

**Manual Técnico**

Marvin Ronaldo Martinez Marroquin

201602520

Descripción

La aplicación de Arit es un interprete desarrollado en Java utilizando otros complementos para el análisis léxico y sintáctico. La aplicación esta implementado con el patrón interprete para un mejor desarrollo, para el análisis léxico y sintáctico se utilizaron dos tipos de gramáticas distintas, una desarrollada con gramática ascendente y otra descendente. Ambas devuelven un objeto árbol AST que tiene un método ejecutar para iniciar la ejecución del código. También se utilizaron otros lenguajes como graphviz para el desarrollo de un reporte del árbol generado.

La aplicación esta dividido en varios paquetes llevando una jerarquía parra que sea más organizado y más fácil de entender como esa dividido el código por clases. También cuenta con una clase global implementado singletone para ir almacenando cosas importantes como lo son el textarea, tabla de errores y métodos que sean necesarios en otras clases.

En la carpeta del proyecto se encuentra el diagrama de clases y el diagrama de paquetes para un mejor entendimiento.

Requisitos

Uno de las principales herramientas para que la aplicación se ejecute es tener instalado java en nuestra computadora.

Java 8

Windows

* Windows 10 (8u51 y superiores)
* Windows 8.x (escritorio)
* Windows 7 SP1
* Windows Vista SP2
* Windows Server 2008 R2 SP1 (64 bits)
* Windows Server 2012 y 2012 R2 (64 bits)
* RAM: 128 MB
* Espacio en disco: 124 MB para JRE; 2 MB para Java Update
* Procesador: Mínimo Pentium 2 a 266 MHz
* Exploradores: Internet Explorer 9 y superior, Firefox

Linux

* Oracle Linux 5.5+1
* Oracle Linux 6.x (32 bits), 6.x (64 bits)2
* Oracle Linux 7.x (64 bits)2 (8u20 y superiores)
* Red Hat Enterprise Linux 5.5+1, 6.x (32 bits), 6.x (64 bits)2
* Red Hat Enterprise Linux 7.x (64 bits)2 (8u20 y superiores)
* Suse Linux Enterprise Server 10 SP2+, 11.x
* Suse Linux Enterprise Server 12.x (64 bits)2 (8u31 y superiores)
* Ubuntu Linux 12.04 LTS, 13.x
* Ubuntu Linux 14.x (8u25 y superiores)
* Ubuntu Linux 15.04 (8u45 y superiores)
* Ubuntu Linux 15.10 (8u65 y superiores)
* Exploradores: Firefox

Java 7

Windows

* Windows 10 (7u85 y superiores)
* Windows 8.x (escritorio)
* Windows 7 SP1
* Windows Vista SP2
* Windows Server 2008 SP2 y 2008 R2 SP1 (64 bits)
* Windows Server 2012 (64 bits) y 2012 R2 (64 bits)
* RAM: 128 MB; 64 MB para Windows XP (32 bits)
* Espacio en disco: 124 MB
* Exploradores: Internet Explorer 7.0 y superior, Firefox 3.6 y superior

Linux

* Windows 10 (7u85 y superiores)
* Windows 8.x (escritorio)
* Windows 7 SP1
* Windows Vista SP2
* Windows Server 2008 SP2 y 2008 R2 SP1 (64 bits)
* Windows Server 2012 (64 bits) y 2012 R2 (64 bits)
* RAM: 128 MB; 64 MB para Windows XP (32 bits)
* Espacio en disco: 124 MB
* Exploradores: Internet Explorer 7.0 y superior, Firefox 3.6 y superior

