

**Universidad Da Vinci De Guatemala**

**Facultad de Ingeniería, Industria y Tecnología**

**Carrera: Ingeniería de Sistemas**



**FACULTAD DE  
INGENIERÍA**

**UNIVERSIDAD DA VINCI  
DE GUATEMALA**

## **Proyecto No. 1 Analizador léxico**

**Nombre completo: Kevin Alberto Tinay Pérez.**

**Carné: 202304533**

**Curso: Compiladores**

**Guatemala, marzo 2025**

## Manual de Usuario: Analizador Léxico para Lenguaje de Marcado Similar a HTML.

### Descripción General

Este manual describe cómo utilizar el analizador léxico desarrollado en Java 21 con JFlex. El sistema es capaz de identificar y analizar etiquetas y contenido dentro de un lenguaje de marcado similar a HTML. El analizador recibe un archivo de entrada (entrada.txt) y genera un archivo de salida con la lista de tokens detectados.

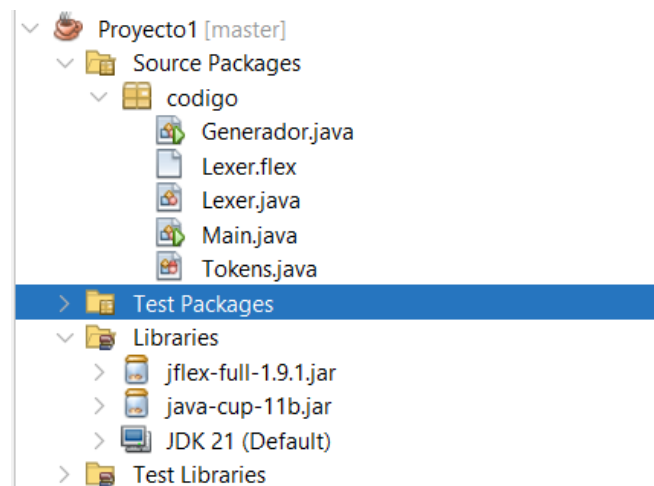
### Requisitos del Sistema

- **Java Development Kit (JDK) 21:** Asegúrate de tener instalado JDK 21 en tu sistema.
- **NetBeans IDE:** El proyecto fue desarrollado en NetBeans, pero puedes usar cualquier IDE compatible con Java.
- **JFlex:** Herramienta para generar analizadores léxicos en Java.
- Es necesario poder cargar las librerías necesarias en nuestro proyecto para que se puedan ejecutar correctamente nuestros archivos.

### Estructura del Proyecto

El proyecto contiene los siguientes archivos y directorios:

- **Lexer.flex:** Archivo de especificación léxica para JFlex.
- **Generador.java:** Clase que genera el archivo Lexer.java a partir de lexer.flex.
- **Tokens.java:** Clase que define los tipos de tokens que el analizador puede reconocer.
- **Main.java:** Clase principal que ejecuta el analizador léxico.
- **Lexer.java:** Su propósito principal es escanear el archivo de entrada (entrada.txt), identificar patrones definidos en lexer.flex y clasificar cada fragmento de texto en tokens según las reglas especificadas.
- **entrada.txt:** Archivo de entrada que contiene el código en el lenguaje de marcado similar a HTML.



