



Servicios **API**

Carla, Patricia y Andrea R

Índice

1. HTTP

1.1. Definición

1.2. ¿Para qué sirve?/ Funciones principales

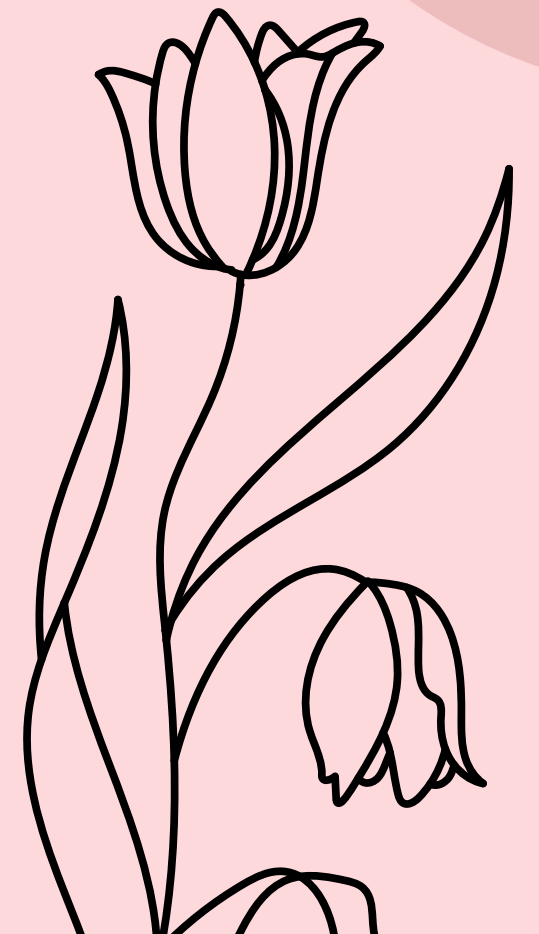
1.3. Código

2. FTP

2.1. Definición

2.2. ¿Para qué sirve?/ Funciones principales

2.3. Código



1. HTTP





Definición:

HTTP (HyperText Transfer Protocol): Es un protocolo de comunicación utilizado para la transferencia de información en la web. Permite la conexión entre navegadores y servidores para acceder a páginas web, imágenes, videos y otros recursos.

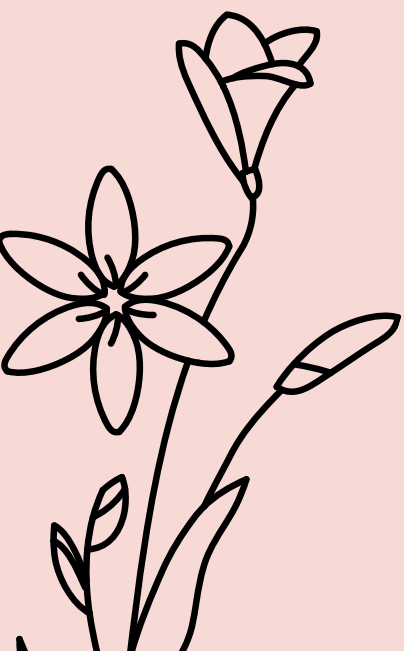


¿Para qué sirve?

HTTP se usa para cargar páginas web y transmitir información en Internet. Es el protocolo base para la navegación en sitios web.

Funciones principales:

- Permite la comunicación entre navegadores y servidores web.
- Facilita la carga de páginas, imágenes, videos y otros recursos.
- Se usa en aplicaciones web para enviar y recibir datos.
- Su versión segura (HTTPS) protege la información mediante cifrado.



NUESTRO TRADUCTOR



The image shows a web-based translator interface. At the top, there is a light blue header bar containing a button labeled "Salir". Below the header, on the left side, is a button labeled "Limpiar". The main area features two dropdown menus: "Idioma de Origen:" set to "Francés" and "Idioma de Destino:" set to "Inglés". Below these are two large text input areas. The left area contains the text "Martin est le meilleur professeur" and the right area contains the text "Martin is the best teacher". At the bottom center, there is a button labeled "Traducir".

En esta imagen vemos la interfaz de nuestro traductor. Hemos elegido tonos lilas y blancos. En la parte superior hay un botón para salir del aplicativo. Tenemos dos botones más que son para limpiar y/o traducir el texto que hemos introducido. Contamos con seis idiomas: español, inglés, francés, portugués, alemán e italiano.

Nos hemos inspirado en el traductor de Google, ya que queremos una apariencia simple y agradable a la vista.

