Reporte de Creación del Código: “TCPServidorHolaMundo.java”

Jeni Pamela Espinoza Rivera

1. Objetivo

El objetivo principal de este código es crear un servidor TCP básico que escuche en un puerto específico, acepte conexiones de clientes, reciba un mensaje, y lo muestre en la consola. Este código se compiló y ejecutó utilizando el entorno de desarrollo JGRASP.

2. Pasos de Desarrollo

a. Configuración del Entorno

- Entorno de Desarrollo: JGRASP.

- Librerías Importadas:

  - `java.io.`: Para manejar operaciones de entrada/salida.

  - `java.net.`: Para las clases necesarias para la programación de redes.

b. Desarrollo del Código

1. Definición de la Clase:

  - Se creó una clase pública llamada `TCPServidorHolaMundo`.

2. Configuración del Puerto:

  - Se definió el puerto en el cual el servidor escuchará las conexiones entrantes. En este caso, se utilizó el puerto `12345`.

3. Creación del Socket del Servidor:

  - Se instanció un objeto `ServerSocket` en el puerto especificado para permitir que el servidor escuche las conexiones entrantes.

4. Esperar la Conexión de un Cliente:

  - El servidor quedó en espera de que un cliente se conecte utilizando el método `accept()`. Cuando un cliente se conecta, se devuelve un objeto `Socket` que representa la conexión.

5. Recepción del Mensaje:

  - Una vez establecida la conexión, se creó un `BufferedReader` para leer los datos enviados por el cliente a través del flujo de entrada (`InputStream`).

  - El mensaje recibido se leyó utilizando `readLine()` y se imprimió en la consola.

6. Cierre de la Conexión:

  - Después de recibir y mostrar el mensaje, se cerraron tanto el socket del cliente como el socket del servidor para liberar los recursos.

7. Manejo de Excepciones:

  - Se implementó un bloque `try-catch` para manejar posibles errores de E/S (por ejemplo, si el puerto ya está en uso o si ocurre un problema al aceptar la conexión).

c. Compilación y Ejecución

- Compilación:

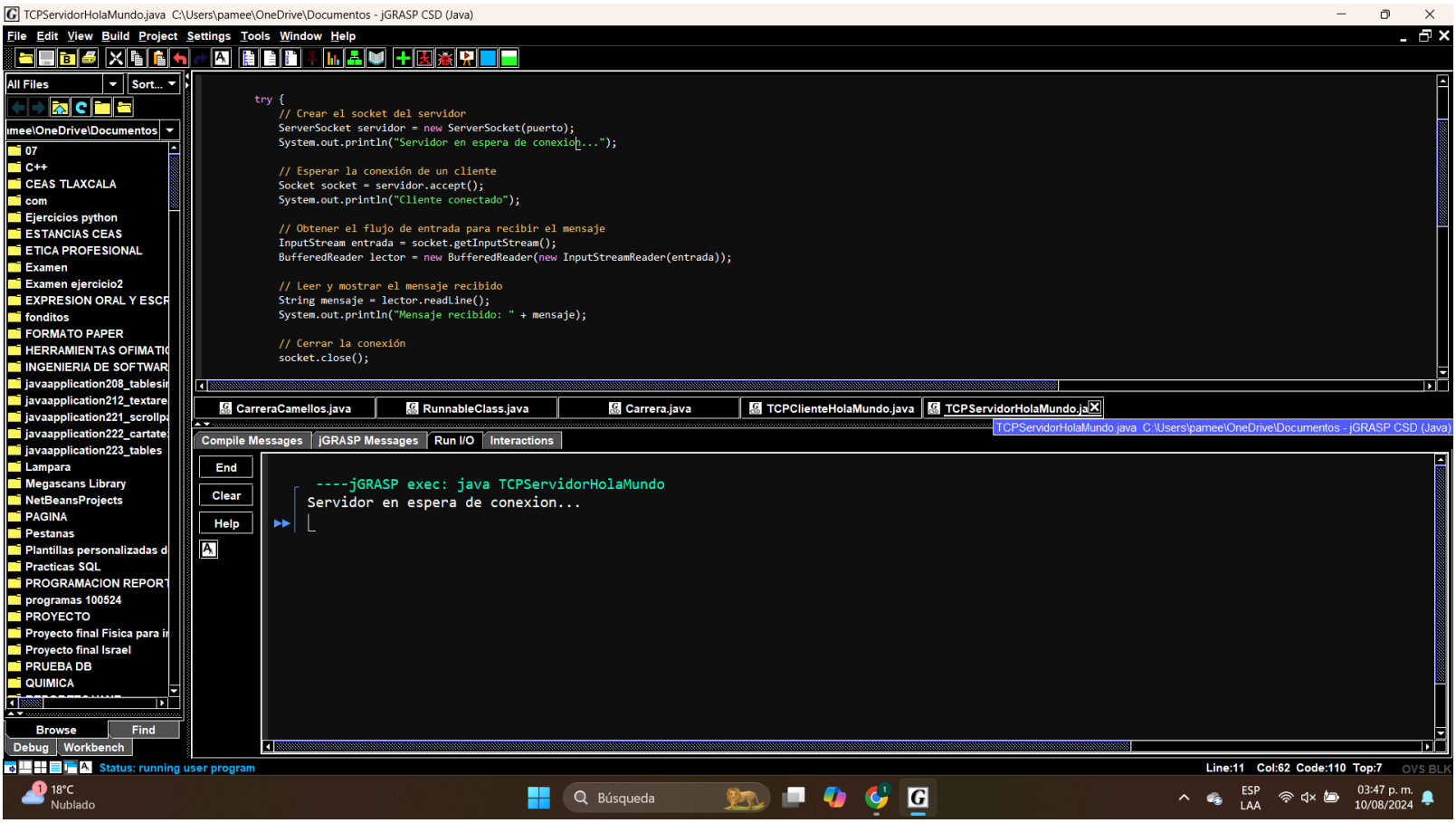
  - El código fue compilado sin errores en JGRASP.

- Ejecución:

  - Al ejecutar el programa, el servidor quedó a la espera de una conexión de un cliente. Una vez que el cliente se conectó y envió un mensaje, el servidor lo mostró correctamente en la consola.

3. Resultado Final

El código `TCPServidorHolaMundo.java` fue creado y ejecutado exitosamente, cumpliendo con el objetivo de establecer una conexión TCP, recibir un mensaje de un cliente, y mostrarlo en la consola. La implementación es básica pero funcional, adecuada para aplicaciones simples de red donde se requiere comunicación directa entre un cliente y un servidor.



1 Reporte de Creación del Código: "TCPClienteHolaMundo.java"

Jeni Pamela Espinoza Rivera.

1. Objetivo

El objetivo principal de este código es crear un cliente TCP básico que se conecte a un servidor en un puerto específico, envíe un mensaje, y cierre la conexión. Este código fue compilado y ejecutado en JGRASP.

2. Pasos de Desarrollo

a. Configuración del Entorno

- Entorno de Desarrollo**:** JGRASP.
- Librerías Importadas**:**
  - java.io.*: Para manejar las operaciones de entrada/salida.
  - java.net.*: Para las clases necesarias para la programación de redes.

b. Desarrollo del Código

1. **Definición de la Clase**:
   - Se creó una clase pública llamada TCPClienteHolaMundo.
2. **Configuración del Servidor y Puerto**:
   - Se especificó la dirección del servidor (en este caso, "localhost", que se refiere a la misma máquina donde se ejecuta el cliente) y el puerto 12345 para conectarse al servidor.
3. **Conexión al Servidor**:
   - Se instanció un objeto Socket utilizando la dirección del servidor y el puerto especificado. Esto estableció una conexión TCP con el servidor.
4. **Envío del Mensaje**:
   - Se obtuvo el flujo de salida (OutputStream) del socket para poder enviar datos al servidor.
   - Un PrintWriter fue utilizado para enviar el mensaje "Hola Mundo" al servidor. El método println envió el mensaje seguido de un salto de línea, lo que facilitó la lectura en el servidor.
5. **Cierre de la Conexión**:
   - Después de enviar el mensaje, se cerró el socket para finalizar la conexión y liberar los recursos asociados.
6. **Manejo de Excepciones**:
   - Se implementó un bloque try-catch para manejar posibles errores de E/S, como la imposibilidad de conectar con el servidor (por ejemplo, si el servidor no está disponible o si ocurre un problema de red).

c. Compilación y Ejecución

- **Compilación**:
  - El código fue compilado sin errores en JGRASP.
- **Ejecución**:
  - Al ejecutar el programa, el cliente se conectó al servidor en localhost y envió el mensaje "Hola Mundo". El proceso se completó correctamente sin errores.

3. Resultado final

El código TCPClienteHolaMundo.java fue creado y ejecutado exitosamente, logrando conectarse al servidor TCP previamente desarrollado (TCPServidorHolaMundo.java), enviar un mensaje y cerrar la conexión de manera ordenada. El código es simple y efectivo para aplicaciones donde se necesita

comunicación directa entre un cliente y un servidor.