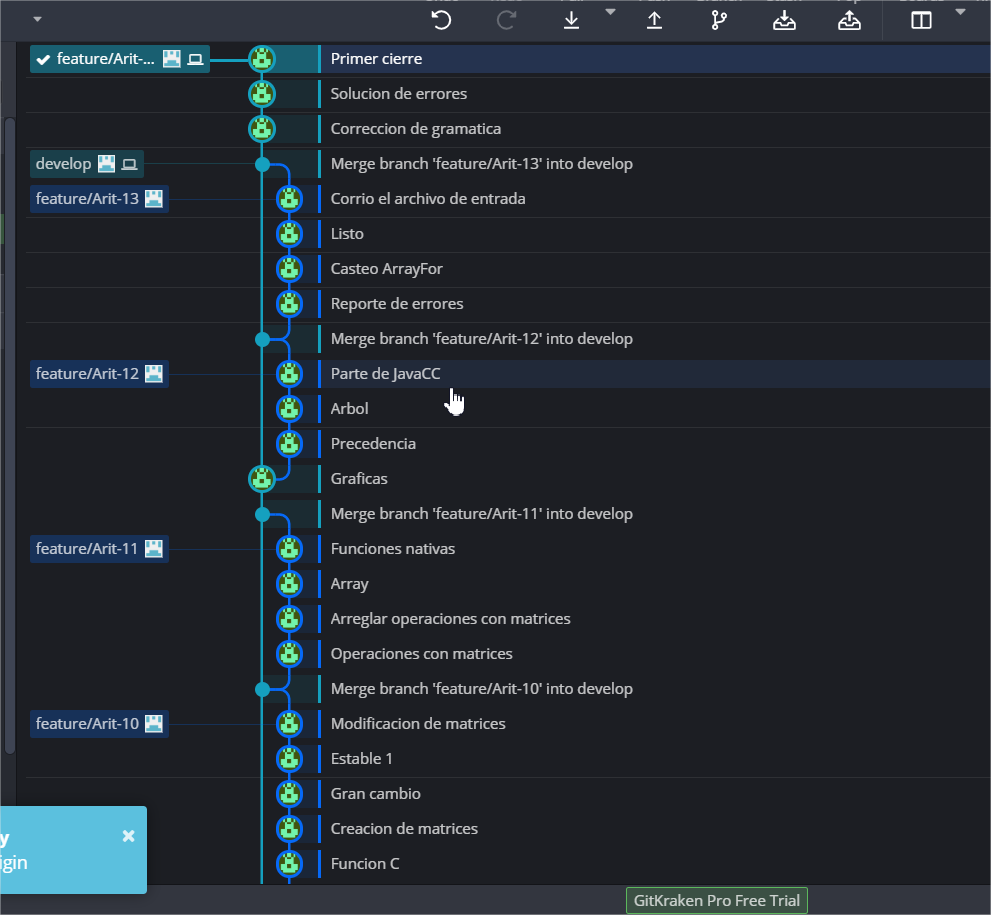
Graphviz

Esta aplicación necesita un complemento muy importante para realizar el reporte de árbol, se necesita que este instalado graphviz y tiene que estar la variable en el sistema operativo.

* 50 mb de espacio en el disco
* Tener instalado la variable en path para ejecutar
* Tener instalado un visor de imágenes

