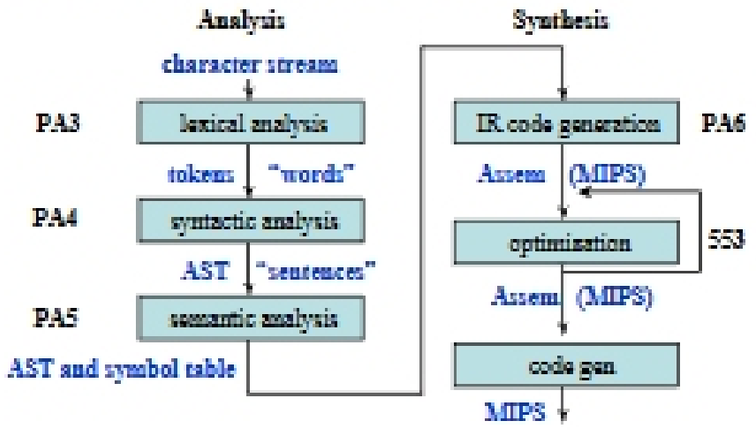
Plan for Today

Structure of the MiniJava Compiler

- Lexer
- Interface between lexer and parser: Symbol and TokenValue
- Parser
- Interface between parser and semantic analysis: ast.node.*
- Interface between semantic analysis and code generation: symbol table

Structure of the MiniJava Compiler



Specifying Tokens with JFlex

JFlex example input file:

```
package mjparser;
import java_cup.runtime.Symbol;


%%
%line
%char
%cup
%public

%eofval{
  return new Symbol(sym.EOF, new
TokenValue("EOF", yyline, yychar));
%eofval}
```

```
LETTER=[a-Za-z]
DIGIT=[0-9]
UNDERSCORE="_"
LETT_DIG_UND=(LETTER)|(DIGIT)|(UNDERSCORE)
ID=(LETTER)((LETT_DIG_UND))*

%%
"&&" { return new Symbol(sym.AND, new
TokenValue(yytext(), yyline, yychar)); }


"boolean" {return new
Symbol(sym.BOOLEAN,...


{ID} { return new Symbol(sym.ID, new ...
```

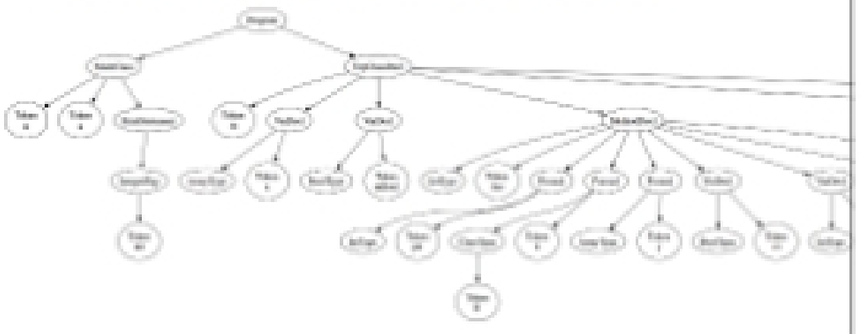

Specifying Grammar with JavaCUP

JavaCUP example input file:

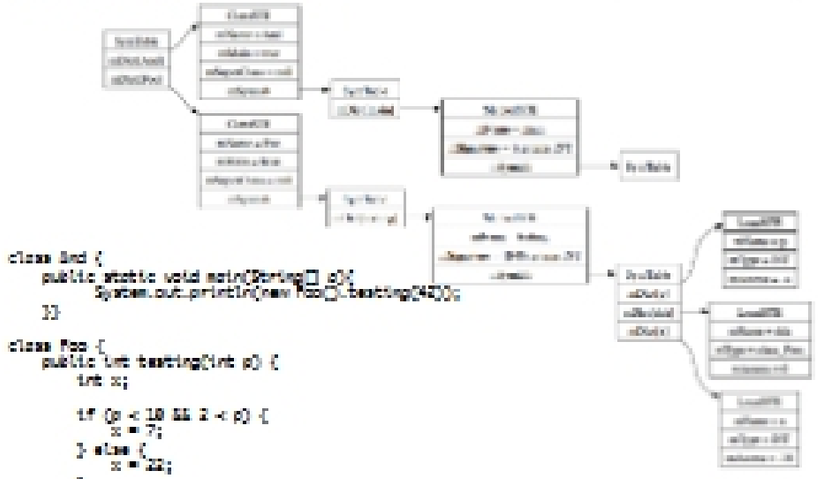
```
package mjparser;
import java_cup.runtime.*;
import ast.node.*;
...
terminal AND, ASSIGN, INT;
terminal mjparser.TokenValue
    NUMBER;
...
non terminal Program  program;

non terminal List<IClassDecl>
    class_decl_list;

non terminal MainClass
    main_class;
```

```
start with program;

program ::=
    main_class:m class_decl_list:l
    {: RESULT = new Program(m,l); :}
    ;

exp ::=
  |   NUMBER:n
    {: Token token = new Token(n.text,
n.line, n.pos);
       RESULT = new IntegerExp( token );
    :}
 ...
```

Abstract Syntax Tree for Memory Layout Example



Example Symbol Table



```
class Bed {
    public static void main(String[] p){
        System.out.println(new Foo().testing(42));
    }}

class Foo {
    public int testing(int p) {
        int x;

        if (p < 10 && 2 < p) {
            x = 7;
        } else {
            x = 22;
        }
        return x;}}
```


Assem(MIPS)

```
        .text
main:
main_framesize=24
main_paramaregsaves=0
        sw $fp, 0($sp)
        subu $sp, $sp, 4
        sw $ra, 0($sp)
        subu $sp, $sp, 4
        addu $fp, $sp, main_paramaregsaves
        subu $sp, $fp, main_framesize
        # BopCALL
        # BopCONST
        li $t2, 0
        sw $t2, -0($fp)
        lw $t2, -0($fp)
        # push parameter onto stack
        sw $t2, 0($sp)
        subu $sp, $sp, 4
        jal _malloc
        #GenMOVE(BopTEMP(t1), e)
        move $t0, $v0
        sw $t0, -12($fp)
        # BopCALL
        lw $t2, -12($fp)
        # push parameter onto stack
        sw $t2, 0($sp)
        subu $sp, $sp, 4
        # BopCONST
        li $t2, 42
        sw $t2, -16($fp)
        lw $t2, -16($fp)
        # push parameter onto stack
        sw $t2, 0($sp)
        subu $sp, $sp, 4
        jal Foo_testing
        #GenMOVE(BopTEMP(t1), e)
        move $t0, $v0
        sw $t0, -20($fp)
        # BopCALL
        lw $t2, -20($fp)
        # push parameter onto stack
        sw $t2, 0($sp)
        subu $sp, $sp, 4
        jal _println
        # epilogue
done1:
        lw $ra, 0($fp)
        move $t0, $fp
        lw $fp, -4($fp)
        move $sp, $t0
        jr $ra
```