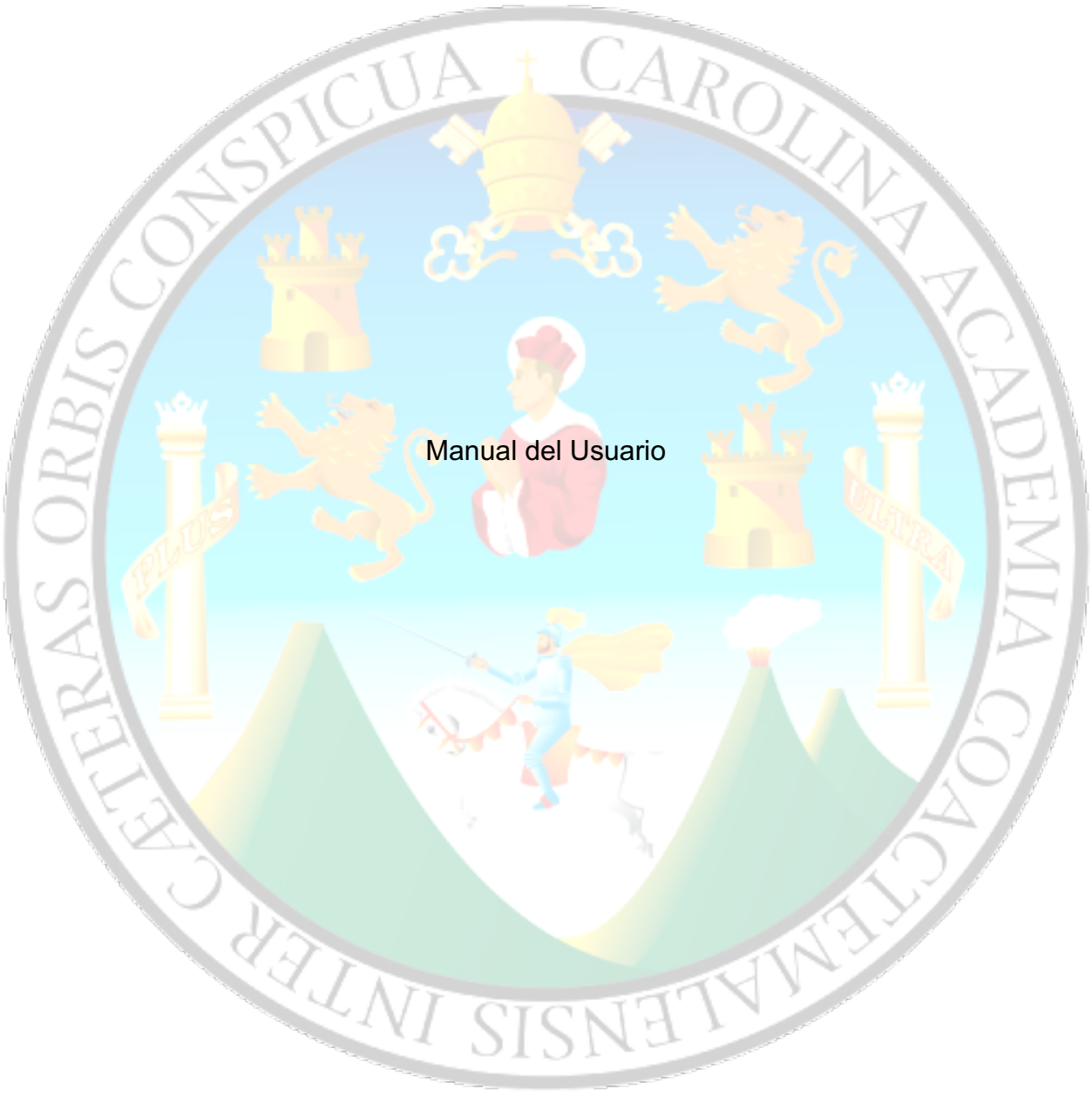
Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Organización de Lenguajes y Compiladores 1
Sección A