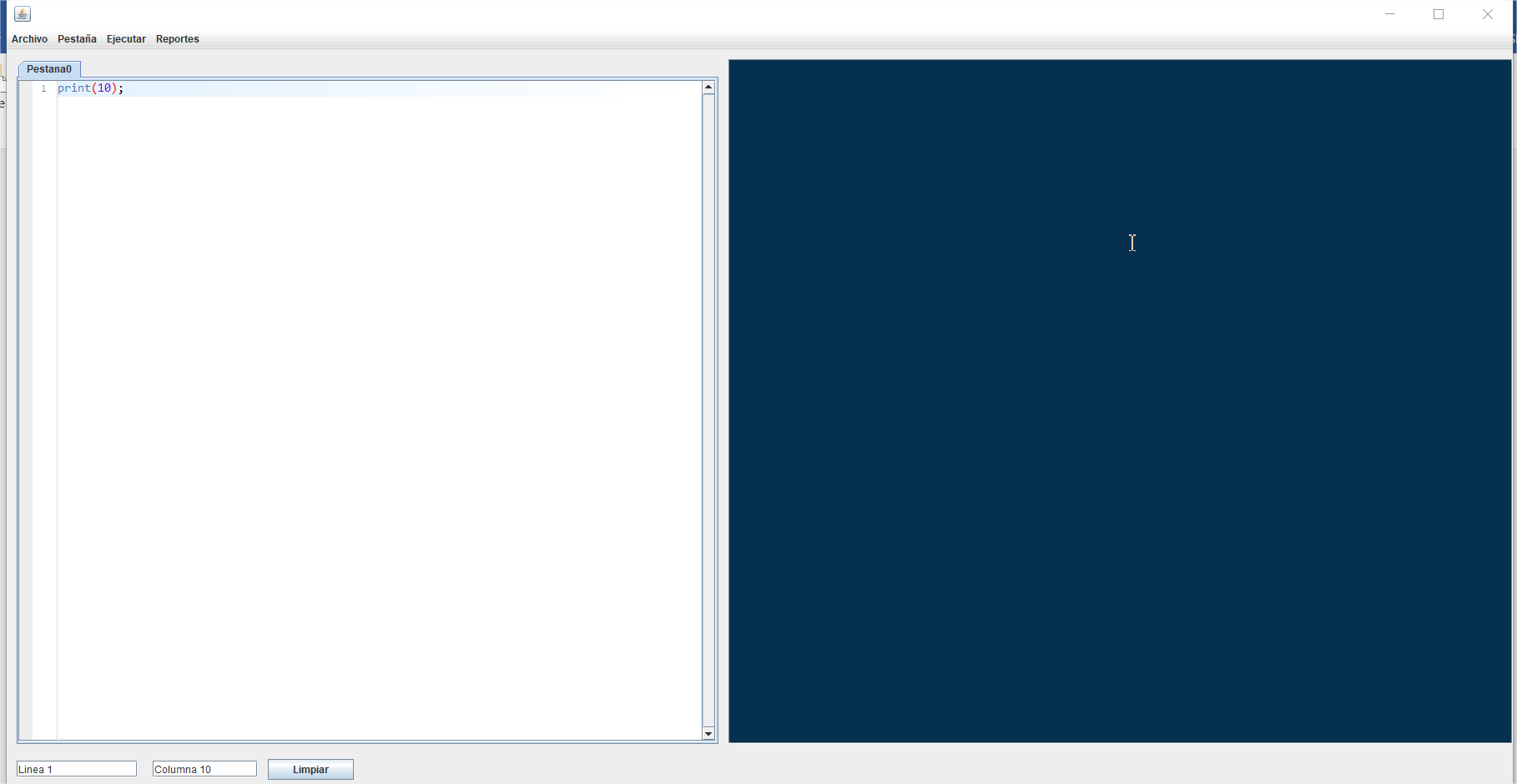
Github

El proyecto esta subido en un control de versiones y tiene implementado un workflow en el cual si se realiza un cambio grande se crea una nueva rama que se le asigna a la develop para luego hacer merge con la master. El repositorio es público para utilizar la herramienta GitKraken y ver el flujo de trabajo.



Arit

Para la aplicación solo se necesita mínimo 500mb de memoria RAM y un espacio de 20 mb en el disco para almacenamiento de la aplicación.



Herramientas

* RsyntaxtTextArea
  + Es un jar que permite utilizar sintaxis de color para los textarea con ayuda de jflex y también permite dar un identado al área de texto.
* JFreeChart
  + Jar que permite graficar datos en diferentes tipos de graficas de una manera mas fácil.
* Graphviz
  + Programa utilizado para generar el reporte de AST

Clases

Una gran parte de las clases hacen uso de una implementación de interfaz o heredad de una clase abstracta por lo que en ellas se encuentran los siguientes métodos.

