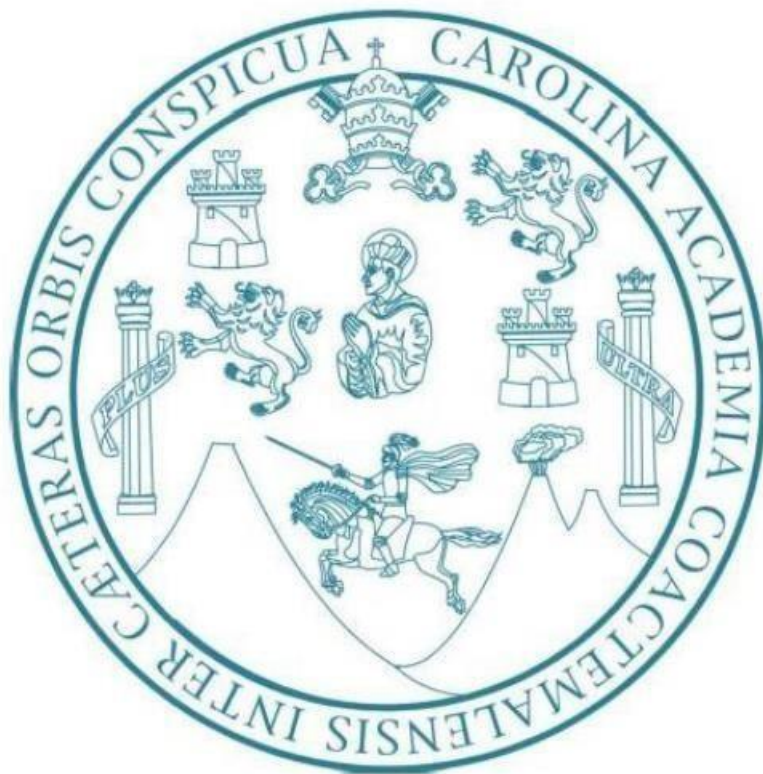


Universidad de San Carlos de Guatemala

Facultad de Ingeniería
Lenguajes Formales y de Programación
Sección: "B" Cat. Ing. Manuel Castillo
Tutor académico: Huriel Gómez

PROYECTO 1: Manual de Usuario



Bryan Estiveen Alarcón Aldana
Carnet: 201800526

ANALIZADOR LÉXICO - PROYECTO
Desarrollo de aplicación de escritorio con enfoque en el análisis léxico y sintáctico.

Manual de Usuario

Este manual tiene como finalidad el explicar cómo utilizar de manera adecuada el programa desarrollado en Python orientado a un análisis léxico/sintáctico, implementado en una interfaz gráfica.

Requerimientos:

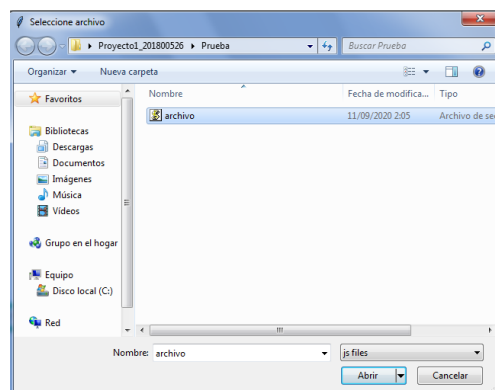
- Contar con un IDE para la correcta utilización del programa desarrollado.

Acceso al menú principal:

Primordialmente, se debe de ejecutar el programa, esta es la primera vista que se obtiene del mismo; un cuadro de entrada par alguna cadena, y con las opciones de poder utilizar el análisis léxico para HTML, CSS, JS o el sintáctico de JS.



Al hacer clic en archivo, se puede seleccionar que archivo se desea escanear, en este caso se utilizara un archivo.js



Análisis léxico:

Luego de seleccionar el archivo a analizar, se procede a presionar el botón de “análisis” para que el programa proceda a ejecutarse, lo cual en esta salida se mostraran la lista de tokens, lexema y línea al que pertenece cada uno de estos, de igual manera, se mostrara en consola si existen errores léxicos.