Clase Language

```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package com.mycompany.serviciosapi.HTTP;
6
7  import java.util.HashMap;
8  import java.util.Map;
9
10 /**
11  *
12  * @author alumno
13  */
14 public class Language {
15
16     private static final Map<String, String> languages = new HashMap<>();
17
18     static {
19         languages.put("Español", "es");
20         languages.put("Inglés", "en");
21         languages.put("Francés", "fr");
22         languages.put("Alemán", "de");
23         languages.put("Italiano", "it");
24         languages.put("Portugués", "pt");
25     }
26
27     public static Map<String, String> getLanguages() {
28         return languages;
29     }
30
31 }
32
```

Este código define una clase llamada **Language** que gestiona un conjunto de idiomas y sus códigos asociados (por ejemplo, "Español" tiene el código "es"). Utiliza un **HashMap** para almacenar estos pares de valores, donde las claves son los nombres de los idiomas en español y los valores son sus correspondientes códigos en inglés. El bloque **static** inicializa el **HashMap** con algunos idiomas predefinidos. La función `getLanguages()` devuelve el **HashMap** con los idiomas. En resumen, la clase organiza los idiomas y sus códigos en un mapa que puede ser consultado más tarde.

Clase TranslatorApp

```
2  /*
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
4   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
5   */
6
7  package com.mycompany.serviciosapi.HTTP;
8
9  import java.awt.BorderLayout;
10 import java.awt.GridLayout;
11 import java.awt.event.ActionEvent;
12 import java.awt.event.ActionListener;
13 import java.util.Map;
14 import javax.swing.JButton;
15 import javax.swing.JComboBox;
16 import javax.swing.JFrame;
17 import javax.swing.JLabel;
18 import javax.swing.JOptionPane;
19 import javax.swing.JPanel;
20 import javax.swing.JScrollPane;
21 import javax.swing.JTextArea;
22 import javax.swing.JTextField;
23 import javax.swing.SwingUtilities;
24
25 /**
26 *
27 * @author alumno
28 */
29 public class TranslatorApp {
30
31     public static void main(String[] args) {
32         SwingUtilities.invokeLater(TranslatorApp::createAndShowGUI);
33     }
34
35     private static void createAndShowGUI() {
36         JFrame frame = new JFrame("Traductor HTTP");
37         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
38         frame.setSize(500, 300);
39         frame.setLayout(new BorderLayout());
40
41         JPanel panel = new JPanel();
42         panel.setLayout(new GridLayout(4, 2, 10, 10));
43
44         JLabel inputLabel = new JLabel("Texto a traducir:");
45         JTextField inputField = new JTextField();
46         JLabel fromLabel = new JLabel("Idioma origen:");
47         JLabel toLabel = new JLabel("Idioma destino:");
48
49         JComboBox<String> fromLang = new JComboBox<>();
50         JComboBox<String> toLang = new JComboBox<>();
51
52         Map<String, String> languages = Language.getLanguages();
53         for (String lang : languages.keySet()) {
54             fromLang.addItem(lang);
55             toLang.addItem(lang);
56         }
57
58         JButton translateButton = new JButton("Traducir");
59         JTextArea resultArea = new JTextArea();
60         resultArea.setEditable(false);
61         JScrollPane scrollPane = new JScrollPane(resultArea);
62
63         panel.add(inputLabel);
64         panel.add(inputField);
65         panel.add(fromLabel);
66         panel.add(fromLang);
67         panel.add(toLabel);
68         panel.add(toLang);
69         panel.add(translateButton);
70
71         frame.add(panel, BorderLayout.NORTH);
72         frame.add(scrollPane, BorderLayout.CENTER);
73
74         translateButton.addActionListener(new ActionListener() {
75             @Override
76             public void actionPerformed(ActionEvent e) {
77                 String text = inputField.getText();
78                 String sourceLang = languages.get(fromLang.getSelectedItem().toString());
79                 String targetLang = languages.get(toLang.getSelectedItem().toString());
80
81                 if (!text.isEmpty() && !sourceLang.equals(targetLang)) {
82                     String translatedText = TranslatorService.translate(text, sourceLang, targetLang);
83                     resultArea.setText(translatedText);
84                 } else {
85                     JOptionPane.showMessageDialog(frame, "Verifique los idiomas y el texto ingresado.", "Error", JOptionPane.WARNING_MESSAGE);
86                 }
87             }
88         });
89
90         frame.setVisible(true);
91     }
92 }
```


Clase TranslatorApp

Este código define una aplicación de traducción utilizando la interfaz gráfica de usuario (GUI) de Java. En la función **main**, se invoca el método **createAndShowGUI** para crear la ventana principal. La ventana contiene varios componentes, como etiquetas, campos de texto y botones. Se utilizan dos **JComboBox** para seleccionar el idioma de origen y el idioma de destino. Además, hay un campo de texto para ingresar el texto a traducir y un área de texto para mostrar el resultado. Cuando el usuario hace clic en el botón "Traducir", se obtiene el texto, los idiomas seleccionados y se realiza la traducción llamando a un servicio de traducción (**TranslatorService.translate**). Si hay algún error (como que los idiomas seleccionados sean iguales o no haya texto), se muestra un mensaje de advertencia. La interfaz está organizada utilizando un diseño de **GridLayout** y **BorderLayout**.