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre | Parametros | Retorno | Descripcion |
| Ejecutar() | Entorno | void | Este es uno de los métodos mas importantes ya que inicia la ejecución de una instruccion |
| getLinea() | Entorno | Int | Retorna la línea del objeto |
| getColumna() | Entorno | Int | Retorno la columna del objeto |
| toDot() | Entorno e,int padre | String | Devuelve un string con el código de su construccion |
| getValor() | Entorno e | Object | Se ejecuta las instrucciones y retorna el valor de la expresion si no retorna nada es null. |

Jerarquía de las clases

* Nodo
  + Instrucción
    - Controles
      * If
      * While
      * Switch
      * Return
      * For
      * Do\_while
    - Declarar
    - DeclararFuncion
  + Expresion
    - Literal
    - Nulo
    - Llamadas
  + Simbolo
    - EstructuraLineal
    - Matrix
    - Lista
    - Array

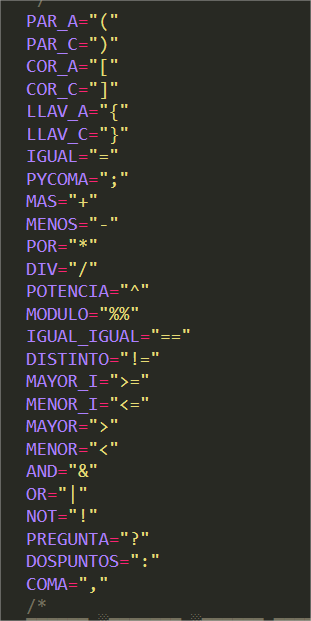
Acciones Semánticas

En los diferentes tipos de gramáticas la idea principal es construir un árbol con una lista de nodos, las clases que se forman heredan de una interfaz nodo para formarlo.

Para la parte de Flex y Cup se utiliza una gramática ascendente por lo que tiene producciones derivadas hacia la izquierda y derecha dependiendo de la necesidad.

JFlex y Cup

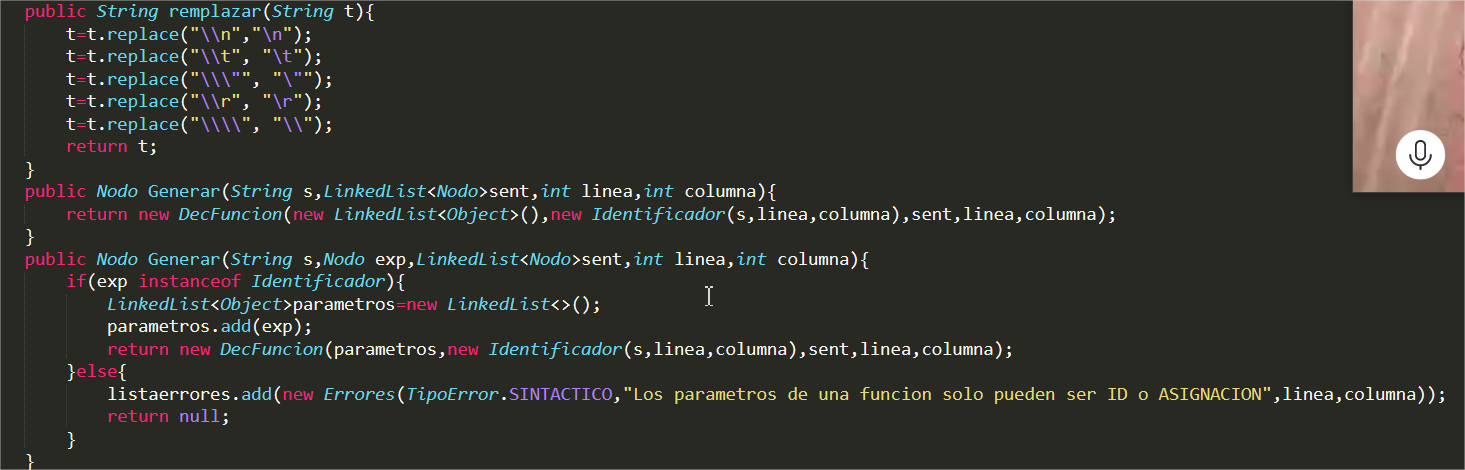
El archivo de jflex cuenta con un área de declaración de políticas

Se declara que se necesita la línea , columna y que sea ignore case. Declaramos una lista que contiene errores para almacenar errores léxicos. Luego se viene el área de declaración de tokens léxicos.

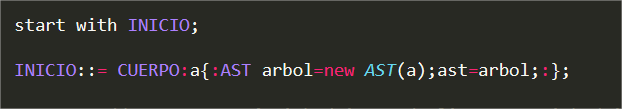
Área de palabras reservadas.



