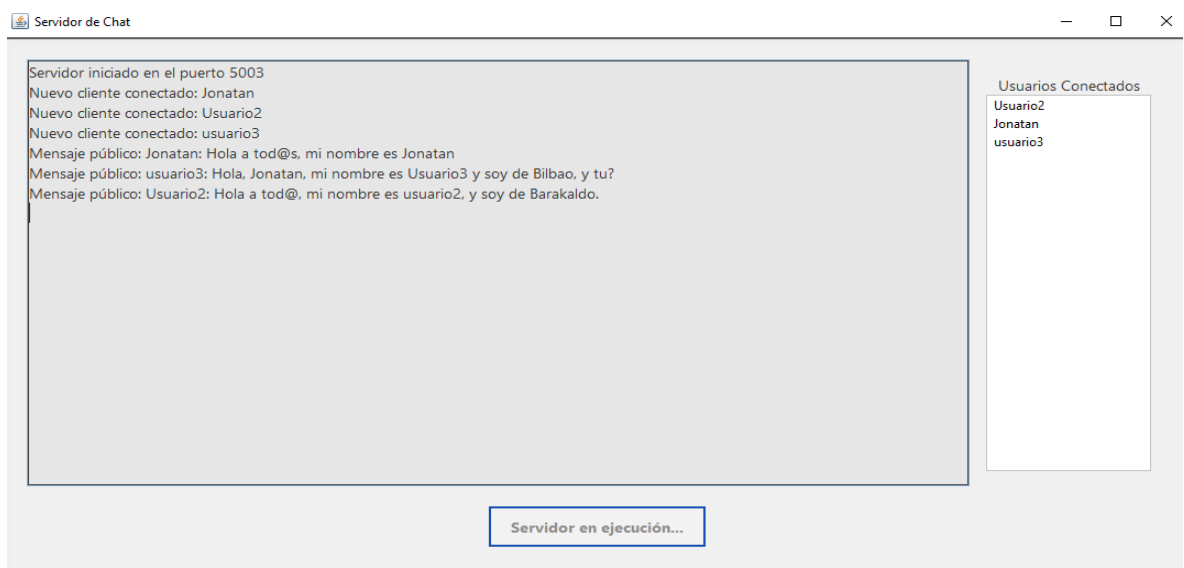
En la imagen se puede ver:

- **Ventanas de chat privado** (abajo a la izquierda): Cada una corresponde a una conversación individual con otro usuario. Si tienes varias conversaciones privadas activas, se abrirán tantas ventanas como chats tengas.
- **Ventana principal del cliente** (arriba a la derecha): Aquí puedes ver el chat general, donde aparecen los mensajes públicos que envía todo el grupo, así como la lista de usuarios conectados.



- **Ventana del servidor** (abajo): El servidor registra toda la actividad, mostrando quién se conecta, quién se desconecta y todos los mensajes públicos que se envían. También muestra la lista completa de usuarios conectados en ese momento.

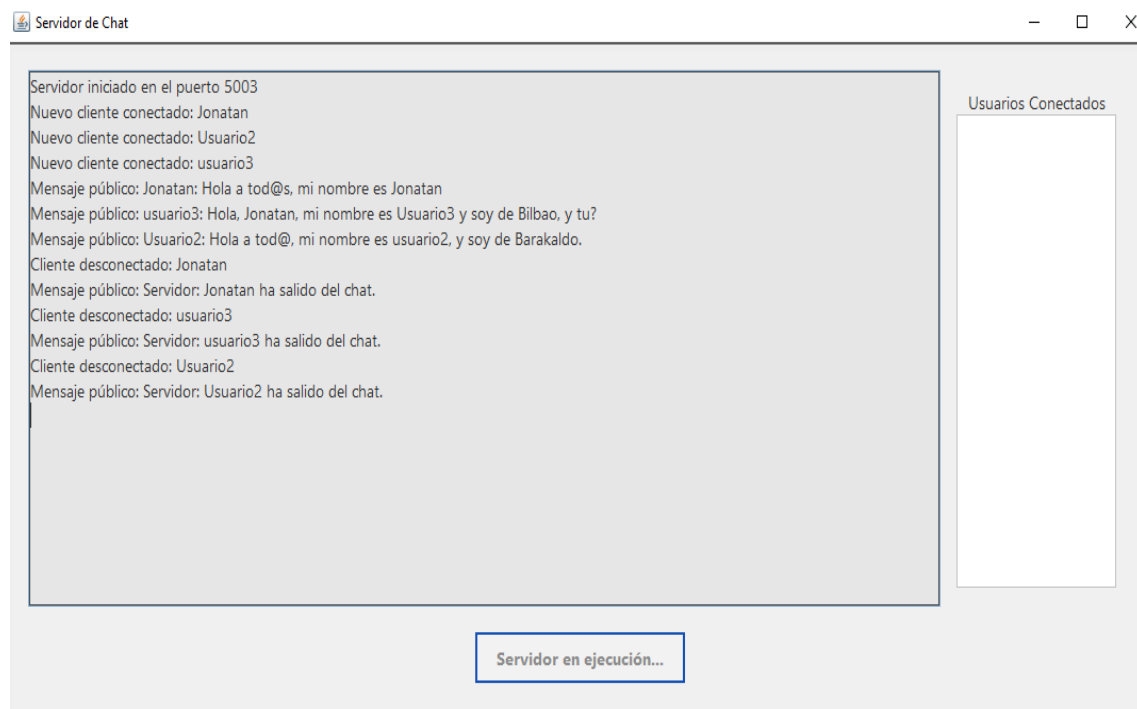
Esto te permite ver de un solo vistazo cómo se distribuye la comunicación: cada usuario puede enviar mensajes públicos al grupo o mantener conversaciones privadas, mientras que el servidor lo controla todo y lleva el registro de la actividad.



El chat permite que se conecten tantos usuarios como quieras (¡N usuarios, sin límite práctico para pruebas normales!). Por ahora, solo puedes enviar mensajes de texto, así que no es posible adjuntar archivos ni fotos de perfil, pero cumple perfectamente para conversaciones fluidas de texto entre distintos usuarios.

Si en algún momento el servidor se apaga (por ejemplo, si cierras la ventana del servidor), **todos los usuarios conectados perderán la conexión**, ya que el servidor es el centro de toda la comunicación. Simplemente vuelve a iniciar el servidor y los clientes podrán reconectarse normalmente.

Cuando un usuario abandona la sala, **el servidor lo notifica automáticamente a todos los usuarios conectados**, mostrando en la ventana del servidor y de los clientes quién ha salido del chat. Así, siempre sabes quién está conectado y quién no, como se ve en la siguiente imagen:



Conclusión y Orientación del Proyecto

Este proyecto de chat cliente-servidor en Java ha servido para poner en práctica los principales conceptos de la programación en red: sockets, multihilo y el diseño de interfaces gráficas con Swing.

Gracias a la implementación tanto del servidor como de los clientes, se ha logrado comprender el flujo de mensajes, la gestión de usuarios y la creación de canales privados de comunicación entre ellos.

Además, el uso de FlatLaf ha permitido modernizar la apariencia de la interfaz, logrando una experiencia más profesional y agradable. El proyecto está orientado al aprendizaje, por lo que prioriza la claridad en el código y la estructura modular, facilitando su comprensión y posible ampliación en el futuro.

En definitiva, este chat es una base sólida para practicar y entender los fundamentos de la comunicación TCP/IP en Java, ideal como ejercicio de clase o punto de partida para proyectos más avanzados.