Generador de Analizador Léxico: Jax

Jax es un compilador léxico creado en lenguaje Java, que genera un escáner a partir de expresiones regulares que existen por defecto en un archivo de java. Jax procesa estas expresiones regulares y genera un fichero Java que pueda ser compilado por Java y así crear el escáner.

Los escáners generados por Jax tienen entradas de búfer de tamaño arbitrario, y es al menos más conveniente para crear las tablas de tokens, Jax utiliza solo 7 bits de caracteres ASCII, y no permite código Unario.

Uno de los aspectos fundamentales que tienen los lenguajes Lex y Yacc es que son parte importante de un compilador. Pues la unión de estos dos genera un compilador, claro cada uno de ellos aportando su propio diseño y su forma de ejecutar sus procesos. Tanto el analizador léxico como el sintáctico pueden ser escritos en cualquier lenguaje de programación. A pesar de la habilidad de tales lenguajes de propósito general como C, lex y yacc son más flexibles y mucho menos complejos de usar.

Código generado: Java.

No soporta entornos, está basado en expresiones regulares.

No soporta Unicode.

Ejemplo

```
public enum Tipo {

NUMERO("^\\d+$"),

OPERADOR_BINARIO("[*|/|+|-|=]$"),

PALABRAS_RESERVADAS("(INICIO|FIN|ESCRIBIR|LEER|MIENTRAS|FINMQ|PARA|FINPARA|SI
|SINO|CASE|BREAK|DEFAULT|"

+ "ENTERO|CADENA|FLOAT)$"),

IDENTIFICADORES("^&[A-Za-z]+$"),

SIMBOLO("[;]$");


public final String patron;


Tipo(String s) {

this.patron = s;


}

}


public class Token {


private Tipo tipo;

private String valor;


public Tipo getTipo() {

return tipo;

}


public void setTipo(Tipo tipo) {
```

```
this.tipo = tipo;

}

public String getValor() {

return valor;

}

public void setValor(String valor) {

this.valor = valor;

}



}

public class Lexer {

public static void main(String[] args) {

new probar().setVisible(true);

}

}
```