COMPILER CONSTRUCTION, 28.07.17

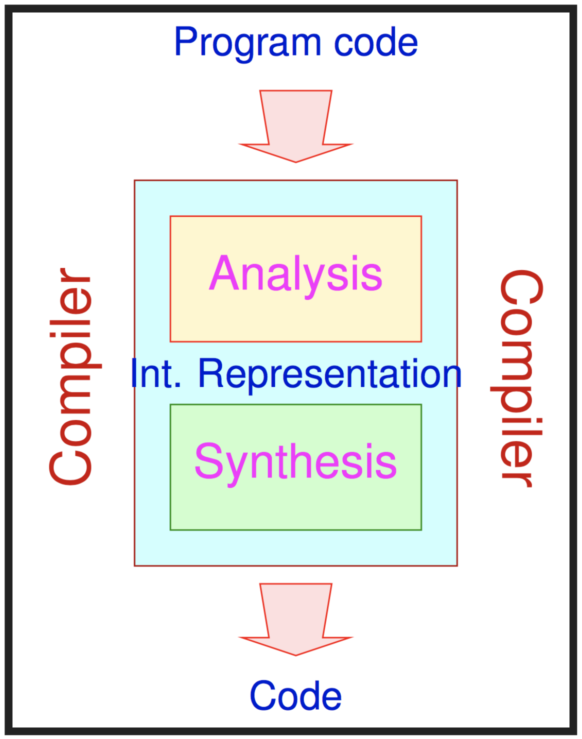
GENERATING A JAVA-BASED PARSER WITH CUP/JFLEX

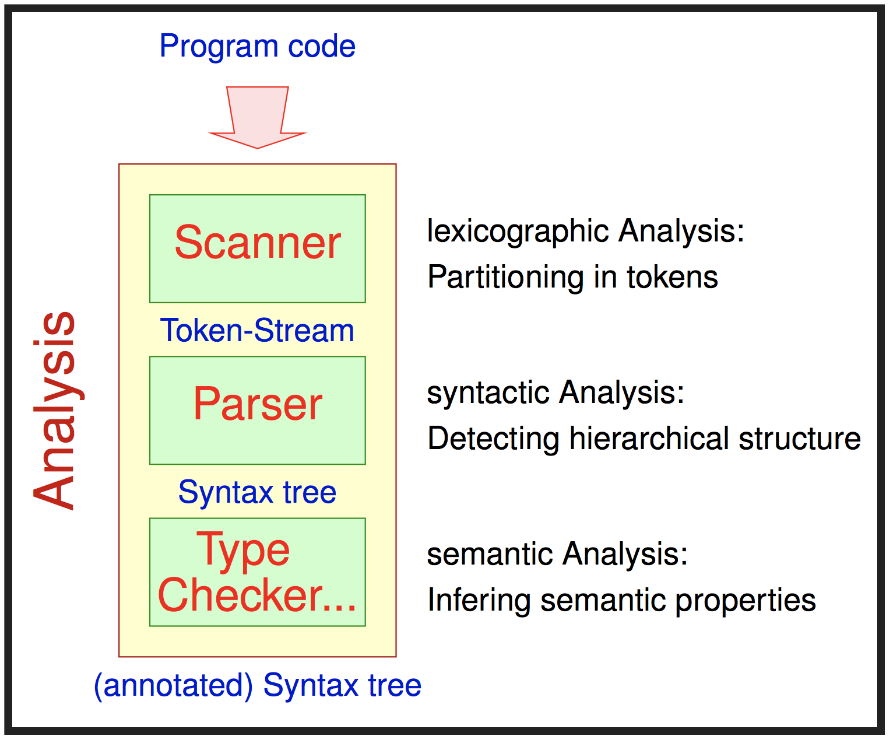
BY MARKUS MÜLLER

OUTINE

1. What are CUP & JFlex?
2. How to install
3. Showcase: Calculator
4. Further reading

BIG PICTURE





Scanner Generator: JFlex, **Parser Generator:** CUP

Images: Lecture Slides

WHY DO WE WANT GENERATORS?