Para la parte de análisis sintactico se inicia con la declaración de métodos que son de ayuda para eliminar caracteres o crear objetos.



La producción inicial se llama INICIO y retorna un nodo AST que será la raíz de nuestro árbol

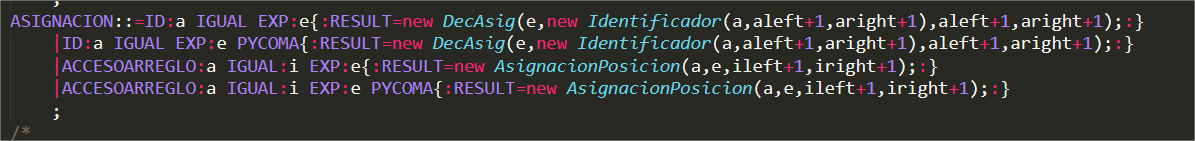


La producción Cuerpo retorna una lista de nodos



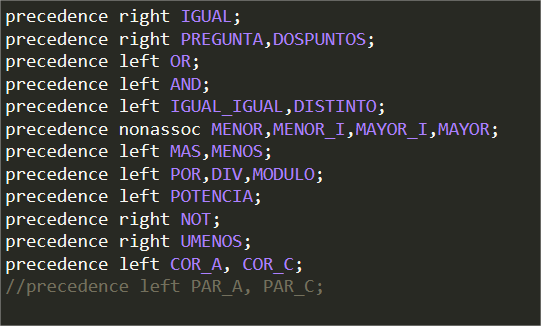
Esta producción tiene una muy similar que se llama CUERPOINTERNO con la única diferencia que adentro no puede venir declaración de funciones.

La producción de asignación retorna un nodo DecAsign la cual es la que genera las variables.

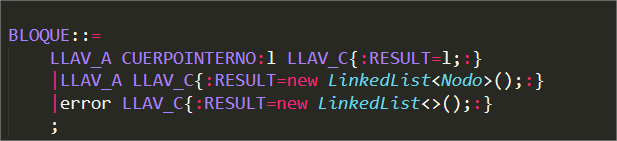


La producción de expresión retorna un objeto expresión, se soluciona la ambigüedad con reglas de prioridad.

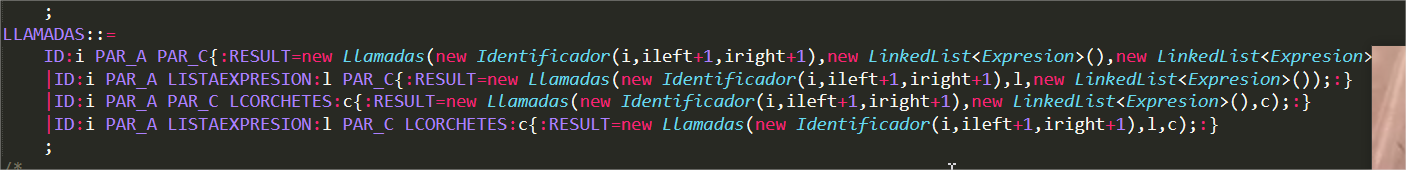




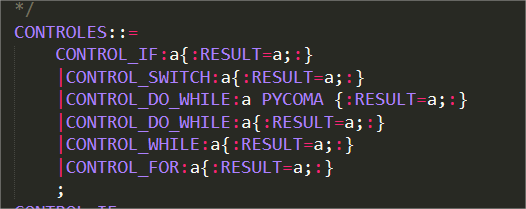
La producción de Bloque retorna una lista de sentencias



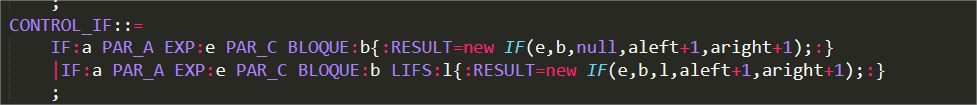
La producción de llamadas retorna un objeto de tipo llamada con el identificador de la función que se desea realizar.



