Semáforos - Proyecto FTP

Objetivo

El objetivo de la guía es el de aplicar los conocimientos acerca de los semáforos en un problema real. Se presentarán algunos conocimientos acerca de cómo se manejan las operaciones utilizando el protocolo FTP y luego se dejará un problema que permita la descarga simultánea de archivos utilizando un conjunto limitado de conexiones.

Competencias

- Conocer cómo utilizar el protocolo FTP mediante Java
- Desarrollar destreza en el uso de los semáforos
- Aplicar los semáforos en la solución de un problema real de descarga concurrente de archivos mediante FTP

FTP: Protocolo de transferencia de archivos

El protocolo de transferencia de archivos (FTP) se utiliza para transferir archivos entre el cliente del ftp y el servidor del ftp. Es la manera más fácil de transferir archivos entre las computadoras (el cliente FTP y el servidor FTP). Una conexión básica FTP necesita un ordenador remoto (el cliente FTP) que llama a un servidor FTP. Utilizando el Cliente FTP para descargar contenido desde el servidor.

Pasos a seguir para descargar archivos desde el servidor FTP:

1. Necesitamos realizar un inicio de sesión (LOGIN) al Servidor de FTP con un nombre de usuario y una clave con el objeto de ganar acceso al sistema remoto.
2. Debemos cambiar al modo de conexión de datos pasivo (LocalPassiveMode) desde servidor-a-cliente a cliente-a-servidor. Puede haber algunos problemas de conexión si este método no se invoca.
3. Especificar el directorio del servidor remoto que se usará para descargar los archivos del servidor.
4. Descargue el archivo desde el servidor ftp.

Ejemplo de código para descargar archivos a un equipo local desde el servidor ftp:

```
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;

import org.apache.commons.net.ftp.FTP;
import org.apache.commons.net.ftp.FTPClient;

public class FtpFile Download{
  public static void main( String[] args) {
       String serverAddress = "www.peru-software.com"; // ftp server address
       int port = 21; // ftp uses default port Number 21
       String username = "pp20172@peru-software.com"; // username of ftp server
       String password = "fisi20172"; // password of ftp server

       FTPClient ftpClient = new FTPClient();
       try {

              // realizar el login
             ftpClient.connect(server Address, port);
              ftpClient.login(username,password);

              ftpClient.enterLocalPassiveMode();
              ftpClient.setFileType(FTP.BINARY_FILE_TYPE/FTP.ASCII_FILE_TYPE);

              String remoteFilePath = "/so01.pptx";

              File localfile = new File("D:/so01.pptx");
              OutputStream outputStream = new BufferedOutputStream(new
FileOutputStream(localfile));
              boolean success = ftpClient.retrieveFile(remoteFilePath, outputStream);
              outputStream.close();

              if (success) {
                 System.out.println("Archivo descargado.");
              } else {
                System.out.println("Archivo no se pudo descargar");
             }
       } catch (IOException ex) {
              System.out.println("Ocurrio un error en la descarga : " +
ex.getMessage());
       } finally {
              try {
              if (ftpClient.isConnected()) {
                     ftpClient.logout();
                     ftpClient.disconnect();
              }
              } catch (IOException ex) {
```

```
                    ex.printStackTrace();
                }
            }
        }
    }
```

Nota: Aquí se utiliza el método retrieveFile el cual no presenta información acerca del porcentaje descargado del archivo, sería mejor utilizar el método retrieveFileStream para mejorar la información de la descarga (por ejemplo el porcentaje de descarga del archivo).

Explicación del programa anterior:

- El ejemplo es un programa que se conecta a un servidor FTP identificado por www.peru-software.com las credenciales para la conexión son UserName=pp20172@peru-software.com y Clave=fisi20172.
- Una vez realizada la conexión se coloca el modo pasivo y se indica cual es el archivo a descargar.
- Se procede a crear el archivo local donde se guardará lo descargado.
- Se procede a realizar la descarga y una vez terminada se cierra el archivo de salida.
- Se pregunta si la descarga fue exitosa y se informa.
- Si hay errores se imprime el mensaje de error
- Por último se cierra la conexión.