- Allow us to limit ourselves on specifying patterns, rules and expected actions
- Take care of the actual implementation & code generation
- Optimized to generate fast components

JFLEX: SCANNER GENERATOR (JFLEX.DE)

- Lexical specification (set of reguar expressions & corresponding actions) ⇒ Lexer program
- `scanner.jflex` ⇒ `scanner.java`

JFLEX (2)

- *1998: Gerwin Klein (TUM), Steve Rowe, Régis Décamps
- Rewrite of the tool *JLex* (Berk 1996, Princeton)
- Current stable version: 1.6.1 (Mar 16, 2015)

`SCANNER.JFLEX` PARTS

```
UserCode
%%
Options and declarations
%%
Lexical rules
```

SCANNER.JFLEX CODE EXAMPLE

```
import java_cup.runtime.SymbolFactory;
%%
%cup
%class Scanner
%{
    public Scanner(java.io.InputStream r, SymbolFactory sf){
        this(r);
        this.sf=sf;
    }
private SymbolFactory sf;
%}
%%
";" { return sf.newSymbol("Semicolon",sym.SEMI); }
"+" { return sf.newSymbol("Plus",sym.PLUS); }
"-" { return sf.newSymbol("Minus",sym.MINUS); }
```

```
// SymbolFactory methods used
Symbol newSymbol(String name, int id)
Symbol newSymbol(String name, int id, Object value)
```

SOME LEXICALLY CORRECT INPUTS

- `8+33;`
- `11  ;`
- ...

but also:

- `11+`
- `;8`

Syntactic Correctness is Parser's task!

PROBLEM

```
x=";";
```

Within the quotes context, a semicolon should be recognized as **string part**, **not** as an own symbol.

SOLUTION: STATES

```
StringBuffer text = new StringBuffer();
...
 <YYINITIAL> {
    "\""  { text.setLength(0); yybegin(STRING); }
    ";" { return sf.newSymbol("Semikolon",sym.SEMI); }
    ...
}
 <STRING> {
     "\"" { yybegin(YYINITIAL); return symbol(sym.STRINGLITERA
text.toString()); }
     [^\n\r\"\\]+ { text.append(yytext()); }
 }
 . { System.err.println("Illegal character: "+yytext());}
```

CUP: LALR PARSER GENERATOR

- CUP: **C**onstruction of **U**seful **P**arsers
- Syntactic specification (grammar symbols, productions, opt. action code) ⇒ Parser program
- `parser.cup` ⇒ `parser.java` & `sym.java`
- Developed by C. Scott Ananian, Frank Flannery, Dan Wang, Andrew W. Appel and Michael Petter
- **Michael Petter** is the current maintainer
- Current stable version: 0.11b (Jun 15, 2016)

CUP CODE EXAMPLE

```
import java_cup.runtime.*;
...
terminal SEMI, PLUS, MINUS;
terminal Integer NUMBER;
non terminal Integer expr;
...
expr_list ::= expr_list expr:e SEMI {: System.out.println(e);:
              | expr:e SEMI         {: System.out.println(e);:
;
expr::= NUMBER:n                 {: RESULT=n; :}
        | expr:e1 PLUS  expr:e2  {: RESULT = e1+e2; :}
        | expr:e1 MINUS expr:e2  {: RESULT = e1-e2; :}
        | expr:e1 TIMES expr:e2  {: RESULT = e1*e2; :}
        | MINUS expr:e           {: RESULT = -e;    :}
 ;
```