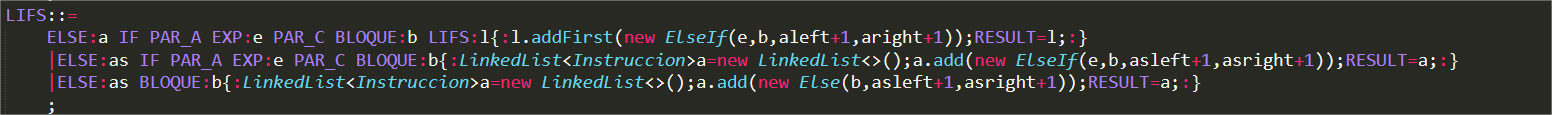
La producción de CONTROL retorna el tipo de control que se desea realizar



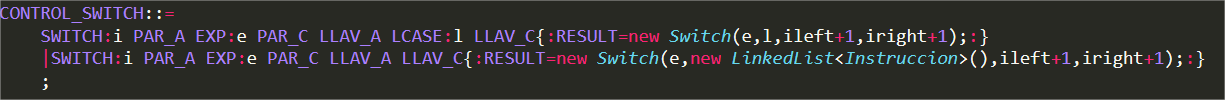
La producción de control IF retorna un objeto de tipo IF que tiene una condición de tipo expresión y una lista de sentencias



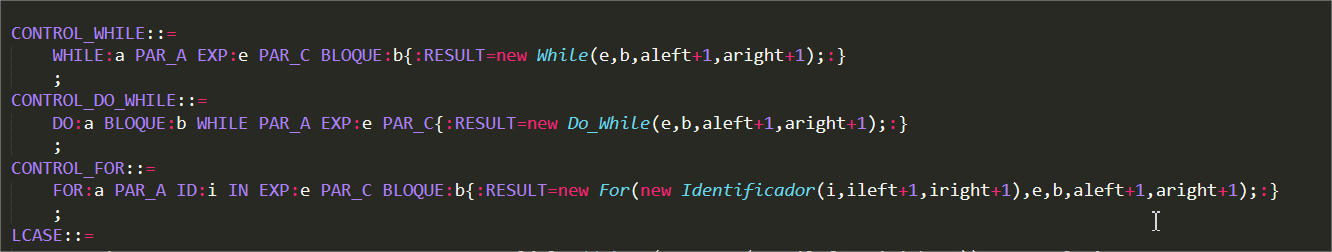
Si en esta producción vinieran mas else if o else se retorna una linkedlist de mas ifs .



Para la producción de CONTROL\_SWITCH tiene la misma lógica que el IF la cual tiene un condición , una lista de case y un default si lo tiene.



Para las producciones de ciclo tienen una lista de sentencias y un objeto de condición

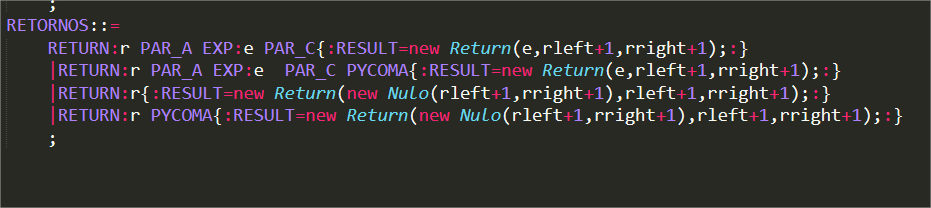


Para las producciones de Accesos a las estructuras se viene una lista de Accesos.



Para la producción de declarar objetos se utiliza una producción especial en lo cual se llama a un método para validar si se puede declarar de esa forma ya que tiraba error de Reduce/Reduce al momento de declarar variables y funciones

La producción de retornos que puede o no traer una expresión en respuesta.