Apache Commons Net API para descargar archivos mediante el protocolo FTP

La clase org.apache.commons.net.ftp.FTPClient proporciona dos métodos para descargar archivos desde un servidor FTP:

boolean retrieveFile(String remote, OutputStream local): Este método recupera un archivo remoto cuya ruta es especificada por el parámetro remote y lo escribe en el OutputStream especificado por el parámetro local. El método devuelve true si la operación se completó satisfactoriamente, o false en caso contrario. Este método es adecuado en caso de que no nos importe cómo se escribe el archivo en el disco, simplemente deje que el sistema utilice el OutputStream dado para escribir el archivo. Debemos cerrar el OutputStream después del retorno del método.

InputStream retrieveFileStream(String remote): Este método no utiliza un OutputStream, sino que devuelve un InputStream que podemos usar para leer los bytes del archivo remoto. Este método nos da **más control** sobre cómo leer y escribir los datos. Pero hay dos puntos importantes al usar este método:

- Se debe llamar al método completePendingCommand() después de la descarga para finalizar la transferencia de archivos y comprobar su valor de retorno para verificar si la descarga se realizó correctamente.

- Debemos cerrar el InputStream explícitamente.

¿Qué método se utiliza adecuado para usted? Aquí hay algunos consejos:

- El primer método proporciona la forma más sencilla de descargar un archivo remoto, ya que sólo con pasar un OutputStream se escribirá el archivo remoto en el disco local.
- El segundo método requiere que se escriba más código, ya que tenemos que crear un nuevo OutputStream para escribir el contenido del archivo mientras leemos sus arrays de bytes del InputStream devuelto. Este método es útil cuando queremos medir el progreso de la descarga, es decir, cuántos porcentaje del archivo se han transferido. Además, tenemos que llamar a completePendingCommand () para finalizar la descarga.

Ambos métodos lanzan una excepción IOException (o uno de sus descendientes, FTPConnectionClosedException y CopyStreamException). Por lo tanto, asegúrese de manejar estas excepciones al llamar a los métodos.

Además, los dos métodos siguientes deben invocarse antes de llamar a los métodos retrieveFile() y retrieveFileStream():

void enterLocalPassiveMode(): este método cambia el modo de conexión de datos de servidor-a-cliente (modo predeterminado) a cliente-a-servidor, el cual puede pasar a través del cortafuegos. Puede haber algunos problemas de conexión si este método no se invoca.

boolean setFileType (int fileType): este método establece el tipo de archivo que se va a transferir, ya sea como archivo de texto ASCII o archivo binario. Se recomienda establecer el tipo de archivo en FTP.BINARY_FILE_TYPE, en lugar de FTP.ASCII_FILE_TYPE.

En la referencia Bibliográfica 2 hay un ejemplo del uso de ambos métodos.

Tareas

- Cambie el ejemplo de esta guía, de modo que utilice el método retrieveFileStream para descargar el archivo informando el porcentaje de descarga.
- Revise la documentación y el Internet para verificar cómo recuperar el listado de un directorio remoto almacenado en un servidor FTP.

Proyecto

Con los conocimientos adquiridos y en grupos se procederá a construir un programa que haga lo siguiente:
Se conecte al servidor ftp del ejemplo
Recupere la lista de todos los archivos que contiene el directorio remoto.
Lance un hilo por cada archivo de la lista, este hilo debe descargar el archivo en un directorio local.

Mientras se realiza la descarga se debe ir pintando el porcentaje de descarga en incrementos de 10% aproximadamente.
Se debe limitar la cantidad de conexiones simultáneas a 3 descargas. Se debe usar un semáforo contador para limitar las conexiones.
En caso de error de alguna descarga se debe informar.
Al finalizar todas las descargas se debe imprimir la cantidad de archivos descargados con éxito y la cantidad de archivos que no se pudieron descargar. Utilice un semáforo mutex cuando actualice estos contadores.

Bibliografía

1. Sanjay Saini, OodlesTechnologies, Download files from FTP Server using Java Program
   http://www.oodlestechnologies.com/blogs/Download-files-from-FTP-Server-using-Java-Program
2. Code Java. Java Java FTP file download tutorial and example
   http://www.codejava.net/java-se/networking/ftp/java-ftp-file-download-tutorial-and-example
3. Apache Commons Net Library, https://commons.apache.org/proper/commons-net/
4. Descarga de la Libreria Apache Commons Net API
   https://commons.apache.org/proper/commons-net/download_net.cgi
5. Documentación de la clase FTPClient de Apache Commons Net API
   https://commons.apache.org/proper/commons-net/javadocs/api-3.6/index.html
6. Code Java, Java FTP listar archivos y directorios
   http://www.codejava.net/java-se/networking/ftp/java-ftp-list-files-and-directories-example