Flex/Cup files - Java

Flex standalone

Listing 1 – numbers.flex

```
1
/**
Files with numbers
*/

6 %%

%public
%class Numbers
%standalone
11
%unicode

%{
        /*no temp variable*/
16 %}

/*macros defs*/

CHIFFRE=[0-9]
21 LETTRE=[A-Za-z]
NEWLINE=\r|\n|\r\n|\t

%%
{CHIFFRE}+          { }
26 {NEWLINE}        { }
"+"                 { }

<<EOF>>             { System.out.println("end of file!");return 0;}
[^]                 { System.out.println("char non reconnu !");return 1;}
```

Listing 2 – standalone.flex

```
%public
%class Subst
4 %standalone

%unicode

%{
9   String name;
%}
```

```
%%

"name " [a-zA-Z]+   { name = yytext().substring(5);   }
[Hh] "ello"         { System.out.print(yytext()+" "+name+"!"); }
```

Listing 3 – counts.flex

```
/**
Files with lines and words
*/
%%

%public
%class Counts
%standalone

%unicode

%{/*declaration des variables*/
    int num_lines;
    int num_words;
%}

%init{/*Code embarque dans le constructeur !*/
    num_lines=0;
    num_words=0;
%init}

%eof{ /*Code execute a la fin de l'analyse (quand EOF est atteint)*/
   System.out.println(num_lines+" lignes et "+num_words+" mots !");
%eof}

/*macros defs*/
NEWLINE=\r|\n|\r\n
SPACE=\ |\t
LETTER=[A-Za-z]
%%
{NEWLINE}           { num_lines++; }

{SPACE}             {}

{LETTER}+               {num_words++; }
```

Recognizing a language with CUP

Listing 4 – anbn.flex

```
/* anbn.flex */
import java_cup.runtime.*; // import Symbol class etc
%%
%class Anbn
%unicode
%line
%column
%cup

%{
  /* To create a new java_cup.runtime.Symbol with information about the current
     token, the token will have no value in this case. */
  private Symbol symbol(int type) {
    return new Symbol(type, yyline, yycolumn);
  }
  /* Also creates a new java_cup.runtime.Symbol with information about the
     current token, but this object has a value. */
  private Symbol symbol(int type, Object value) {
    return new Symbol(type, yyline, yycolumn, value);
  }

%}

/* models */
SEPARATEUR = " "|\n|\t|\r
%%
/* rules */
{SEPARATEUR} {}
"a"          {return symbol(sym.TKA) ; }
"b"          {return symbol(sym.TKB) ; }
.            { throw new Error("Lexical error : illegal character <"+yytext()+">
```

Listing 5 – anbn.cup

```
import java_cup.runtime.*; // import Symbol class etc
parser code {:

    /*Taken from http://www.linuxgazette.net/issue41/lopes/lcalc.htm*/
    /* Change the method report_error so it will display the line
     and column of where the error occurred in the input as well as the reason
     for the error which is passed into the method in the String 'message'.
*/
    public void report_error(String message, Object info) {
        StringBuffer m = new StringBuffer("Error");
        if (info instanceof java_cup.runtime.Symbol) {
            java_cup.runtime.Symbol s = ((java_cup.runtime.Symbol) info);
```

```
            if (s.left >= 0) {
                m.append(" in line "+(s.left+1));

                if (s.right >= 0)
                    m.append(", column "+(s.right+1));
            }
        }
        m.append(" : "+message);
        System.err.println(m);
    }

    public void report_fatal_error(String message, Object info ) {
        report_error (message, info) ;
        System.exit(1);
    }

    public void unrecovered_syntax_error(Symbol cur_token)  {
        System.out.println(cur_token.toString());
        System.out.println("At Line "+cur_token.left
                           +", column"+cur_token.right);
        System.exit(1);
        }
:}



terminal TKA,TKB;
non terminal S;

start with S;

S::= TKA S TKB
|
;
```

Listing 6 – anbn.cup

```
import java.io.*;

public class Analyseur {
    static public void main ( String argv[] ){

        try {
            parser p = new parser(new Anbn(new FileReader(argv[0]))) ;
            Object result = p.parse();
            System.out.println("\n file  OK") ;
        } catch ( FileNotFoundException fe) {
            System.out.println("\n file not found") ;
            System.exit(1);
```

```
        } catch (Error e ) {
            System.out.println(e.getMessage()) ;
            System.exit(1);
        }
        catch ( Exception e) {
            System.out.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

Expressions with implicit AST

Listing 7 – expr.flex