Clase TranslatorService

```
2  /*
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
4  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
5  */
6
7  package com.mycompany.serviciosapi.HTTP;
8
9  import java.io.BufferedReader;
10 import java.io.InputStreamReader;
11 import java.net.HttpURLConnection;
12 import java.net.URL;
13 import java.net.URLEncoder;
14 import java.nio.charset.StandardCharsets;
15 import org.json.JSONArray;
16
17 /**
18 *
19 * @author alumno
20 */
21 public class TranslatorService {
22
23     private static final String API_URL = "https://translate.googleapis.com/translate_a/single?client=gtx&sl=";
24
25     public static String translate(String text, String sourceLang, String targetLang) {
26         try {
27             String encodedText = URLEncoder.encode(text, StandardCharsets.UTF_8);
28             String urlString = API_URL + sourceLang + "&tl=" + targetLang + "&dt=t&q=" + encodedText;
29             URL url = new URL(urlString);
30             HttpURLConnection conn = (HttpURLConnection) url.openConnection();
31             conn.setRequestMethod("GET");
32
33             BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream()));
34             StringBuilder response = new StringBuilder();
35             String inputLine;
36
37             while ((inputLine = in.readLine()) != null) {
38                 response.append(inputLine);
39             }
40             in.close();
41
42             // Procesar la respuesta JSON
43             JSONArray jsonArray = new JSONArray(response.toString());
44             return jsonArray.getJSONArray(0).getString(0);
45
46         } catch (Exception e) {
47             e.printStackTrace();
48             return "Error en la traducción.";
49         }
50     }
51 }
```

Este código define un servicio de traducción que se conecta a la API de Google Translate para traducir texto. En el método **translate**, primero se codifica el texto a traducir y se construye la URL de la API con los parámetros correspondientes: el idioma de origen (**sourceLang**), el idioma de destino (**targetLang**) y el texto codificado. Luego, realiza una solicitud HTTP GET a la URL de la API. Después de obtener la respuesta, se lee y procesa el contenido JSON recibido, extrayendo la traducción del texto. Si ocurre algún error durante este proceso, se captura la excepción y se retorna un mensaje de error. Este servicio permite realizar traducciones entre diferentes idiomas utilizando la API de Google.

2. FTP





Definición:

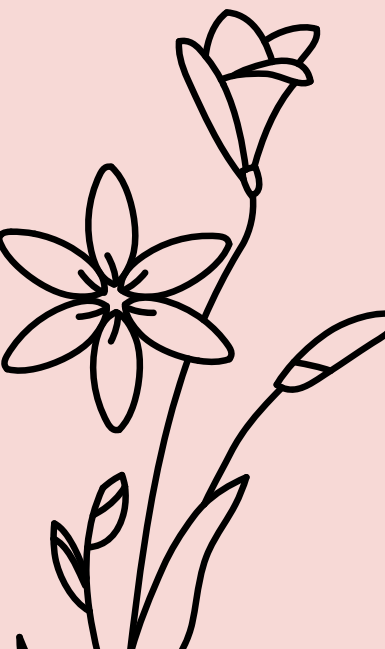
FTP (File Transfer Protocol): Es un protocolo de red diseñado para la transferencia de archivos entre un cliente y un servidor. Se utiliza para subir o descargar archivos en servidores remotos, siendo común en el mantenimiento de sitios web.