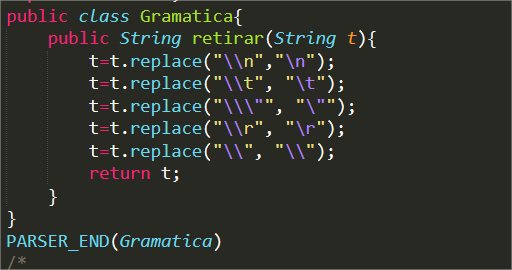


JavaCC

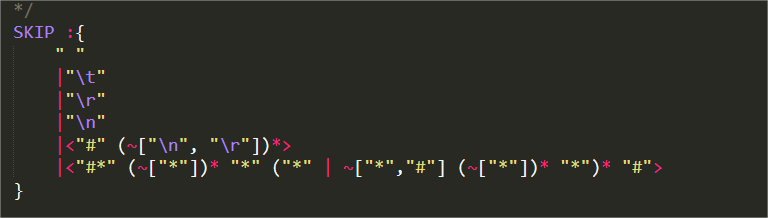
Cuenta con un área de importación de directivas.



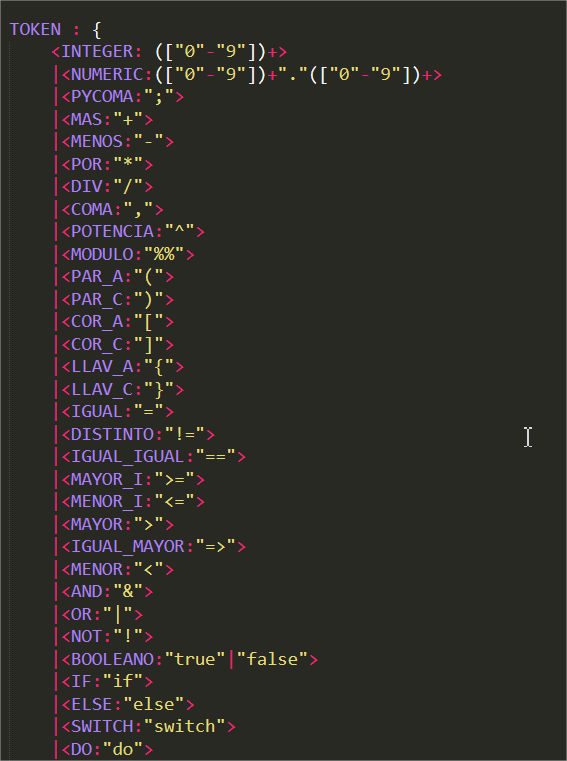
Luego se declaran los métodos que van a servir para la ejecución de la aplicación como lo son para limpiar la cadena de caracteres que no se esperan



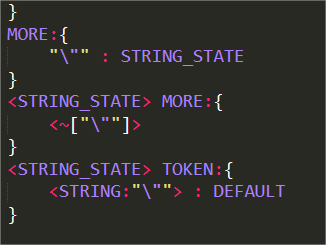
La siguiente parte del documento es la declaración de tokens léxicos.



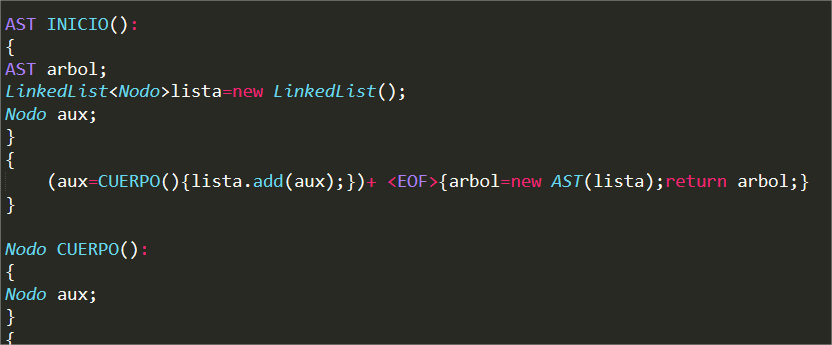
Esta parte son los tokens que no se desean ser procesados.



En la parte de tokens se declaran los símbolos y palabras reservadas del lenguaje.



Los estados para el token de cadena



Las producciones iniciales igual que en la otra gramática se busca hacer de la misma manera el árbol.