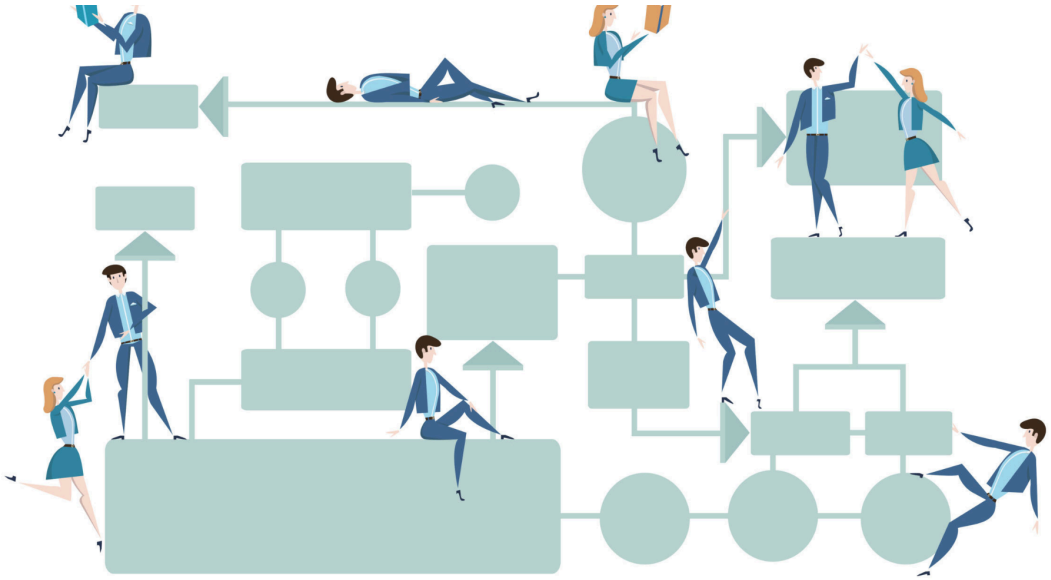
HITO 2 DEL 1º TRIMESTRE DE Programación de Servicios y Procesos

David Ronaldo García

F.Entrega: 20/11/2024



Capa de acceso a datos

AccesoDatos.java:

```
package psp_h2_servidor_cliente;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class AccesoDatos {
  private String rutaArchivo; // Ruta del archivo donde buscaremos la información.
  // Constructor que inicializa la ruta del archivo
  public AccesoDatos(String rutaArchivo) {
     this.rutaArchivo = rutaArchivo;
  }
  // Método que busca la información en el archivo usando una clave de búsqueda
  public String buscarInformacion(String clave) {
     // Tratamos de leer el archivo usando FileReader
     try (BufferedReader br = new BufferedReader(new FileReader(this.rutaArchivo))) {
        String linea;
        // Lee el archivo línea por línea
        while ((linea = br.readLine()) != null) {
           // Comprobamos si la línea contiene la clave, sin importar mayúsculas o minúsculas
           if (linea.toLowerCase().contains(clave.toLowerCase())) {
              return linea; // Si encuentra la clave, devuelve la línea
           }
        }
     } catch (IOException e) {
        e.printStackTrace(); // Si ocurre un error, se muestra en la consola
     }
```

Explicación: Esta clase se encarga de leer un archivo (datos.txt) y buscar información dentro de él. Utiliza un BufferedReader para leer el archivo línea por línea y busca si la línea contiene una palabra clave proporcionada como parámetro. Si la encuentra, devuelve la línea, si no, devuelve un mensaje indicando que no se encontró información. Es la parte del sistema que interactúa directamente con el archivo donde se almacena la información.

Capa de Servidor

Principal.java:

```
package psp_h2_servidor_cliente;
public class Principal {
  public static void main(String[] args) {
    int puerto = 12346; // Puerto donde el servidor escuchará
    String rutaArchivo = "datos.txt"; // Ruta del archivo de datos que el servidor usará
    // Crea una instancia del servidor con el puerto y la ruta del archivo
    Servidor servidor = new Servidor(puerto, rutaArchivo);
    servidor.iniciar(); // Inicia el servidor
  }
}
```

Explicación: En esta clase se crea una instancia del servidor con el puerto y la ruta del archivo que se va a utilizar para las búsquedas. El servidor es iniciado mediante el método iniciar() del objeto Servidor. Esta clase está centrada en iniciar el servidor que atenderá las solicitudes de los clientes. Busca en este caso el archivo datos.txt y el servidor por el puerto que hemos creado (números aleatorios los cuales no estén ya en funcionamiento para que no de problemas).