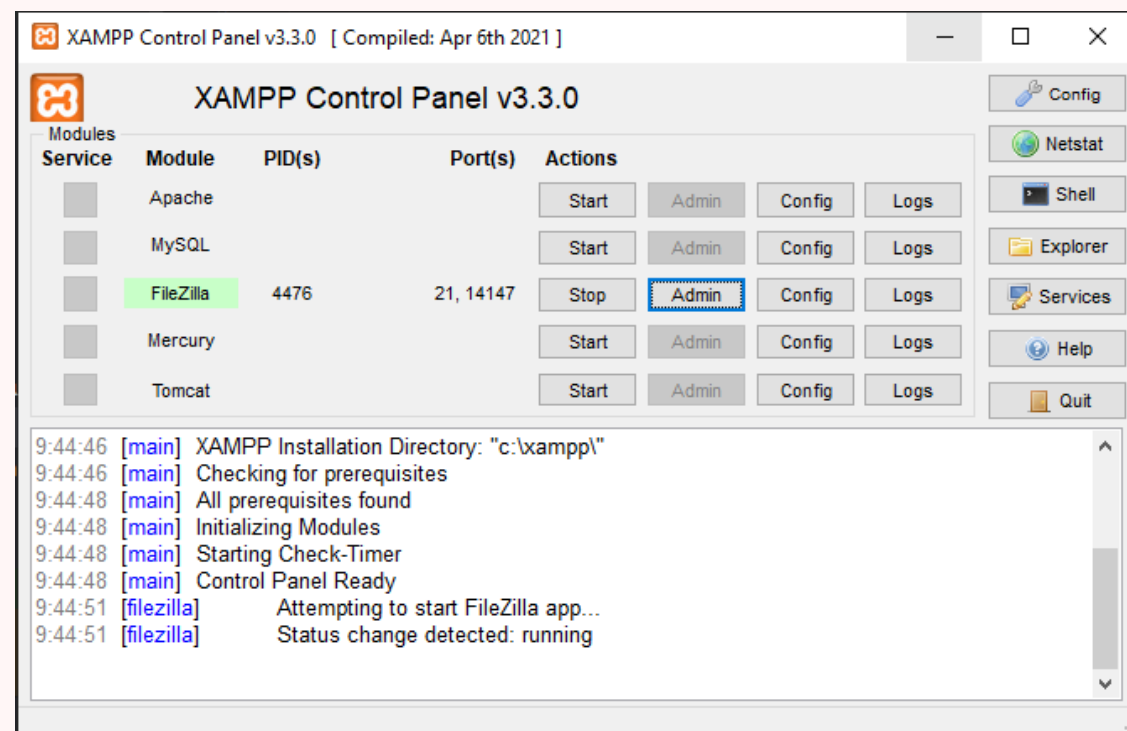
¿Para qué sirve?

FTP se usa para transferir archivos entre computadoras o servidores en una red. Es común en la gestión de sitios web y almacenamiento de archivos.

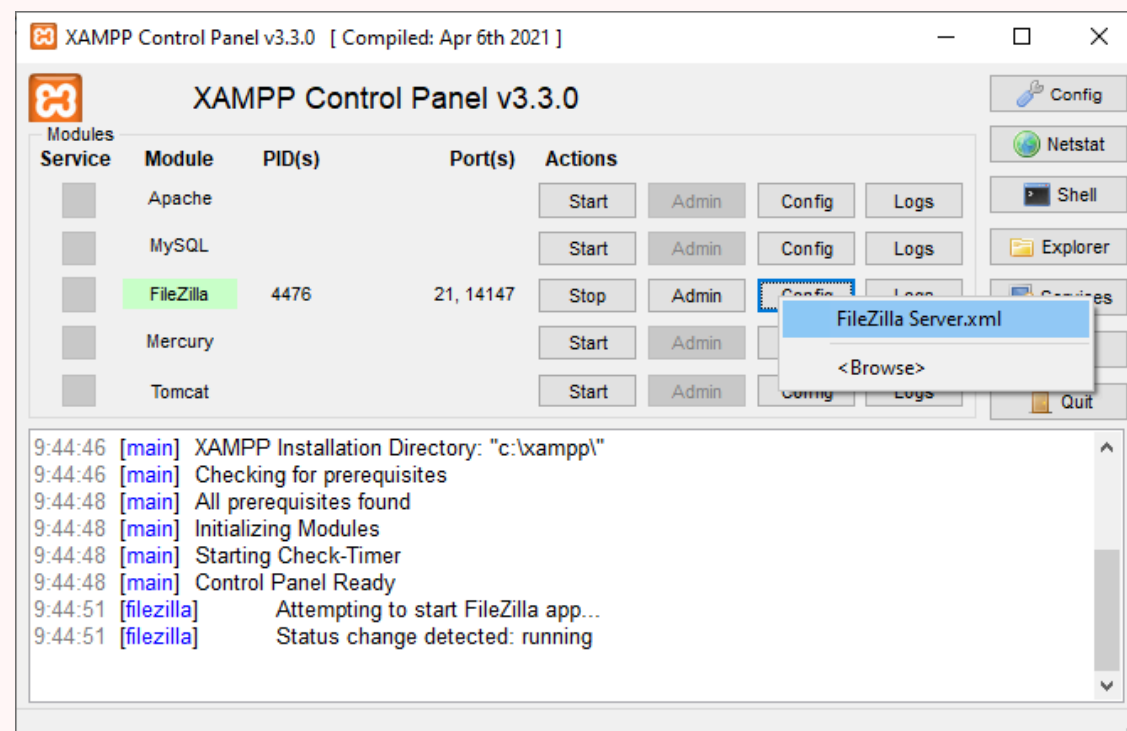
Funciones principales:

- Permite subir o descargar archivos en servidores remotos.
 - Facilita la copia de seguridad de datos.
 - Se usa para transferir grandes volúmenes de archivos de forma eficiente.
 - Puede configurarse con autenticación para mayor seguridad.
- 

SUBIDA DE ARCHIVOS



Para poder realizar el proyecto lo primero es abrir Xampp y darle a **START** en FileZilla.