PROBLEM

`4*2+3`

Precendences unclear (PLUS vs. TIMES) => Shift/Reduce-Conflict

SOLUTION: DEFINE PRECENDENCES

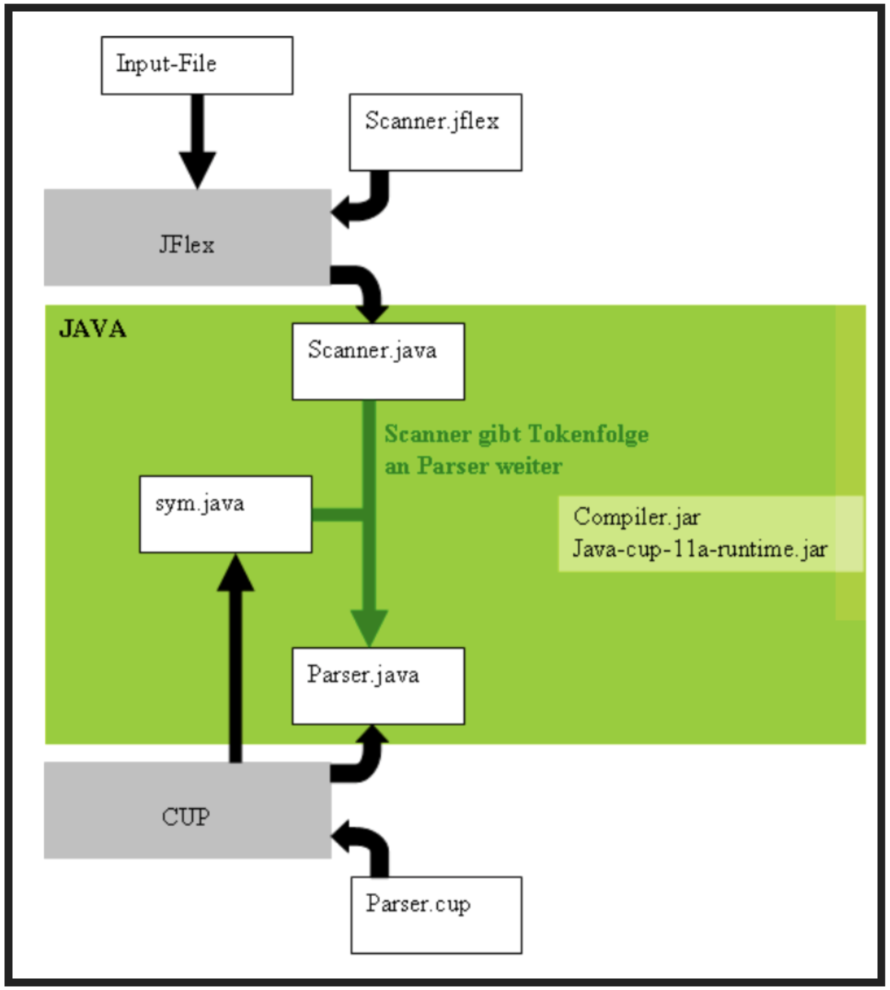
After terminal/non-terminal declarations:

```
precedence left PLUS;
precedence left MINUS;
precedence left TIMES;
```

Last line gets highest precedence

JFLEX & CUP ARE OFTEN USED TOGETHER

(Cuno 2009)



INSTALLATION

INSTALLATION: JFLEX (UNIX/MACOS)

```
wget http://jflex.de/release/jflex-1.6.1.tar.gz
sudo tar -C /usr/local -xvzf jflex-1.6.1.tar.gz
ln -s /usr/local/jflex-1.6.1/bin/jflex /usr/local/bin/jflex
```

Then:

```
jflex <options> <inputfiles>
```

(if we don't provide options or inputfiles, JFlex will ask via GUI window)

```
$ jflex scanner.jflex
Reading "scanner.jflex"
Constructing NFA : 16 states in NFA
Converting NFA to DFA :
.......
9 states before minimization, 7 states in minimized DFA
Writing code to "Scanner.java"
```

INSTALLATION: CUP (UNIX/MACOS)

```
wget http://www2.cs.tum.edu/projekte/cup/releases/java-cup-bin
tar -xvzf java-cup-bin-11b-20160615.tar.gz
```

Then:

```
java -jar java-cup-11b.jar <inputfile>
```

```
java -jar java-cup-11b.jar parser.cup
------- CUP v0.11b 20160615 (GIT 4ac7450) Parser Generation Su
  0 errors and 0 warnings
  10 terminals, 2 non-terminals, and 9 productions declared,
  producing 19 unique parse states.
  0 terminals declared but not used.
  0 non-terminals declared but not used.
  0 productions never reduced.
  0 conflicts detected (0 expected).
  Code written to "parser.java", and "sym.java".
---------------------------------------------------- (CUP v0.1
```

(`parser.cup` code from CUP website)

SHOWCASE:

CALCULATOR

Live Coding

- from CUP Installation page (scroll to bottom)
- JFlex & CUP don't need to be already installed

```
wget http://www2.cs.tum.edu/projekte/cup/releases/minimal.tar.
tar -xvzf minimal.tar.gz
cd ./template
```

```
# Install Apache Ant (http://ant.apache.org/manual/install.htm
# e.g. with
brew install ant
```

```
# download & extract JFlex
cp <jflex>/lib/jflex-1.6.1.jar <template>/tools/JFlex.jar
```

```
// in build.xml
// change
classname="JFlex.anttask.JFlexTask"
// to
classname="jflex.anttask.JFlexTask"
```

```
// in ./jflex/Scanner.jflex
// change
java.io.InputStream r
// to
java.io.Reader r
```

```
// in ./cup/Parser.jflex
// change
if (args.length==0) new Parser(new Scanner(System.in,sf),sf).p
else new Parser(new Scanner(new java.io.FileInputStream(args[0
// to
if (args.length==0) new Parser(new Scanner(new java.io.Buffere
else new Parser(new Scanner(new java.io.InputStreamReader(new
```

input.test

```
4+4;
5++5;
5+5*2;
```

Run it:

