# MANUAL TECNICO

## ANALIZADOR LEXICO

## DESCRIPCIÓN

Este analizador léxico se encarga de procesar entradas proporcionadas por el usuario y convertirlas en paneles de color, dependiendo del tipo de entrada. Cada tipo de token o lexema se detalla en el manual de usuario, proporcionando una representación visual clara y precisa.

## Estructura:

Este programa de analizador léxico cuenta con 5 paqueterías:

1. **Vistas:** representan la parte frontal que interactúa con el usuario.
2. **Analizador Léxico:**Contiene clases que se encargan del modelo de tokens , lexemas así como la clase de tokenizador y analizador léxico los cuales hacen uso de los autómatas.
3. **Controladores:**Clase de validación de datos con la parte frontal.
4. **Autómatas:** Contiene la lógica de cada token que evalúa que tipo de token es siguiendo las características de autómatas finitos deterministas (AFD).
5. **Utilitaria:** Clases de adaptación como la generación de tipos de color dependiendo del tipo token , clase generador de imágenes con apoyo con graphViz e enumeraciones

El programa de analizador léxico contiene 35 clases distribuidas en las diferentes paqueterías, en la cual la que contiene mas clases: paquetería Autómatas.

## Tecnologías usadas

1. **Backend:** Lógica del programa desarrollada en Java 21.

2. **Frontend:** Interfaz gráfica implementada con Java Swing.

3. **Generación de Gráficas:** Uso de Dot - GraphViz versión 2.43.0.

4. **Distribución:** Archivo ejecutable empaquetado en formato JAR.

5. **Documentación:** Manuales generados utilizando LibreOffice.

6. **Nota:** No se utilizaron expresiones regulares como Regex o Matcher en el desarrollo.
7. Nota: Los automatas que se describen acontinuacion son de tipo finito Determinista (AFD)

# AUTOMATAS IMPORTANTES

## AUTÓMATA TOKENIZADOR

Este autómata esta diseñado para analizar una cadena de texto y dividirla en "lexemas". Este proceso es esencial en el programa de analizador léxico ya que es el encargado de separar los posibles tokens que posteriormente seran analizados.

Este componente funciona como un autómata finito, recorriendo la cadena de entrada y cambiando de estado según el carácter procesado y su contexto. Los estados y transiciones del autómata permiten identificar diversos tipos de lexemas, como palabras, espacios en blanco, o tokens especiales.

**Estados del Autómata:**

El autómata cuenta con los siguientes estados definidos en la enumeración Estado:

- **INICIO:** Estado inicial donde se espera el comienzo de un nuevo lexema.
- **LEXEMA:** Estado donde se construye un lexema (tokens).
- **ESPACIO_EN_BLANCO:** Estado activado al encontrar un espacio en blanco o un salto de línea.
- **TOKEN_ESPECIAL:** Estado para manejar tokens especiales con una sintaxis específica, como "Square.Color(...)". El autómata permanece en este estado hasta que se completa el token.
- **FINAL:** Estado final del autómata, sin operación específica este estado representa el estado de aceptacion del automata.

**Transiciones del Autómata**

El método obtenerLexemas que retorna una lista de objetos de tipo “ Lexema “ define las siguientes transiciones:

estado **INICIO**:

- Si se encuentra un espacio: cambia a ESPACIO_EN_BLANCO.
- Si se detecta el inicio de un token especial: cambia a TOKEN_ESPECIAL y comienza a construir el token.
- Para otros caracteres: cambia a LEXEMA e inicia la construcción del lexema.

estado **LEXEMA**:

- Si se encuentra un espacio: añade el lexema actual a la lista y cambia a ESPACIO_EN_BLANCO.
- Si se encuentra un token especial: cambia a TOKEN_ESPECIAL sin agregar el lexema actual.
- Otros caracteres: continúa en LEXEMA.

estado **ESPACIO_EN_BLANCO**:

- Si hay un salto de línea: actualiza la posición.
- Si se encuentra un carácter no en blanco: cambia a LEXEMA y comienza un nuevo lexema.

estado **TOKEN_ESPECIAL**:

- Continúa construyendo el token especial hasta que se alcanza su cierre.
- Una vez completado, agrega el token a la lista y regresa a INICIO.

**TABLA DE TRANSICIONES:**

| Estado Actual | Carácter de Entrada | Nuevo Estado |
|---|---|---|
| INICIO | “ “ | ESPACIO_EN_BLANCO |
| INICIO | S | TOKEN_ESPECIAL |
| INICIO | Cualquier otro carácter | LEXEMA |
| LEXEMA | “ “ | ESPACIO_EN_BLANCO |
| LEXEMA | S | TOKEN_ESPECIAL |
| LEXEMA | Cualquier otro carácter | LEXEMA |
| ESPACIO_EN_BLANCO | “ “ O “\n” | ESPACIO_EN_BLANCO |
| ESPACIO_EN_BLANCO | Cualquier otro carácter | LEXEMA |
| TOKEN_ESPECIAL | Caracteres token especial | TOKEN_ESPECIAL |
| TOKEN_ESPECIAL | ) | INICIO |
| FINAL | | FINAL ( estado aceptación ) |

## AUTOMATA IDENTIFICADORES

El autómata para identificar tokens de tipo Identificador es el encargado de validar si una cadena de caracteres es un identificador válido según ciertas reglas léxicas. un identificador válido debe comenzar con una letra (A-Z, a-z) y puede contener letras, dígitos (0-9) o guiones bajos (‘_’) estas son las reglas de restricción para este autómata.

