MANUAL TECNICO

ANALIZADOR LEXICO HTML , CSS , JS

DESCRIPCIÓN

Estos analizadores léxicos se encargam de procesar entradas proporcionadas por el usuario y convertirlas en un codigo html funcional.

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. **Autómatas:** Contiene la lógica de cada token que evalúa que tipo de token es siguiendo las características de autómatas finitos deterministas (AFD).
4. **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. **Repositorio en Github:** trabajado en repositorio remoto y local, cuenta con 9 ramas de funcionalidades y una rama de actualizaciones en código.

7. **Nota:** No se utilizaron expresiones regulares como Regex o Matcher en el desarrollo.
8. 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
Q0	Letra	ETIQUETA_REGLA
Q0	`+`, `>`, `~`	COMBINADOR
Q0	`.` o `#`	CLASE_ID
Q0	`%`, `(`, `)`, `,`, `;`, `{`, `}`	OTROS
Q0	Comilla simple `'`	CADENA
Q0	Número	OTROS
ETIQUETA_REGLA	`rgba(`	COLOR_RGBA
CLASE_ID	Carácter hexadecimal	COLOR
CADENA	Comilla simple de cierre	Q0
COLOR_RGBA	Cierre de `rgba()`	Q0

AUTOMATA HTML

El autómata recorre el contenido de la sección HTML, cambiando de estado en función de los caracteres encontrados y su contexto. Los estados definidos permiten identificar y procesar etiquetas HTML, atributos y el texto entre etiquetas.

Estados del Autómata:

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

- **Q0**: Estado inicial donde se espera el comienzo de una etiqueta o texto.
- **Q1**: Estado donde se procesa una etiqueta HTML.
- **Q2**: Estado donde se procesan los atributos de una etiqueta.
- **Q3**: Estado donde se procesa el texto fuera de las etiquetas.

Transiciones del Autómata

El método analizarSeccion define las siguientes transiciones:

- **Estado Q0**:
    - Si se encuentra un carácter <: se cambia al estado Q1 para procesar la etiqueta.
    - Si se encuentra cualquier otro carácter: se permanece en Q0 y se sigue construyendo el lexema (texto).
- **Estado Q1**:
    - Si se encuentra un carácter diferente a >: se sigue construyendo la etiqueta.
    - Si se encuentra el carácter >: se completa la etiqueta y se cambian los tokens relevantes, luego se regresa a Q0 o se permanece en el estado de lectura de atributos o texto.
- **Estado Q2**:
    - Si se encuentra un carácter <: se cambia nuevamente a Q1 para comenzar a procesar otra etiqueta.
    - Si se encuentra cualquier otro carácter: se continúa en Q2, procesando el texto o atributos dentro de la etiqueta.

TABLA TRANSICIONES:

Estado Actual	Carácter de Entrada	Nuevo Estado
Q0	`<`	Q1
Q0	Cualquier otro carácter	Q0
Q1	`>`	Q0
Q1	Cualquier otro carácter	Q1
Q2	`<`	Q1
Q2	Cualquier otro carácter	Q2

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 JAVASCRIPT

El autómata recorre el contenido de una sección de JavaScript, cambiando de estado en función de los caracteres encontrados y su contexto. Los estados definidos permiten identificar correctamente lexemas como identificadores, palabras reservadas, operadores, comentarios, números, cadenas y booleanos.

Estados del Autómata:

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

- **Q0**: Estado inicial donde se espera el comienzo de un nuevo lexema.
- **IDENTIFICADOR_RESERVADA**: Estado donde se construye un identificador o palabra reservada.
- **ERROR**: Estado de error donde el lexema actual no es válido.
- **SIGNO_SIMBOLO**: Estado donde se procesan operadores y símbolos.
- **COMENTARIO**: Estado donde se procesa un comentario (línea o bloque).
- **EVALUAR_SLASH**: Estado de evaluación de un posible comentario o división.
- **EVALUAR_NUMERO**: Estado donde se construye un número.
- **EVALUAR_CADENA**: Estado donde se construye una cadena de caracteres.

- **EVALUAR_BOOLEANO**: Estado donde se evalúa un valor booleano (true/false).

Transiciones del Autómata:

El método analizarSeccion define las siguientes transiciones:

- **Estado Q0:**
  - Si se detecta una letra: cambia a IDENTIFICADOR_RESERVADA e inicia la construcción de un posible identificador o palabra reservada.
  - Si se detecta un número o un guion: cambia a EVALUAR_NUMERO y comienza la construcción de un número.
  - Si se detecta un símbolo: cambia a SIGNO_SIMBOLO.
  - Si se detecta una barra /: cambia a EVALUAR_SLASH para verificar si es un comentario o un operador de división.
  - Si se detecta una comilla simple, doble, o una tilde inversa: cambia a EVALUAR_CADENA para procesar una cadena de texto.
- **Estado IDENTIFICADOR_RESERVADA:**
  - Si se detecta un carácter alfanumérico o un guion bajo: continúa en el estado y sigue construyendo el identificador o palabra reservada.
  - Si se detecta un espacio, salto de línea, o caracteres como paréntesis, puntos o

comas: evalúa si el lexema es una palabra reservada, un identificador o un booleano, y cambia a Q0 después de procesar el token.

- **Estado SIGNO_SIMBOLO:**
    - Si se detectan más símbolos, se sigue construyendo un posible operador.
    - Si se encuentra un espacio, salto de línea, o una barra, evalúa el lexema como un símbolo simple o doble (operador) y regresa a Q0.
- **Estado COMENTARIO:**
    - Si se detecta un salto de línea, el comentario se completa y el estado regresa a Q0.
- **Estado EVALUAR_NUMERO:**
    - Si se detectan dígitos o un punto, se sigue construyendo el número.
    - Si se detecta un espacio, salto de línea, o un punto y coma, el número se completa y se cambia a Q0.
- **Estado EVALUAR_CADENA:**
    - Si se encuentra la misma comilla con la que se comenzó la cadena (comilla simple, doble o tilde inversa), la cadena se completa y regresa a Q0.

Estado Actual	Carácter de Entrada	Nuevo Estado
Q0	Letra	IDENTIFICADOR_RESERVADA
Q0	Número o `-`	EVALUAR_NUMERO
Q0	Símbolos (`+`, `-`, `*`, `/`, etc.)	SIGNO_SIMBOLO
Q0	`/`	EVALUAR_SLASH
Q0	`'`, `"`, `` ` ``	EVALUAR_CADENA
IDENTIFICADOR_RESERVADA	Espacio, salto de línea, `(`, `)`, `.`	Q0
SIGNO_SIMBOLO	Más símbolos o espacio	Q0
EVALUAR_NUMERO	Espacio, salto de línea, `;`	Q0
EVALUAR_CADENA	Comillas de cierre	Q0