Después le daremos a **CONFIG** y nos saldrán dos opciones, y le daremos a FileZilla Server.xml

SUBIDA DE ARCHIVOS

FileZilla Server: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
<FileZillaServer>
  <Settings>
    <Item name="Admin port" type="numeric">14147</Item>
  </Settings>
</FileZillaServer>
```

Nos saldrá este texto y tendremos que modificarlo. En vez de llamarlo **Admin port** se llamará **adminftp** (en nuestro caso, ya que nosotras hemos decidido llamarlo así), y el Pass será **12345** (para nosotras).

FileZilla Server.xml: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
<FileZillaServer>
  <Settings>
    <Item name="Admin port" type="numeric">14147</Item>
  </Settings>
  <Groups />
  <Users>
    <User Name="adminftp">
      <Option Name="Pass">12345</Option>
      <Option Name="Group"></Option>
      <Option Name="Bypass server userlimit">0</Option>
      <Option Name="User Limit">0</Option>
      <Option Name="IP Limit">0</Option>
      <Option Name="Enabled">1</Option>
      <Option Name="Comments"></Option>
      <Option Name="ForceSsl">0</Option>
      <IpFilter>
        <Disallowed />
        <Allowed />
      </IpFilter>
      <Permissions>
        <Permission Dir="C:\Users\alumno\Desktop">
          <Option Name="FileRead">1</Option>
          <Option Name="FileWrite">1</Option>
          <Option Name="FileDelete">0</Option>
          <Option Name="FileAppend">0</Option>
          <Option Name="DirCreate">1</Option>
          <Option Name="DirDelete">0</Option>
          <Option Name="DirList">1</Option>
          <Option Name="DirSubdirs">1</Option>
          <Option Name="IsHome">1</Option>
          <Option Name="AutoCreate">0</Option>
        </Permission>
      </Permissions>
      <SpeedLimits DLType="0" DLimit="10" ServerDLLimitBypass="0" ULType="0" ULimit="10" ServerULLimitBypass="0">
        <Download />
        <Upload />
      </SpeedLimits>
    </User>
  </Users>
</FileZillaServer>
```

Línea 1, columna 1 100% Windows (CRLF) UTF-8

SUBIDA DE ARCHIVOS

```
Source Design History
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
4   */
5   package com.mycompany.pruebaftp;
6
7   import java.awt.Desktop;
8   import java.io.File;
9   import java.io.FileInputStream;
10  import java.io.IOException;
11  import org.apache.commons.net.ftp.FTP;
12  import javax.swing.JFileChooser;
13  import javax.swing.JOptionPane;
14  import javax.swing.SwingUtilities;
15  import org.apache.commons.net.ftp.FTPClient;
16
17  /**
18   *
19   * @author alumno
20   */
21  public class InterfazFTP extends javax.swing.JFrame {
22
23      /**
24       * Creates new form InterfazFTP
25       */
26
27      private FTPClient ftpClient;
28
29      public InterfazFTP() {
30          initComponents();
31          connectToFTP(); // Se conecta al FTP al iniciar
32          SUBIR.setEnabled(false); // Desactiva el botón hasta conectar
33      }
34
35      /**
36       * This method is called from within the constructor to initialize the form.
37       * WARNING: Do NOT modify this code. The content of this method is always
38       * regenerated by the Form Editor.
39       */
40
41      private void connectToFTP() {
42          ftpClient = new FTPClient();
43          new Thread() -> {
44              try {
45                  String server = "localhost"; // Cambia esto
46                  int port = 21;
47                  String user = "adminftp";
48                  String pass = "12345";
49
50                  ftpClient.connect(server, port);
51                  if (ftpClient.login(user, pass)) {
52                      ftpClient.enterLocalPassiveMode();
53                      ftpClient.setFileType(FTP.BINARY_FILE_TYPE);
54
55                      SwingUtilities.invokeLater(() -> {
56                          Titulo.setText("Estás conectado, a continuación sube tu archivo");
57                          SUBIR.setEnabled(true); // Activa el botón tras la conexión
58                      });
59                  }
60              } catch (IOException e) {
61                  JOptionPane.showMessageDialog(null, "Error al conectar al FTP: " + e.getMessage());
62              }
63          }.start();
64      }
65  }
```


