MANUAL TÉCNICO

Tecnologías usadas:
Angular 12.2.6
Node 14.17.6

Herramienta usada para generar los analizadores
Jison 0.4.18

Librerías usadas en el frontend
Angular Material 13.3.7

Lenguaje principal de desarrollo
Typescript 3.8.3

Librería usada para la generación de las imágenes
Graphviz

Librería usada para el editor
CodeMirror 5.65.3

ORGANIZACIÓN DEL PROYECTO

Por motivos de tiempo toda la aplicación fue desarrollada en el proyecto de Angular.
La parte lógica de esta aplicación se encuentra en la carpeta src/backend. En este carpeta se podrá encontrar el archivo que se utilizó para la generación de los analizadores usando Jison, también se podrá encontrar las diferentes clases que se utilizaron para el funcionamiento tanto para la generación del árbol sintáctico como para su ejecución.

ANÁLISIS DE GRAMÁTICA - ANALIZADOR LÉXICO

Palabras reservadas

Incerteza	Char	Sino	DibujarAST
Importar	Void	Para	DibujarEXP
Double	true	Mientras	DibujarTS
Boolean	false	Detener	Principal
String	Retorno	Continuar	crl
Int	Si	Mostrar	

Signos

++	**^**	**!=**	**)**
+	**<=**	**!**	**:**
-	**>=**	**~**	**,**
-	**<**	**&&**	**;**
*****	**>**	**||**	**.**
/	**==**	**|&**	
%	**=**	**(**	

Expresiones regulares para detectar valores

Cadena = "cualquierCosa"
Identificador = ([A-Z])([A-Z] | [0-9])*
Decimal = ([0-9])([0-9])*(.)([0-9])([0-9])
Entero = ([0-9])([0-9])*
Char = 'c' donde c es un carácter

ANÁLISIS DE GRAMÁTICA - ANALIZADOR SINTÁCTICO

Precedencia y asociatividad de operadores

Símbolo	Precedencia	Asociatividad
+,-	1	Izquierda
*,/,%	2	Izquierda
^	3	Derecha
-(unario)	4	No aplica
==,!=,<,>,<=,>=,~	5	No aplica
||	6	Izquierda
!&	7	Izquierda
&&	8	Izquierda
!	9	Izquierda
()	10	No aplica

En esta descripción de la gramática no se tomó en cuenta los saltos que serán necesarios para la indentación del código. El objetivo de esta descripción es presentar las diferentes instrucciones del lenguaje CRL.

```
<inicio> ::= <encabezado> <instrucciones> "EOF"
| <instrucciones>

<encabezado> ::= <importaciones> <incerteza>
| <importaciones>
| <incerteza>

<importaciones> ::= <importaciones> <importar>
| <importar>

<importar> ::= "importar" "id" "punto" "crl"

<incerteza> ::= "incerteza" "decimal"
| "incerteza" "entero"

<instrucciones> ::= <instrucciones> <instruccion>
| <instruccion>
```

```
<instruccion> ::= <declaraciones>
| <asignacion>
| <llamada_funcion>
| <retorno>
| <si>
| <sino>
| <mostrar>
| <para>
| <mientras>
| <detener>
| <continuar>
| <funcion_principal>
| <dibujar_ast>
| <dibujar_exp>
| <dibujar_ts>

<funcion_principal> ::= "void" "principal" "par_a" "par_c" "dos_p"

<dibujar_ast> ::= "dib_ast" "par_a" "id" "par_c"

<dibujar_exp> ::= "dib_exp" "par_a" <exp> "par_c"

<dibujar_ts> ::= "dib_ts" "par_a" "par_c"

<continuar> ::= "continuar"

<detener> ::= "detener"

<mientras> ::= "mientras" "par_a" <exp> "par_c" "dos_p"

<para> ::= "para" "par_a" <tipo_variable_nativa> "id" "asig" <exp> "pyc"
<exp> "pyc" <op> "par_c" "dos_p"
| "para" "par_a" <asignacion> "pyc" <exp> "pyc" <op> "par_c" "dos_p"

<op> ::= "inc"
| "dec"

<mostrar> ::= "mostrar" "par_a" <lista_expresiones> "par_c"

<si> ::= "si" "par_a" <exp> "par_c" "dos_p"

<sino> ::= "sino" "dos_p"

<retorno> ::= "retorno" <exp>
| "retorno"

<declaraciones> ::= <tipo_variable_nativa> "id"
| <tipo_variable_nativa> "id" "coma" <ids> "asig" <exp>
| <tipo_variable_nativa> "id" "asig" <exp>
| <tipo_variable_nativa> "id" "par_a" "par_c" "dos_p"
| <tipo_variable_nativa> "id" "par_a" <lista_parametros> "par_c" "dos_p"
```

```
<lista_parametros> ::= <lista_parametros> "coma" <parametro>
| <parametro>

<parametro> ::= <tipo_variable_nativa> "id"

<asignacion> ::= "id" "asig" <exp>

<ids> ::= <ids> "coma" "id"
| "id"

<tipo_variable_nativa> ::= "double"
| "boolean"
| "string"
| "int"
| "char"
| "void"

<exp> ::= "menos" <exp>
| <exp> "mas" <exp>
| <exp> "menos" <exp>
| <exp> "por" <exp>
| <exp> "div" <exp>
| <exp> "mod" <exp>
| <exp> "pot" <exp>
| "par_a" <exp> "par_c"
| <exp> "mayor" <exp>
| <exp> "menor" <exp>
| <exp> "mayor_igual" <exp>
| <exp> "menor_igual" <exp>
| <exp> "igual" <exp>
| <exp> "dif" <exp>
| <exp> "sig_inc" <exp>
| <exp> "and" <exp>
| <exp> "or" <exp>
| <exp> "xor" <exp>
| "not" <exp>
| "entero"
| "decimal"
| "id"
| "true"
| "false"
| "cadena"
| "char_exp"
| <llamada_funcion>


<llamada_funcion> ::= "id" "par_a" "par_c"
| "id" "par_a" <lista_expresiones> "par_c"

<lista_expresiones> ::= <lista_expresiones> "coma" <exp>
| <exp>
```