**Estados del Autómata**

El autómata se define con los siguientes estados en la enumeración Producción:

- S0: Estado inicial donde solo se aceptan letras como el primer carácter del identificador.
- S1: Estado de aceptación donde se permite cualquier letra, dígito, o guion bajo.

- ERROR: Estado de error que indica que la cadena no es un identificador válido.

**Transiciones del Autómata**

El autómata posee un método principal de tipo booleano el cual contiene los estados:

S0: Si el carácter es una letra (A-Z, a-z), cambia a S1.

Cualquier otro carácter lleva a ERROR.

S1: Si el carácter es una letra, dígito, o guion bajo, permanece en S1.

Cualquier otro carácter lleva a ERROR.

**Tabla de Transiciones del Autómata**

| Estado Actual | Carácter de Entrada | Nuevo Estado |
|---|---|---|
| S0 | Letra (A-Z, a-z) | S1 |
| S0 | Cualquier otro carácter | ERROR |
| S1 | Letra (A-Z, a-z), dígito (0-9), '_' | S1 |
| S1 | Cualquier otro carácter | ERROR |
| ERROR | | ERROR |

Al presentar error el método devuelve false y se invalida que el lexema sea de este tipo de token.

## AUTÓMATA PALABRAS RESERVADAS

Este autómata de Palabras reservadas verifica si una cadena de entrada es una palabra reservada específica, como "ELSE IF". Este autómata procesa la entrada carácter por carácter y valida si sigue la estructura de una palabra reservada siguiendo las restricciones.

Este autómata se mueve a través de varios estados (S0 a S3 y ERROR), procesando cada carácter de la cadena. El autómata termina en un estado de aceptación (S1 o S3) si la cadena es válida.

**Estados del Autómata**

El autómata cuenta con los siguientes estados definidos en la enumeración Producción:

- S0: Estado inicial. Se espera una letra mayúscula.
- S1: Estado que permite letras mayúsculas o minúsculas y el carácter '.'.
- S2: Estado que se alcanza cuando se encuentra un '.' después de una secuencia válida.
- S3: Estado que permite letras mayúsculas o minúsculas después de un '.'.
- ERROR: Estado de error. Indica que la cadena no es una palabra reservada válida.

**Transiciones del Autómata**

El método esPalabraReservada de tipo boleano define las siguientes transiciones:

S0:

- Si el carácter es una letra mayúscula (A-Z), cambia a S1.
- Cualquier otro carácter lleva a ERROR.

S1:

- Si el carácter es una letra minúscula (a-z), permanece en S1.
- Si el carácter es '.', cambia a S2.
- Si el carácter es 'I', permanece en S1.
- Cualquier otro carácter lleva a ERROR.

S2:

- Si el carácter es una letra mayúscula (A-Z), cambia a S3.
- Cualquier otro carácter lleva a ERROR.

S3:

- Si el carácter es una letra mayúscula (A-Z) o minúscula (a-z), permanece en S3.
- Cualquier otro carácter lleva a ERROR.

**Tabla de Transiciones del Autómata**

| Estado Actual | Carácter de Entrada | Nuevo Estado |
|---|---|---|
| S0 | Letra mayúscula (A-Z) | S1 |
| S0 | Cualquier otro carácter | ERROR |
| S1 | Letra minúscula (a-z) | S1 |
| S1 | '.' | S2 |
| S1 | 'I' | S1 |
| S1 | Cualquier otro carácter | ERROR |
| S2 | Letra mayúscula (A-Z) | S3 |
| S2 | Cualquier otro carácter | ERROR |
| S3 | Letra mayúscula (A-Z) o minúscula (a-z) | S3 |
| S3 | Cualquier otro carácter | ERROR |
| ERROR | | ERROR |

## AUTOMATA SQUARE COLOR ESPECIAL

El automata SquareColorEspecial esta diseñado para validar si una cadena de entrada sigue un patrón de un token de la forma "Square.Color(#FFFFFF, 8, 1)". Este autómata finito determinista procesará la cadena carácter por carácter, asegurando que coincida exactamente con el formato esperado.

El autómata consta de varios estados (S0 a FINAL y ERROR) que se utilizan para validar cada parte del patrón de entrada.

