

# Documentación de calculadora con ANTLR

Jhonatan Valdes

Febrero - 26

## 1. Introducción

Este documento describe la construcción de una calculadora aritmética utilizando ANTLR y el patrón Visitor. La aplicación permite evaluar expresiones, manejar variables y respetar la precedencia de operadores.

## 2. Instalación del Entorno

Para ejecutar la calculadora fue necesario instalar Java y ANTLR.

### 2.1. Instalación de Java

```
apt update
apt install default-jdk
java -version
```

### 2.2. Instalación de ANTLR

Se descargó el runtime completo y se configuró el acceso desde la terminal.

```
cd /usr/local/lib
wget https://www.antlr.org/download/antlr-4.13.1-
complete.jar
```

Configuración de alias:

```
alias antlr4='java -jar /usr/local/lib/antlr-4.13.1-
complete.jar'
alias grun='java -jar /usr/local/lib/antlr-4.13.1-
complete.jar org.antlr.v4.gui.TestRig'
```

### 3. Generación del Analizador

A partir de la gramática `Calc.g4` se generan el lexer y el parser:

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

Archivos generados automáticamente:

- `CalcLexer.java`, `CalcLexer.class`, `CalcLexer.tokens`, `CalcLexer.interp`
- `CalcParser.java`, `CalcParser.class`, `Calc.tokens`, `Calc.interp`
- `CalcVisitor.java`, `CalcBaseVisitor.java`
- `CalcVisitor.class`, `CalcBaseVisitor.class`

### 4. Archivos Implementados Manualmente

- `Calc.java` (clase principal del programa)
- `EvalVisitor.java` (evaluación de expresiones)

### 5. Compilación

Todas las clases se compilan usando el runtime de ANTLR:

```
javac -cp " ./usr/local/lib/antlr-4.13.1-complete.jar "  
*.java
```

### 6. Ejecución

El programa procesa expresiones desde un archivo o entrada estándar:

```
java -cp " ./usr/local/lib/antlr-4.13.1-complete.jar "  
Calc < input.txt
```

### 7. Ejemplo de Uso

Entrada:

```
a = 5  
b = 3  
a + b * 2  
(1 + 2) * 4
```

Salida:

11 12
----------

## 8. Arquitectura del Sistema

- **Calc.g4**: define la gramática del lenguaje.
- **CalcLexer**: convierte la entrada en tokens.
- **CalcParser**: construye el árbol sintáctico.
- **EvalVisitor**: evalúa expresiones mediante Visitor.
- **Calc**: punto de entrada del programa.

## 9. Conclusión

Se implementó una calculadora funcional utilizando ANTLR y el patrón Visitor, demostrando la generación automática de analizadores y la evaluación estructurada de expresiones.