The screenshot shows the ML Web Editor application window. The title bar reads "Proyecto 1 - 201800526 - ML WEB EDITOR". The main editor area displays the contents of a file named "prueba.js". The code is as follows:

```

/*=====
 * =====
 * =====
 * -----ARCHIVO DE PRUEBA DE JS-----
 * =====
 * -----PATHL -> /home/user/output/js/-----
 * -----PATHW -> c:\user\output\js\-----
 * =====
 * =====
 */

/*****
 * Dentro de un archivo de tipo javascript
 * pueden encontrarse comentarios de tipo
 * multilinea o de tipo unilinea, estos
 * pueden aparecer en cualquier parte del
 * archivo de entrada tomando en cuenta que,
 * el primero es el que contiene el path del
 * directorio al cual se enviara la salida
 * ya analizada y limpiada.
 *****/

Consola: HTM CSS JavaScript SintacticoJS Analyze
-Token: 'RESERVADA' -Lexema: 'function' -Linea: 30
-Token: 'ID' -Lexema: 'session' -Linea: 30
-Token: 'PAREN_IQZ' -Lexema: '(' -Linea: 30
-Token: 'PAREN_DER' -Lexema: ')' -Linea: 30
-Token: 'LLAVE_IQZ' -Lexema: '{' -Linea: 30
-Token: 'RESERVADA' -Lexema: 'var' -Linea: 32
-Token: 'ID' -Lexema: 'success' -Linea: 32
-Token: 'IGUAL' -Lexema: '=' -Linea: 32
-Token: 'ID' -Lexema: 'sessionStorage' -Linea: 32
-Token: 'PUNTO' -Lexema: '.' -Linea: 32
-Token: 'ID' -Lexema: 'getItem' -Linea: 32
-Token: 'PAREN_IQZ' -Lexema: '(' -Linea: 32

```

At the bottom, there is a console panel with tabs for "HTM", "CSS", "JavaScript", "SintacticoJS", and "Analyze". The "Analyze" tab is active, displaying tokenization results for the code above.

```
*Python 3.8.1 Shell*
```

```
File Edit Shell Debug Options Window Help
```

```
===== RESTART: C:\Users\user\Desktop\Proyecto1_201800526\main.py =====
```

```
Comentario multilinea:
```

```
/*****  
 * ****  
 * ****  
 * ****  
 * -----ARCHIVO DE PRUEBA DE JS-----  
 * ****  
 * -----PATHL -> /home/user/output/js/-----  
 * -----PAIHW -> c:\user\output\js\-----  
 * ****  
 * ****  
 * ****  
 * ****  
 *****/  
Comentario multilinea:
```

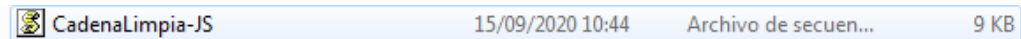
```
/*****  
 * Dentro de un archivo de tipo javascript *  
 * pueden encontrarse comentarios de tipo *  
 * multilinea o de tipo unilinea, estos *  
 * pueden aparecer en cualquier parte del *  
 * archivo de entrada tomando en cuenta que, *  
 * el primero es el que contiene el path del *  
 * directorio al cual se enviara la salida *  
 * ya analizada y limpiada. *  
 *****/
```

```
Error lexico: ^ en Fila: 42  
  
-----  
  
Comentario de una linea: // YYYYMMDD  
  
Comentario de una linea: // fecha de pedido
```

```
LIn: 5 Col: 1
```

Generación de salida:

Si el archivo contiene errores léxicos, se genera un archivo con la extensión correspondiente de cada archivo, pero este archivo generado está totalmente limpio de errores léxicos.



Reportes:

Luego de que cualquier análisis seleccionado haya sido culminado, el programa procede a generar un archivo.html con una tabla en la cual se denotan los errores léxicos en la cadena de entrada. En otro caso, imprime la cadena y tanto si es correcta o incorrecta si se analizó de manera sintáctica.



Numero	Error lexico	Fila No.
1	^	42
2	^	73
3	^	94
4	^	94
5	~	114
6	%	136
7	#	144
8	&	152
9	@	160
10	\$	169
11	;	174
12	@	186

201800526

Por último, si el archivo escaneado es de tipo.js se procede a generar un árbol de derivación en el cual se muestra la lista de tokens que contiene la cadena.

