Java Tools for Scanning and Parsing

CS 4447 / CS 9545 – Stephen M. Watt
University of Western Ontario

Java Tools

- We have already seen the C-based tools Lex & Yacc
- For those who want to do development in Java, ∃
  - JFlex (Java version of Flex, Fast Lex)
    http://jflex.de
  - CUP (Constructor of Useful Parsers)
    http://www2.cs.tum.edu/projects/cup
- Both are free.

A Calculator Example, Using JFlex and CUP

- *The source files are*:
  - MyMain.java -- main driver program
  - MyParser.cup -- modified example from the manual
  - MyScanner.lex -- a compatible JFlex scanner
- *The commands to build it* are given in the Makefile, except the commands to compile .java into .class files (handled automatically by implicit make rules).
- *To build or run this program on GAUL* you need to set the CLASSPATH to contain /usr/local/JClass,
  e.g. setenv CLASSPATH /usr/local/JClass:.
- *To run the calculator program* do: java MyMain
  Then type input commands, separated by semicolons, followed by ^D to exit.

MyMain.java

```
class MyMain {
    public static void main(String[] args) {
        MyParser parser =
            new MyParser(new MyScanner(System.in));
        try {
            parser.parse();
        }
        catch (Exception e) {
            System.out.println("Caught an exception.");
        }
    }
}
```

MyParser.cup – Part 1

```
// CUP specification for a simple expression evaluator (w/ actions)
 import java_cup.runtime.*;

 /* Terminals (tokens returned by the scanner). */
 terminal ERROR,
             SEMI, PLUS, MINUS, TIMES, DIVIDE, MOD, UMINUS,
             LPAREN, RPAREN;
 terminal Integer   NUMBER;

 /* Non-terminals */
 non terminal              expr_list, expr_part;
 non terminal Integer    expr;

 /* Precedences */
 precedence left PLUS, MINUS;
 precedence left TIMES, DIVIDE, MOD;
 precedence left UMINUS;
```

MyParser.cup – Part 2

```
/* The grammar */
expr_list ::= expr_list expr_part | expr_part ;
expr_part ::= expr:e {: System.out.println("gives " + e); :} SEMI
|             error SEMI
;
expr      ::= expr:e1 PLUS expr:e2
            {: RESULT = new Integer(e1.intValue() + e2.intValue()); :
|           expr:e1 MINUS expr:e2
            {: RESULT = new Integer(e1.intValue() - e2.intValue()); :}
|           expr:e1 TIMES expr:e2
            {: RESULT = new Integer(e1.intValue() * e2.intValue()); :}
|           expr:e1 DIVIDE expr:e2
            {: RESULT = new Integer(e1.intValue() / e2.intValue()); :}
|           expr:e1 MOD expr:e2
            {: RESULT = new Integer(e1.intValue() % e2.intValue()); :}
|           NUMBER:n
            {: RESULT = n; :}
|           MINUS expr:e
            {: RESULT = new Integer(0 - e.intValue()); :}   %prec UMINUS
|           LPAREN expr:e RPAREN
            {: RESULT = e; :}
;
```

JFlex Scanner – Part 1

```
/* JFlex Scanner for CUP example. */
 import java_cup.runtime.*;
 %%
 %class  MyScanner
 %cupsym MySymbol
 %cup
 %unicode
 %line
 %column

 %{
    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);
    }
 %}
```

JFlex Scanner – Part 2

```
WhiteSpace = [ \t\f\r\n]
Number     = [0-9]+

%%
";"             { return symbol(MySymbol.SEMI); }
"+"             { return symbol(MySymbol.PLUS); }
"-"             { return symbol(MySymbol.MINUS); }
"*"             { return symbol(MySymbol.TIMES); }
"/"             { return symbol(MySymbol.DIVIDE); }
"%"             { return symbol(MySymbol.MOD); }
"("             { return symbol(MySymbol.LPAREN); }
")"             { return symbol(MySymbol.RPAREN); }
{Number}        { return symbol(MySymbol.NUMBER, new Integer(yytext()) ); }
{WhiteSpace} { /* ignore */ }
.               { return symbol(MySymbol.ERROR, yytext()); }
```

Makefile

```
all: MySymbol.class MyScanner.class MyParser.class MyMain.class

MyParser.java MySymbol.java: MyParser.cup
	java java_cup.Main -parser MyParser -symbols MySymbol < MyParser.cup

MyScanner.java: MyScanner.lex
	jflex MyScanner.lex

clean:
	rm -f *.class MyScanner.java MySymbol.java MyParser.java *~
```