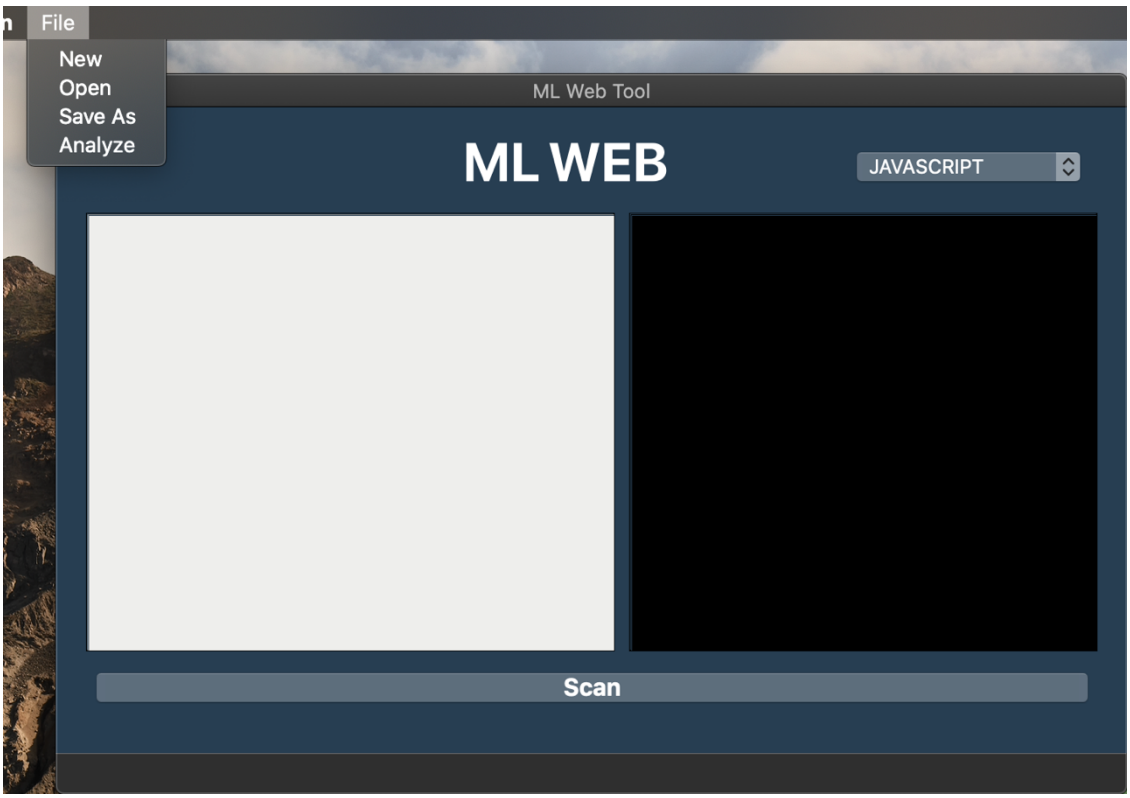
Manual del Usuario

Carlos Ojani NG Valladares
201801434

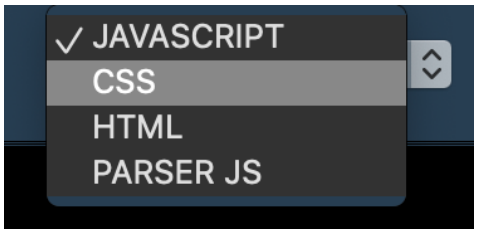
Para este proyecto se llevó a cabo un prototipo de detección de errores léxicos para los principales lenguajes orientados a la programación web. JavaScript, CSS y HTML.

El programa consiste en el análisis léxico de estos tres lenguajes, acorde a los archivos de entrada que se le proporciona al programa. Este proceso sirve para reconocer todos esos caracteres que no forman parte de cada uno de estos lenguajes. Por lo que se implementaron tres analizadores para los tres respectivos lenguajes.