## Instrucciones de Uso

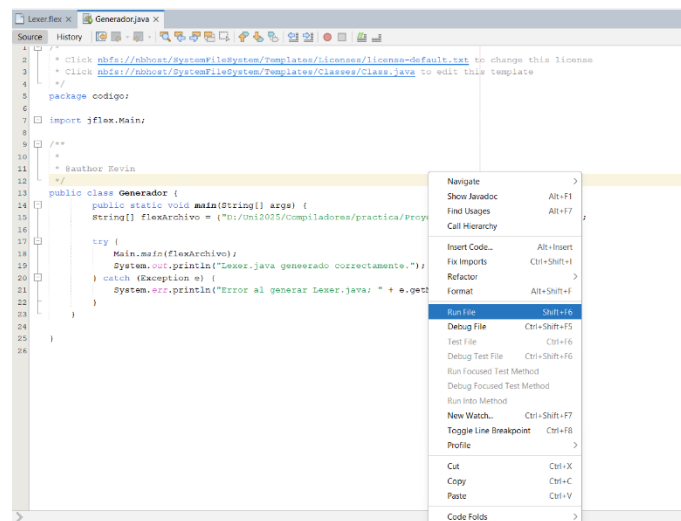
### 1. Configuración del Entorno

1. **Instalar JDK 21:** Si no lo tienes instalado, descárgalo e instálalo desde Oracle JDK.
2. **Instalar NetBeans:** Descarga e instala NetBeans desde NetBeans.
3. **Configurar JFlex:** Asegúrate de que JFlex esté configurado en tu entorno. Puedes descargarlo desde JFlex.

### 2. Generar el Analizador Léxico

#### 1. Generar Lexer.java:

- Abre el archivo Generador.java en NetBeans.
- Ejecuta la clase Generador.java. Esto generará el archivo Lexer.java a partir de lexer.flex.
- Para ejecutar la clase Generador.java das clic derecho y clic en Run File esto nos generará el archivo Lexer.java.