SUBIDA DE ARCHIVOS

```
59     } else {
60         throw new IOException("Error en login");
61     }
62     } catch (IOException ex) {
63         ex.printStackTrace();
64         SwingUtilities.invokeLater(() -> Título.setText("Error de conexión al servidor."));
65     }
66     }).start();
67 }
68
69 private void subirArchivo() {
70     JFileChooser fileChooser = new JFileChooser();
71     int returnValue = fileChooser.showOpenDialog(this);
72
73     if (returnValue == JFileChooser.APPROVE_OPTION) {
74         File selectedFile = fileChooser.getSelectedFile();
75         new Thread(() -> uploadFileToFTP(selectedFile)).start();
76     }
77 }
78
79 private void uploadFileToFTP(File file) {
80     try (FileInputStream fis = new FileInputStream(file)) {
81         boolean uploaded = ftpClient.storeFile(file.getName(), fis);
82         if (uploaded) {
83             SwingUtilities.invokeLater(() -> mostrarOpcionVerDocumento(file));
84             System.out.println("Archivo subido con éxito");
85         } else {
86             SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(this, "Error al subir archivo."));
87             System.out.println("Error al subir archivo: El archivo no se subió.");
88             // Mostrar más detalles
89             String replyString = ftpClient.getReplyString();
90             System.out.println("Respuesta del servidor FTP: " + replyString);
91         }
92     } catch (IOException e) {
93         e.printStackTrace();
94         System.out.println("Error al subir archivo: " + e.getMessage());
95         SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(this, "Error de conexión al subir archivo."));
96     }
97 }
```

SUBIDA DE ARCHIVOS

```
98
99
100 private void mostrarOpcionVerDocumento(File file) {
101     int option = JOptionPane.showOptionDialog(this, "El archivo se ha subido con éxito.",
102         "Archivo Subido", JOptionPane.YES_NO_OPTION, JOptionPane.INFORMATION_MESSAGE,
103         null, new String[]{"Ver documento", "No ver documento"}, "Ver documento");
104
105     if (option == JOptionPane.YES_OPTION) {
106         try {
107             Desktop.getDesktop().open(file);
108         } catch (IOException e) {
109             JOptionPane.showMessageDialog(this, "No se puede abrir el archivo.");
110         }
111     }
112
113     @SuppressWarnings("unchecked")
114     Generated Code
115
116
117
118 private void SUBIRMouseClicked(java.awt.event.MouseEvent evt) {
119     System.out.println("Botón SUBIR presionado");
120     subirArchivo();
121 }
122
123 /**
124  * @param args the command line arguments
125  */
126 public static void main(String args[]) {
127     /* Set the Nimbus look and feel */
128     Look and feel setting code (optional)
129
130     /* Create and display the form */
131     java.awt.EventQueue.invokeLater(new Runnable() {
132         public void run() {
133             new InterfazFTP().setVisible(true);
134         }
135     });
136 }
137
138 // Variables declaration - do not modify
139 private javax.swing.JButton SUBIR;
140 private javax.swing.JLabel Título;
141 private javax.swing.JLabel jLabel1;
142 private javax.swing.JPanel jPanel1;
143 // End of variables declaration
144 }
145
```

SUBIDA DE ARCHIVOS

Este programa en Java es una aplicación de escritorio que permite a los usuarios conectarse a un servidor FTP y subir archivos a él. Utiliza la biblioteca Apache Commons Net para manejar la conexión FTP y Java Swing para la interfaz gráfica.

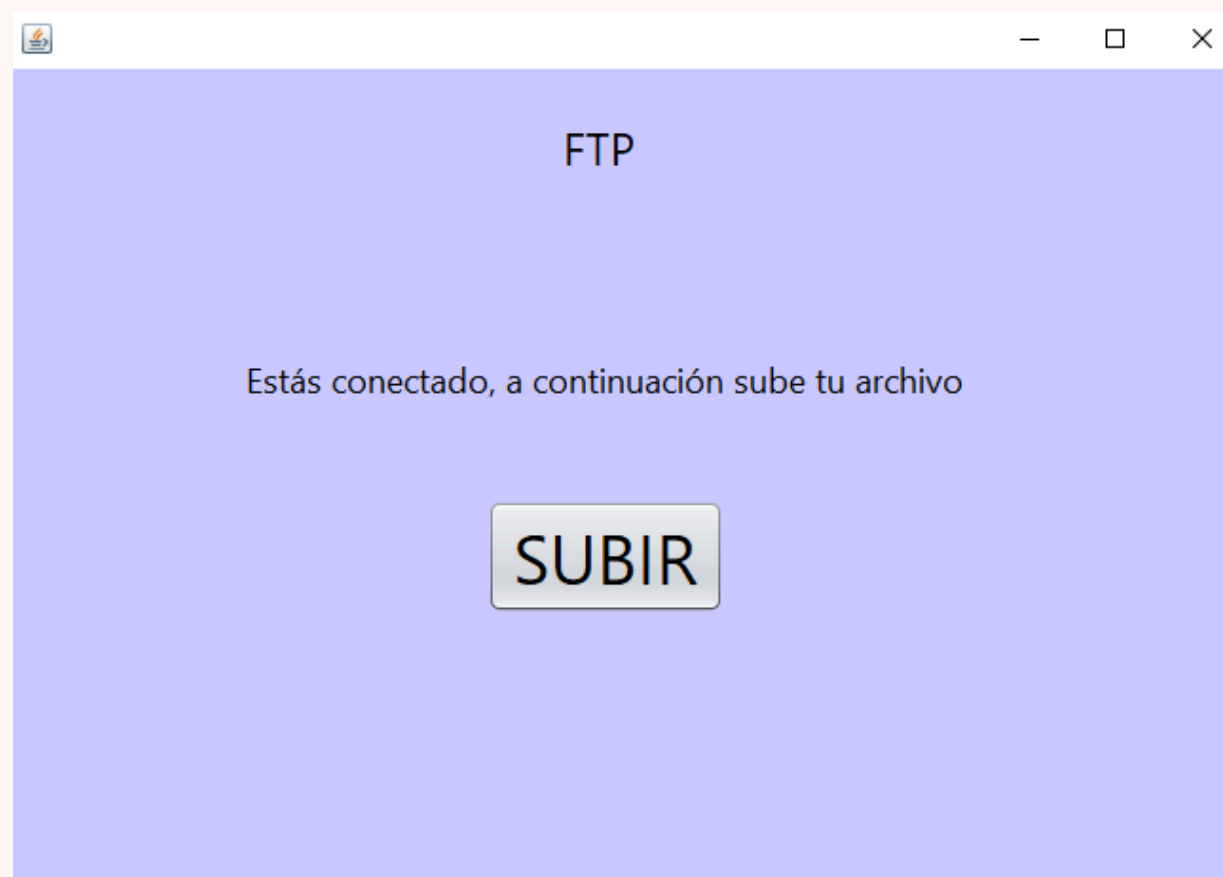
Al abrir la aplicación, esta intenta conectarse automáticamente al servidor FTP utilizando las credenciales predefinidas (adminftp y 12345). Si la conexión es exitosa, se muestra un mensaje indicando que el usuario está conectado y habilita el botón para subir archivos.