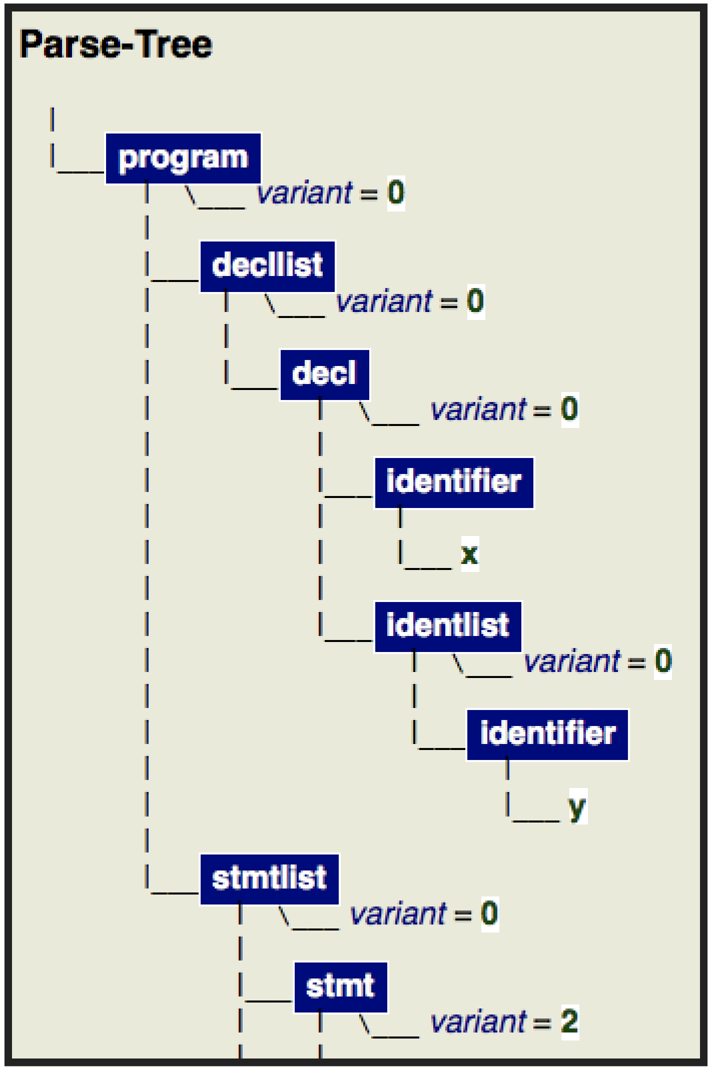
```
$ ant run
<....>
run:
     [java]  = 8;
     [java]  = 15;
     [java] Syntax error in input from :2/1(5) to :2/5(9)

BUILD SUCCESSFUL
Total time: 1 second
```

IT WORKS!

FURTHER READING

FROM THE CUP DOCUMENTATION

- Graphical AST generation
- Code generation
- Error recovery



```
Parse-Tree

 |
 |___ program
        |   \___ variant = 0
        |
        |___ decllist
        |      |   \___ variant = 0
        |      |
        |      |___ decl
        |             |   \___ variant = 0
        |             |
        |             |___ identifier
        |             |       |
        |             |       |___ x
        |             |
        |             |___ identlist
        |                     |   \___ variant = 0
        |                     |
        |                     |___ identifier
        |                              |
        |                              |___ y
        |
        |___ stmtlist
               |   \___ variant = 0
               |
               |___ stmt
               |      |   \___ variant = 2
               |      |
```

CUP2

- Developed by Michael Petter
- Supports not only LALR(1), but also LR(0) and LR(1)
- Uses modern Java features

YACC & LEX

- "Classical" combination (Yacc: Parser Generator, Lex: Scanner Generator)
- Implemented in C
- Website

MISCELLANEOUS

Martin Fowler on CUP/JFlex

THANK YOU!

SOURCES

- CUP Project Website
- JFlex Project Website
- Andrea Cuno. Compilertechnik - Parser, Scanner & Co. (Proseminar 2009)