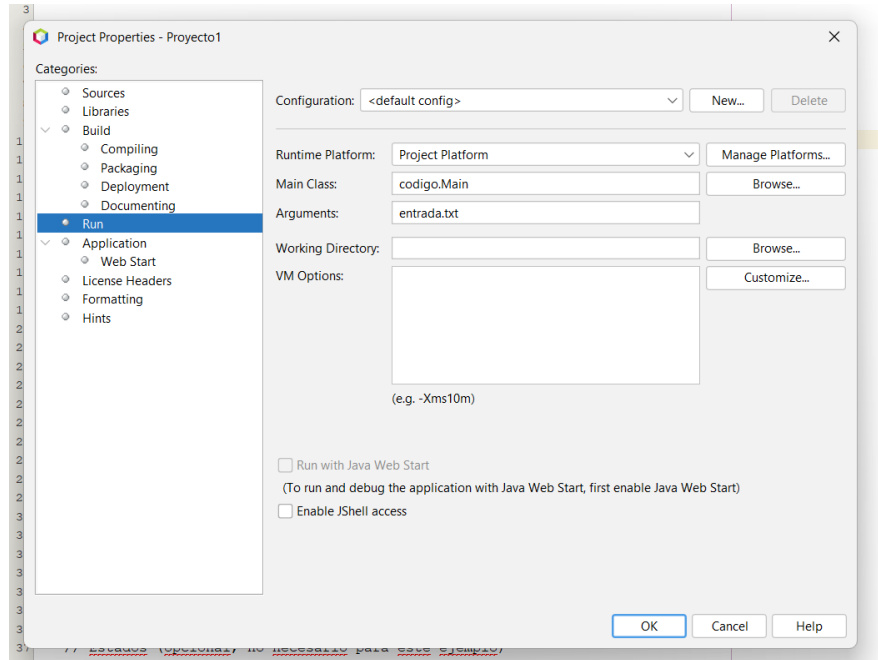


### 3. Ejecutar el Analizador Léxico

#### 1. Preparar el Archivo de Entrada:

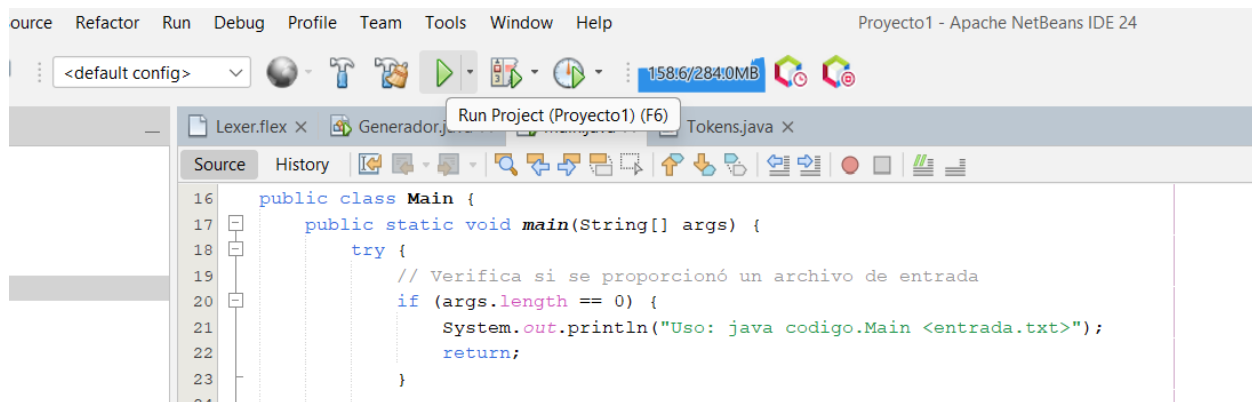
- Abre el archivo entrada.txt y escribe el código en el lenguaje de marcado similar a HTML que deseas analizar.
- Guarda los cambios en entrada.txt y ubicar el archivo en nuestra carpeta raíz para ser detectado por el ejecutor de nuestro analizador.
- Es importante configurar nuestro entorno para que podamos leer correctamente el archivo de entrada.

- Damos clic derecho en nuestro proyecto y seleccionamos PROPERTIES (normalmente es la última opción).
- Nos dirigimos a la parte de \*Run y como Main Class configuramos el archivo codigo.Main, luego en Arguments escribimos el nombre de nuestro archivo de entrada, en este caso entrada.txt.



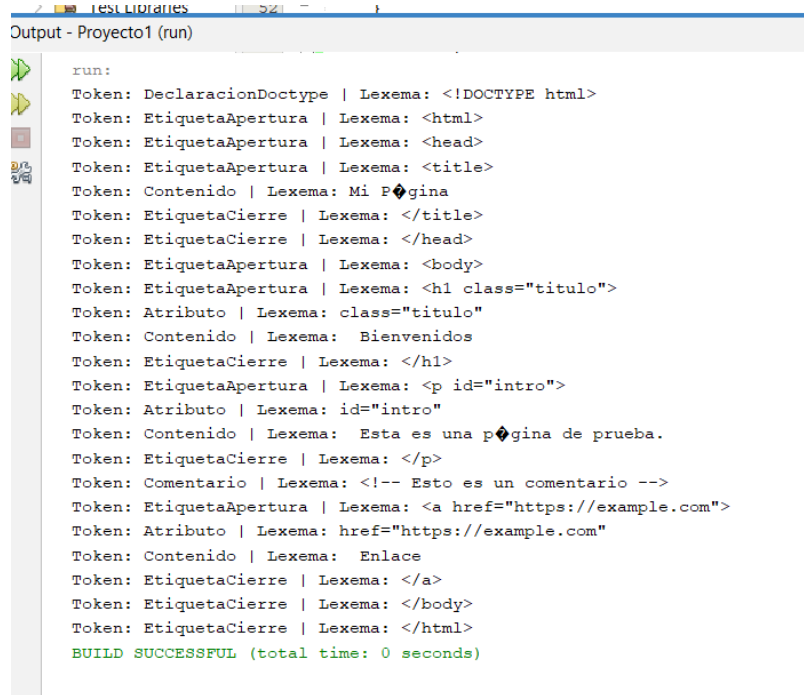
## 2. Ejecutar el Analizador:

- Abre la clase Main.java en NetBeans.
- Ejecuta la clase Main.java. Esto leerá el contenido de entrada.txt, lo analizará y generará el archivo salida.txt con la lista de tokens detectados.
- Basta con dar clic en el botón de Run Project para que se ejecute.