```
/* expr.flex */
import java_cup.runtime.*; // import Symbol class etc
%%
%class Expr
%unicode
%line
%column
%cup
%{ /* a function to create tokens along with line,col. numbers */
private Symbol symbol(int type) {
        return new Symbol(type, yyline, yycolumn);
 }
 private Symbol symbol(int type, Object value) {
 return new Symbol(type, yyline, yycolumn, value) ;
}

%}
 /* models */
 integer=[0-9]+
 space=\r|\n|\r\n|\t|\
%%
 /* rules */
{integer} { return symbol(sym.TKINT,new Integer(yytext()));}
"+" {return symbol(sym.TKPLUS) ; }
"*" {return symbol(sym.TKTIMES) ; }
";" {return symbol(sym.TKSEMICOL) ; }
{space}           { }
```

Listing 8 – expr.cup

```
/*expr.cup*/

import java_cup.runtime.*;
parser code {:
    public void report_fatal_error( String message, Object info)
        throws Exception {report_error (message, info );
        throw new Exception("Syntax Error");
    }
:};

terminal Integer TKINT;
terminal TKPLUS,TKTIMES,TKSEMICOL;
non terminal S;
non terminal Integer E;

precedence left TKPLUS;
```

```
precedence left TKTIMES;

S ::= E:e TKSEMICOL {:System.out.println(e.intValue());:}
;

E ::= TKINT:n
      {: RESULT = new Integer(n.intValue()); :}
|     E:e1 TKPLUS E:e2
      {: RESULT = new Integer(e1.intValue() + e2.intValue()) ; :}
|     E:e1 TKTIMES E:e2
      {: RESULT = new Integer(e1.intValue() * e2.intValue()) ; :}
;
```

Expressions with explicit AST

Listing 9 – expr.flex(V2)

```
/* expr.flex */
import java_cup.runtime.*; // import Symbol class etc
%%
%class Expr
%unicode
%line
%column
%cup
%{ /* a function to create tokens along with line,col. numbers */
private Symbol symbol(int type) {
        return new Symbol(type, yyline, yycolumn);
}
 private Symbol symbol(int type, Object value) {
 return new Symbol(type, yyline, yycolumn, value) ;
}

%}
 /* models */
 integer=[0-9]+
 space=\r|\n|\r\n|\t|\
%%
 /* rules */
{integer} { return symbol(sym.TKINT,new Integer(yytext()));}
"+" {return symbol(sym.TKPLUS) ; }
"*" {return symbol(sym.TKTIMES) ; }
";" {return symbol(sym.TKSEMICOL) ; }
{space}           { }
```

Listing 10 – expr.cup (V2)

```
/*expr.cup*/

import java_cup.runtime.*;
parser code {:
    public void report_fatal_error( String message, Object info)
        throws Exception {report_error (message, info );
        throw new Exception("Syntax Error");
    }
:};

terminal Integer TKINT;
terminal TKPLUS,TKTIMES,TKSEMICOL;
non terminal S;
non terminal ASTExpr E;

/* est prioritaire sur + ! */
```

```
precedence left TKPLUS;
precedence left TKTIMES;

S ::= E:e TKSEMICOL {:System.out.println(e.eval());:}
;

E ::= TKINT:n
      {: RESULT = new ASTExpr(n.intValue()); :}
|     E:e1 TKPLUS E:e2
      {: RESULT = new ASTExpr(e1,ASTExpr.ADD,e2); :}
|     E:e1 TKTIMES E:e2
      {: RESULT = new ASTExpr(e1,ASTExpr.MUL,e2); :}
;
```

Listing 11 – AST class

```
public class ASTExpr {
    final static int INT=0, ADD=1, MUL=2 ;
    int tag ;
    int asInt ; // value used if tag = INT
    ASTExpr e1, e2 ; // used if ADD or MUL

    //Constructors

    ASTExpr(int i) { this.tag = INT ; this.asInt = i ; }

    ASTExpr(ASTExpr e1, int op, ASTExpr e2) {
        this.tag = op; this.e1 = e1; this.e2 = e2; }

    //evaluation of an expression
    int eval() {
        switch (this.tag) {
        case ASTExpr.INT: return this.asInt;
        case ASTExpr.ADD: return this.e1.eval()+this.e2.eval();
        case ASTExpr.MUL: return  this.e1.eval()*this.e2.eval();
        default : throw new Error("incorrect tag");
        }

    }

}
```

Listing 12 – Main class

```
import java.io.*;

public class Analyseur {
    static public void main ( String argv[] ){

        try {
```

```
            parser p = new parser(new Expr(new FileReader(argv[0]))) ;
            Object result = p.parse();
            System.out.println("\n file  OK") ;
        } catch ( Exception e ) {
            System.out.println("\n Syntax Error" ) ;
        }
    }
}
```