Implementación de la Calculadora del Capítulo 4 en ANTLR

Santiago Céspedes

26 de febrero de 2026

1. Objetivo

Implementar una calculadora aritmética utilizando ANTLR 4.13.2 con Java como lenguaje objetivo, aplicando el patrón *Visitor* explicado en el Capítulo 4. La calculadora permite evaluar expresiones con suma, resta, multiplicación, división, paréntesis y asignación de variables.

2. Instalación y Configuración del Entorno

El proyecto se desarrolló en WSL (Ubuntu en Windows).

Instalación de Java

```
sudo apt update
sudo apt install openjdk-17-jdk -y
```

Verificación

```
java -version
```

Instalación de ANTLR

Se configuró el archivo `antlr-4.13.2-complete.jar` en el sistema y se creó el alias:

```
alias antlr4='java␣-jar␣/usr/local/lib/antlr-4.13.2-complete.jar'
```

3. Diseño de la Gramática

Se creó el archivo `Calc.g4` con la siguiente estructura:

```
grammar Calc;

prog:   stat+ ;

stat:   expr NEWLINE                # printExpr
    |   ID '=' expr NEWLINE         # assign
    |   NEWLINE                     # blank
    ;

expr:   expr op=('*'|'/') expr      # MulDiv
    |   expr op=('+'|'-') expr      # AddSub
    |   INT                         # int
    |   ID                          # id
    |   '(' expr ')'                # parens
    ;

MUL :   '*' ;
DIV :   '/' ;
ADD :   '+' ;
SUB :   '-' ;
ID  :   [a-zA-Z]+ ;
INT :   [0-9]+ ;
NEWLINE:'\r'? '\n' ;
WS  :   [ \t]+ -> skip ;
```

La gramática define:

- Reglas sintácticas para expresiones aritméticas.
- Precedencia correcta de operadores.
- Tokens léxicos para números, identificadores y operadores.

4. Generación del Parser

Se generó el parser y visitor con el siguiente comando:

```
antlr4 -visitor Calc.g4
```

Archivos generados:

- CalcLexer.java
- CalcParser.java
- CalcBaseVisitor.java
- CalcVisitor.java

5. Implementación del Visitor

Se creó la clase `EvalVisitor.java` que extiende `CalcBaseVisitor<Integer>`.

El visitor evalúa recursivamente las expresiones y utiliza un `HashMap` para almacenar variables.

Ejemplo del método para suma y resta:

```
@Override
public Integer visitAddSub(CalcParser.AddSubContext ctx) {
    int left = visit(ctx.expr(0));
    int right = visit(ctx.expr(1));
    if (ctx.op.getType() == CalcParser.ADD)
        return left + right;
    return left - right;
}
```

6. Clase Principal

Se implementó la clase `Main.java` para ejecutar el parser:

```
CharStream input = CharStreams.fromStream(System.in);
CalcLexer lexer = new CalcLexer(input);
CommonTokenStream tokens = new CommonTokenStream(lexer);
CalcParser parser = new CalcParser(tokens);

ParseTree tree = parser.prog();
EvalVisitor eval = new EvalVisitor();
eval.visit(tree);
```

7. Compilación

```
javac -cp ".:/usr/local/lib/antlr-4.13.2-complete.jar" *.java
```

8. Ejecución

```
java -cp ".:/usr/local/lib/antlr-4.13.2-complete.jar" Main
```

Ejemplo de ejecución:

```
3+4*2
```

Salida:

```
11
```

9. Conclusiones

Se logró implementar correctamente la calculadora utilizando ANTLR y el patrón Visitor en Java. El sistema respeta la precedencia de operadores y permite asignación de variables. La integración entre el lexer, parser y visitor demuestra el funcionamiento completo de un pequeño intérprete aritmético.