Cuando el usuario hace clic en el botón "SUBIR", se abre un explorador de archivos para seleccionar el archivo a subir. Una vez seleccionado, la aplicación lo envía al servidor FTP en un hilo secundario para no bloquear la interfaz gráfica. Si la subida es exitosa, aparece un mensaje de confirmación con la opción de abrir el archivo directamente desde la aplicación.

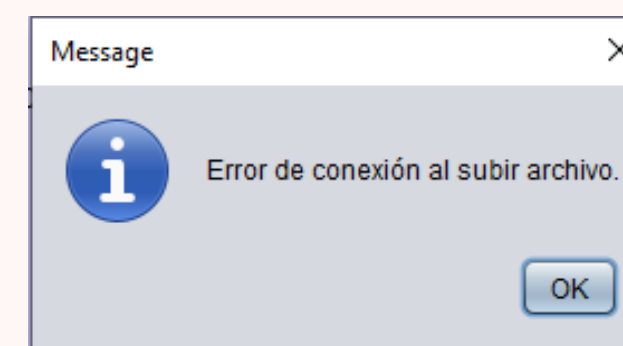
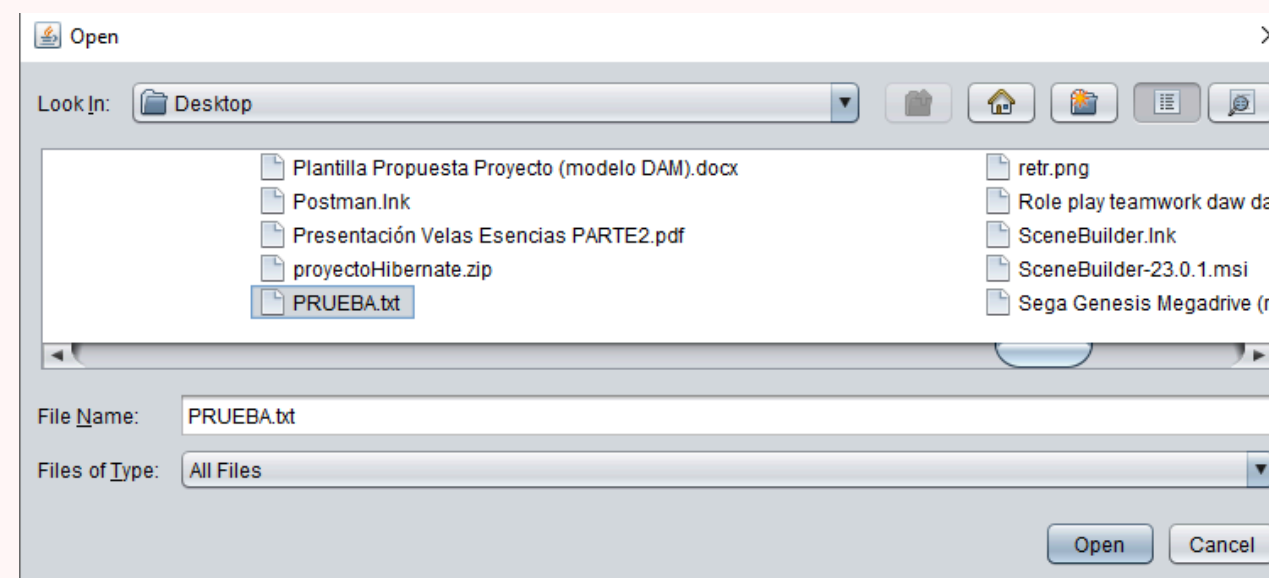
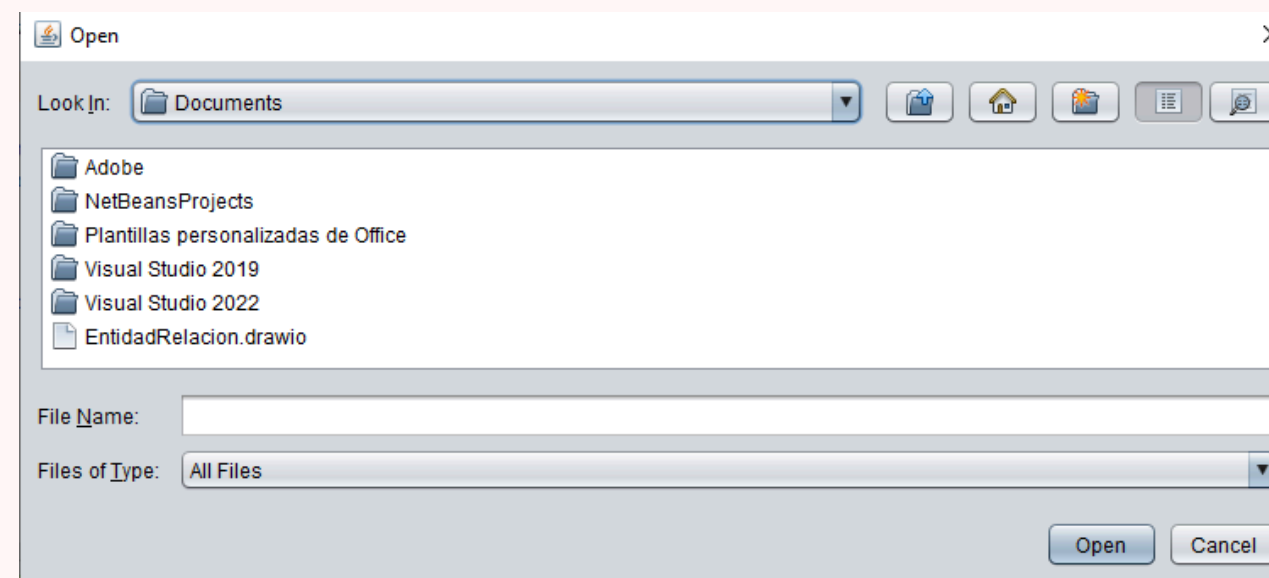
En caso de error, ya sea al conectarse o al subir el archivo, se muestra un mensaje de advertencia al usuario. La interfaz gráfica está diseñada con Swing, utilizando un JPanel con un botón (JButton) y etiquetas (JLabel) para mostrar el estado de la conexión.

Esta aplicación es útil para realizar transferencias de archivos a un servidor FTP de manera sencilla sin necesidad de utilizar software externo como FileZilla.

SUBIDA DE ARCHIVOS



Esta es nuestra Interfaz, simple y bastante intuitiva. Te salta un mensaje avisándote si estás conectado o si no lo estás. Nosotras conseguimos conectarnos. Cuando pulsas el botón **SUBIR** te abre la siguiente pantalla, para buscar el archivo que quieres subir. Nosotras probamos buscando un archivo en el escritorio. Pulsamos el archivo **PRUEBA.txt**, pero ahí es cuando empiezan los problemas ya que no conseguimos subir el archivo.



Conclusión

HTTP y FTP son dos protocolos fundamentales en la comunicación en redes. HTTP (HyperText Transfer Protocol) es el protocolo utilizado para la transferencia de páginas web y datos a través de internet, permitiendo aplicaciones como nuestro traductor, que se comunica con servidores para obtener respuestas en distintos idiomas. Por otro lado, FTP (File Transfer Protocol) está diseñado específicamente para la transferencia de archivos entre un cliente y un servidor, como en nuestro programa para subir archivos.

A pesar de haber implementado ambos protocolos, hemos encontrado dificultades con FTP en XAMPP, principalmente debido a problemas con la configuración de FileZilla Server, lo que impidió su correcto funcionamiento. Esto demuestra que mientras HTTP es más sencillo de implementar en aplicaciones modernas, FTP puede presentar más dificultades debido a restricciones de seguridad y configuraciones específicas del servidor.



Gracias

Si tenéis alguna duda no dudéis en preguntar.

Esperamos que os haya gustado