Servidor.java

```
package psp_h2_servidor_cliente;
import java.io.*;
import java.net.*;
public class Servidor {
  private AccesoDatos accesoDatos; // Objeto que maneja el acceso al archivo
  private int puerto; // Puerto donde el servidor escuchará
  // Constructor que inicializa el puerto y el objeto AccesoDatos
  public Servidor(int puerto, String rutaArchivo) {
    this.puerto = puerto;
    this.accesoDatos = new AccesoDatos(rutaArchivo); // Inicializamos con la ruta del archivo
  }
  // Método para iniciar el servidor y aceptar conexiones
  public void iniciar() {
    try (ServerSocket servidor = new ServerSocket(puerto)) { // Crear el servidor que escucha en el
puerto
      System.out.println("Servidor iniciado en el puerto " + puerto + ". Esperando conexiones...");
      while (true) {
        // Acepta una conexión del cliente
        Socket cliente = servidor.accept();
        System.out.println("Cliente conectado.");
        // Configura el flujo de entrada y salida con el cliente
        BufferedReader entrada = new BufferedReader(new
InputStreamReader(cliente.getInputStream()));
        PrintWriter salida = new PrintWriter(cliente.getOutputStream(), true);
        // Lee la clave de búsqueda que envió el cliente
        String claveBusqueda = entrada.readLine();
```

```
            System.out.println("Buscando información para la clave: " + claveBusqueda);
            // Busca la información usando la clase AccesoDatos
            String resultado = this.accesoDatos.buscarInformacion(claveBusqueda);
            salida.println(resultado); // Enviar el resultado al cliente
            cliente.close(); // Cierra la conexión con el cliente
            System.out.println("Cliente desconectado.");
        }
    } catch (IOException e) {
        e.printStackTrace(); // Si ocurre algún error, se imprime en consola
    }
  }
}
```

Explicación: Esta clase es la responsable de crear un servidor que escucha solicitudes de los clientes. Una vez que el servidor recibe una solicitud (una palabra clave), utiliza la clase AccesoDatos para buscar la información en un archivo de datos. Luego, envía la respuesta al cliente. Es el corazón del sistema que maneja la comunicación y lógica de las solicitudes de los clientes, accediendo a los datos cuando es necesario.

Capa de Cliente

Cliente.java:

```
package psp_h2_servidor_cliente;
import java.io.*;
import java.net.*;
public class Cliente {
  private String direccionServidor; // Dirección del servidor (en este caso, localhost)
  private int puerto; // Puerto del servidor
  // Constructor que inicializa la dirección y el puerto del servidor
  public Cliente(String direccionServidor, int puerto) {
    this.direccionServidor = direccionServidor;
    this.puerto = puerto;
  }
  // Método para solicitar información al servidor
  public void solicitarInfo(String palabraClave) {
    try (Socket socket = new Socket(direccionServidor, puerto);
         BufferedReader entrada = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
         PrintWriter salida = new PrintWriter(socket.getOutputStream(), true)) {
        // Envia la palabra clave al servidor
        salida.println(palabraClave);
        // Recibe y muestra la respuesta del servidor
        String respuesta;
        while ((respuesta = entrada.readLine()) != null) {
          System.out.println("Respuesta del servidor: " + respuesta);
        }
    } catch (IOException e) {
        System.out.println("Error al conectar con el servidor: " + e.getMessage());
    }
  }
}
```

Explicación: Esta clase simula un cliente que se conecta a un servidor a través de un Socket para solicitar información. El cliente envía una palabra clave (Java) al servidor, que es procesada y buscada en el archivo del servidor. Luego, el cliente recibe la respuesta del servidor y la muestra en la consola. Es responsable de hacer la solicitud de búsqueda y mostrar los resultados al usuario.

PruebaCliente.java:

```
package psp_h2_servidor_cliente;
public class PruebaCliente {
  public static void main(String[] args) {
      String direccionServidor = "localhost"; // Dirección del servidor (localhost significa que está en el mismo equipo)
      int puerto = 12345; // Puerto donde el servidor está escuchando
      // Crea el cliente y enviar una solicitud con la palabra clave
      Cliente cliente = new Cliente(direccionServidor, puerto);
      cliente.solicitarInfo("Java"); // Palabra clave para buscar en el archivo
  }
}
```

Explicación: Esta clase se encarga de probar el cliente. Crea una instancia de Cliente y solicita al servidor información sobre "Java" a través de una conexión de red. Esta clase tiene como objetivo simular el comportamiento del cliente que solicita información al servidor, enviando una clave de búsqueda y recibiendo la respuesta del servidor.

Resultado :

```
Servidor iniciado en el puerto 12346. Esperando conexiones...
```

Primero ejecutamos la clase principal que como hemos dicho antes crea una instancia del servidor con el puerto y la ruta del archivo que se va a utilizar para las búsquedas

```
Java - Libro sobre programación en Java. Una guía completa para aprender Java desde cero.
```

Luego se ejecutará el resto el cual como he explicado anteriormente tienen sus funciones de filtrar la información con palabras clave en este caso con java.

Vamos a realizar una prueba con otra palabra:

```
// Prueba la búsqueda con una clave
      String resultado = accesoDatos.buscarInformacion("HTML"); // Aquí buscamos por "Java"
      System.out.println(resultado); // Imprime el resultado de la búsqueda
  }
}
```

```
HTML - Manual básico de HTML para la creación de páginas web.
```

```
Cliente desconectado.
```

Si volvemos a ejecutar el cliente saldrá como desconectado del servidor

WEBGRAFÍA

Bits, C. [@ContandoBits]. (n.d.). *Cómo funciona el Modelo/arquitectura cliente servidor (explicación SIMPLE)* 💻✅. Youtube. Retrieved November 12, 2024, from https://www.youtube.com/watch?v=QnDW6JlnEcM

Ibero, J. (2021, October 28). Arquitectura Cliente/Servidor: modelo de 3 capas. *IberAsync.es – Blog de tecnología en el mundo IT*. https://iberasync.es/arquitectura-cliente-servidor-modelo-de-3-capas/

***InfoSphere Information Server 11.5.0*. (2021, February 28). Ibm.com. https://www.ibm.com/docs/es/iis/11.5?topic=components-services-tier**