Inicialmente, la aplicación cuenta con una interfaz grafica, amigable para el usuario. Consta de una consola, en donde se mostrarán errores léxicos encontrados y la bitácora de estados del CSS. También de una caja de texto que contendrá el código a analizar y una caja de selección, para elegir cuál lenguaje analizar. Por último, el botón de analizar. A continuación, la interfaz del programa:



La barra de menú en la parte superior puede realizar acciones de nuevo archivo, abrir, guardar cómo y analizar. La caja de selección cuenta con las opciones de lenguajes a analizar, a excepción de PARSER JS, que funciona para realizar el análisis sintáctico de entradas .rmt, para expresiones algebraicas.



Al momento de que el usuario ingrese, ya sea abriendo el archivo o escribiendo el código en la caja de texto, podrá clicar el botón Scan, pero siempre escogiendo en cuál lenguaje analizar. Esto es relevante por el tema de la generación de errores y porque cada lenguaje acepta diferentes entradas.

Una vez analizado el código, automáticamente se abre en el navegador web la tabla de errores y la de tokens, especificando en qué fila y columna se ubica cada elemento. A continuación, un ejemplo:

Tabla de Tokens

Token	Lexema	Columna	Fila
COMMENT	/********** ========== ========== ========== ==========ARCHIVO DE PRUEBA DE JS========== ========== ==========PATHL -> /home/user/output/js/========== ==========PATHW -> c:\user\output\js\========== ========== ========== ========== **********/	1	14
COMMENT	/********** * 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. * **********/	1	27
RESERVED_FUNCTION	function	1	30
ID	session	2	30
LEFT_PARENT	(	3	30
RIGHT_PARENT	)	4	30
LEFT_BRACE	{	5	30
ID	var	1	32

Este es un reporte de tokens generado para una entrada de archivo JavaScript.

Tabla de Errores

Num.	Error
1	La entrada "^" en la fila 47 no pertenece al lenguaje.
2	La entrada "^" en la fila 78 no pertenece al lenguaje.
3	La entrada "`" en la fila 99 no pertenece al lenguaje.
4	La entrada "`" en la fila 99 no pertenece al lenguaje.
5	La entrada "~" en la fila 119 no pertenece al lenguaje.
6	La entrada "!" en la fila 134 no pertenece al lenguaje.
7	La entrada "%" en la fila 141 no pertenece al lenguaje.
8	La entrada "#" en la fila 149 no pertenece al lenguaje.
9	La entrada "@" en la fila 166 no pertenece al lenguaje.
10	La entrada "$" en la fila 175 no pertenece al lenguaje.
11	La entrada "¿" en la fila 180 no pertenece al lenguaje.
12	La entrada "@" en la fila 192 no pertenece al lenguaje.

Estos son los errores obtenidos al analizar dicho archivo.

Para analizar archivos .css y .html, se presenta de forma similar. El reporte de tokens y errores. A diferencia de los demás, CSS generá un reporte adicional, consiste en una bitácora que muestra el recorrido del análisis de las entradas. Esto se proyecta en la consola dentro de la interfaz grafica, como se puede ver a continuación:

```
>> CSS File
>> h Estado: 0
>> ht Estado: 5
>> htm Estado: 5
>> html Estado: 5
>> Entrada html aceptada Estado: 5
>> Entrada , aceptada Estado: 5
>> b Estado: 0
>> bo Estado: 5
>> bod Estado: 5
>> body Estado: 5
>> Entrada body aceptada Estado: 5
>> Entrada { aceptada Estado: 5
>> ***** ERROR ¬ ***** Estado: 0
>> m Estado: 0
>> ma Estado: 5
>> mar Estado: 5
>> marg Estado: 5
```

Asimismo, se realiza un grafico, un diagrama de estados. Cumple la misma función que la bitácora del CSS, pero en este caso será para JS y se verá de forma gráfica. A continuación un ejemplo: