Memoria Práctica Servidor FTP / Servicios en Red

Andreu Sanz Sanz
2DAM

Qué

Servidor FTP

Para qué

Este ejercicio ha resultado útil para aprender y practicar el funcionamiento de los Sockets y comprender cómo funciona el protocolo TCP. La actividad sirve para repasar la programación con hilos y concurrencia, además de tener una idea básica de la programación de un servidor TCP con Java o C, dependiendo del lenguaje que queramos usar. Yo he decidido utilizar Java, ya que estoy más familiarizado con él.

Cómo

Esta práctica consistía en la implementación de un servidor FTP. Un servidor FTP, o Protocolo de Transferencia de Archivos, es un software esencial para facilitar la transferencia eficiente de archivos entre dispositivos en una red. Define un estándar de comunicación mediante conexiones de control y datos, permitiendo la gestión de archivos, directorios y la autenticación de usuarios. Para esta práctica he creado dos clases, `Server` que se encarga de aceptar las conexiones y `Client` que se encarga de gestionar los comandos introducidos por el cliente estándar.

Clase Server

Esta clase solo contiene el método principal que comienza definiendo el puerto que se va a utilizar asignándolo a una variable y crea un `ServerSocket` en el puerto especificado. Este es responsable de aceptar conexiones entrantes de clientes. El servidor entra en un bucle para esperar y aceptar conexiones continuamente. Cuando un cliente se conecta, el método `accept()` del `ServerSocket` devuelve un objeto Socket que representa la conexión con ese cliente. Después se crea un nuevo hilo (`Thread`) para manejar la conexión con el cliente. El hilo utiliza la clase `Client` que implementa la lógica específica para manejar las operaciones del cliente. El hilo recién creado se inicia con el método `start()`. Esto permite que el servidor continúe aceptando conexiones mientras maneja las operaciones de clientes en hilos separados.

Clase Client

Esta clase implementa la interfaz `Runnable`. En el método `run` de esta clase, establece los flujos de entrada y salida (reader y writer) para la comunicación con el cliente. Además, envía un mensaje inicial al cliente indicando que el servidor FTP está listo. Dentro de un bucle, lee los comandos enviados por el cliente, los imprime en la consola y llama a `handleCommand()` para procesarlos. Si el comando es "QUIT", el bucle se rompe y se cierra el socket del cliente.

El método `handleCommand(String command)` procesa los diferentes comandos FTP enviados por el cliente y llama a los métodos correspondientes. Esta es la lista de los comandos implementados:

- `handleUserCommand(String username)`: Maneja el comando "USER" y valida el nombre de usuario.
- `handlePassCommand(String password)`: Maneja el comando "PASS" y valida la contraseña, autenticando al usuario si es correcta.
- `handleCwdCommand(String args)`: Maneja el comando "CWD" (Change Working Directory) para cambiar el directorio actual.
- `handlePwdCommand()`: Maneja el comando "PWD" (Print Working Directory) para imprimir el directorio actual.
- `handleListCommand()`: Maneja el comando "LIST" para listar los archivos en el directorio actual.
- `handleHelpCommand()`: Maneja el comando "HELP" para proporcionar una lista de comandos reconocidos.
- `handleNoopCommand()`: Maneja el comando "NOOP" (No Operation) para mantener la conexión activa.
- `handleReinCommand()`: Maneja el comando "REIN" (Reinitialize) para reiniciar la sesión del usuario.
- `handleMkdCommand(String directory)`: Maneja el comando "MKD" (Make Directory) para crear un nuevo directorio.

Conclusión

En conclusión, considero que esta actividad ha sido útil y ha contribuido significativamente a mejorar mis habilidades de programación con `Sockets` y `Threads`. Además, me intriga conocer las diferentes aproximaciones que mis compañeros han tomado para abordar esta actividad, ya que soy consciente de que hay varias formas de implementarla. Consultar a mis amigos y obtener explicaciones sobre sus enfoques podría proporcionarme valiosas perspectivas y aprender nuevas técnicas. También ver cómo se hace esta actividad en `C` para poder comprender un poco más cómo funciona `C`.

En general, me ha parecido interesante descubrir cómo desarrollar una idea muy básica de los tipos de servidores `TCP`. Considero que la dificultad de esta práctica no ha sido elevada, exceptuando algunos comandos que no he sabido cómo implementarlos.