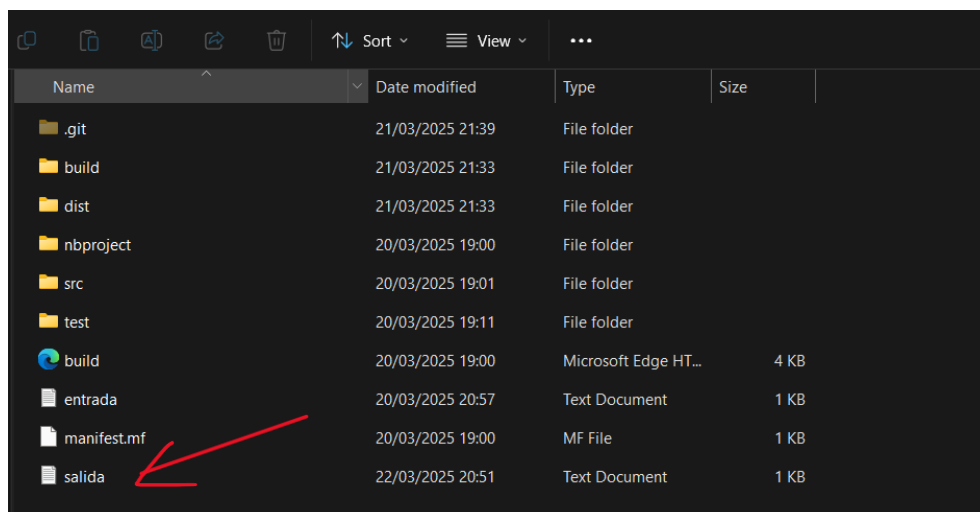
#### 4. Verificar la Salida

- En este último paso verificamos que la salida nos muestre los tokens generados en donde indique cada línea su tipo y valor establecidos anteriormente.



```
run:
Token: DeclaracionDoctype | Lexema: <!DOCTYPE html>
Token: EtiquetaApertura | Lexema: <html>
Token: EtiquetaApertura | Lexema: <head>
Token: EtiquetaApertura | Lexema: <title>
Token: Contenido | Lexema: Mi P gina
Token: EtiquetaCierre | Lexema: </title>
Token: EtiquetaCierre | Lexema: </head>
Token: EtiquetaApertura | Lexema: <body>
Token: EtiquetaApertura | Lexema: <h1 class="titulo">
Token: Atributo | Lexema: class="titulo"
Token: Contenido | Lexema: Bienvenidos
Token: EtiquetaCierre | Lexema: </h1>
Token: EtiquetaApertura | Lexema: <p id="intro">
Token: Atributo | Lexema: id="intro"
Token: Contenido | Lexema: Esta es una p gina de prueba.
Token: EtiquetaCierre | Lexema: </p>
Token: Comentario | Lexema: <!-- Esto es un comentario -->
Token: EtiquetaApertura | Lexema: <a href="https://example.com">
Token: Atributo | Lexema: href="https://example.com"
Token: Contenido | Lexema: Enlace
Token: EtiquetaCierre | Lexema: </a>
Token: EtiquetaCierre | Lexema: </body>
Token: EtiquetaCierre | Lexema: </html>
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Tambi n verificamos que se nos haya generado el archivo de salida.txt donde nos muestra los tokens generados.
- Nos vamos a nuestra carpeta ra z y ah  encontraremos este archivo, el cual podremos abrir.
  - Est  ubicado en la misma ubicaci n de nuestro archivo de entrada.txt.



## **Solución de Problemas**

- **Problema:** El archivo Lexer.java no se genera.
  - **Solución:** Asegúrate de que lexer.flex esté correctamente configurado y que Generador.java se ejecute sin errores.
- **Problema:** La salida no se genera o está vacío.
  - **Solución:** Verifica que entrada.txt contenga el código correcto y que Main.java se ejecute correctamente.

**Conclusión:**

Este manual proporciona una guía paso a paso para utilizar el analizador léxico desarrollado en Java 21 con JFlex. Siguiendo estas instrucciones, podrás analizar archivos de entrada en un lenguaje de marcado similar a HTML y generar una lista de tokens detectados en el archivo de salida.

**Anexo:**

<https://github.com/KevinT812/Proyecto1.git>

Se comparte el enlace al repositorio donde están cada archivo utilizado para este proyecto.