**Estados del Autómata**

El autómata cuenta con los siguientes estados:

- S0 a S12: Estados que verifican la secuencia de caracteres "Square.Color".
- S13: Estado que valida el carácter (.
- S14 a S19: Estados que verifican la cadena de 6 caracteres hexadecimales después del #.
- S20: Estado que valida el carácter “,”.
- S21 a S23: Estados que validan un número entero seguido de un espacio.
- S25 a S27: Estados que verifican otro número entero después del espacio.
- FINAL: Estado final de aceptación, indica que la cadena es válida.
- ERROR: Estado de error, indica que la cadena no cumple con el formato esperado.

**Transiciones del Autómata**

El método esSquareEspecial d tipo boleano define las siguientes transiciones:

Desde S0 a S12:

- Cada estado espera un carácter específico correspondiente a la secuencia "Square.Color".

Desde S13:
- Si el carácter es #, cambia a S14.
- De lo contrario, pasa a ERROR.

Desde S14 a S19:
- Si el carácter es un dígito hexadecimal válido (0-9, A-F, a-f), avanza al siguiente estado.
- Cualquier otro carácter lleva a ERROR.

Desde S20:
- Si el carácter es ,, cambia a S21.
- De lo contrario, pasa a ERROR.

Desde S21:
- Si el carácter es un espacio, cambia a S22.
- Cualquier otro carácter lleva a ERROR.

Desde S22 a S23:
- Si el carácter es un número (0-9), permanece o avanza al siguiente estado.
- Si el carácter es ,, cambia a S25.
- Cualquier otro carácter lleva a ERROR.

Desde S25:
- Si el carácter es un espacio, cambia a S26.
- Cualquier otro carácter lleva a ERROR.

Desde S26 a S27:
- Si el carácter es un número (0-9), permanece o avanza al siguiente estado.
- Si el carácter es ), cambia a FINAL.
- Cualquier otro carácter lleva a ERROR.

**Tabla de Transiciones del Autómata**

| Estado Actual | Carácter de Entrada | Nuevo Estado |
|---|---|---|
| S0 | 'S' | S1 |
| S1 | 'q' | S2 |
| S2 | 'u' | S3 |
| S3 | 'a' | S4 |
| S4 | 'r' | S5 |
| S5 | 'e' | S6 |
| S6 | '.' | S7 |
| S7 | 'C' | S8 |
| S8 | 'o' | S9 |
| S9 | 'l' | S10 |
| S10 | 'o' | S11 |
| S11 | 'r' | S12 |
| S12 | '(' | S13 |
| S13 | '#' | S14 |
| S14 a S19 | Dígito hexadecimal (0-9, A-F, a-f) | S15 a S20 |
| S20 | ',' | S21 |
| S21 | ' ' | S22 |
| S22 | Número (0-9) | S23 |
| S23 | Número (0-9) | S23 |
| S23 | ',' | S25 |
| S25 | ' ' | S26 |
| S26 | Número (0-9) | S27 |
| S27 | Número (0-9) | S27 |
| S27 | ')' | FINAL |
| Cualquier Estado | Carácter no validod | ERROR |

AUTOMATA PALABRA FUNCION

Este automta es el que identifica y procesa cadenas de texto que pueden representar palabras reservadas, identificadores, o símbolos de operación (signos o símbolos). Este autómata recorre una cadena de entrada carácter por carácter para verificar si corresponde a una palabra reservada, un identificador , este se encarga de separar dos lexemas juntos como ejemplo “ Saludar() “ .

Este autómata utiliza otros autómatas como el de PalabrasReservadas, Identificador, y SignoSimbolo para la validacion.

**Estados del Autómata**

El autómata utiliza los siguientes estados:

- S0: Estado inicial donde se espera una letra (mayúscula o minúscula).
- S1: Estado que acepta letras mayúsculas o minúsculas y verifica si es parte de una palabra.
- S3: Estado que acepta símbolos o signos después de que una palabra ha sido validada.
- ERROR: Estado de error que indica que la cadena no cumple con el formato esperado.

**Transiciones del Autómata**

El método esPalabraFuncion define las siguientes transiciones:

S0:

- Si el carácter es una letra mayúscula o minúscula, cambia a S1.
- Cualquier otro carácter lleva a ERROR.

- S1:
    - Si el carácter es una letra mayúscula o minúscula, permanece en S1.
    - Si el carácter es un signo o símbolo válido, cambia a S3.
    - Cualquier otro carácter lleva a ERROR.

- S3:
    - Si el carácter es un signo o símbolo válido, permanece en S3.
    - Cualquier otro carácter lleva a ERROR.

**Tabla de Transiciones del Autómata**

| Estado Actual | Carácter de Entrada | Nuevo Estado |
|---|---|---|
| S0 | Letra mayúscula (A-Z) o minúscula (a-z) | S1 |
| S0 | Cualquier otro carácter | ERROR |
| S1 | Letra mayúscula (A-Z) o minúscula (a-z) | S1 |
| S1 | Signo o símbolo válido | S3 |
| S1 | Cualquier otro carácter | ERROR |
| S3 | Signo o símbolo válido | S3 |
| S3 | Cualquier otro carácter | ERROR |
| ERROR | | ERROR |

Este autómata tambien se encarga de separar dos lexemas que vengan juntos y validar que tipo de token es cada uno.