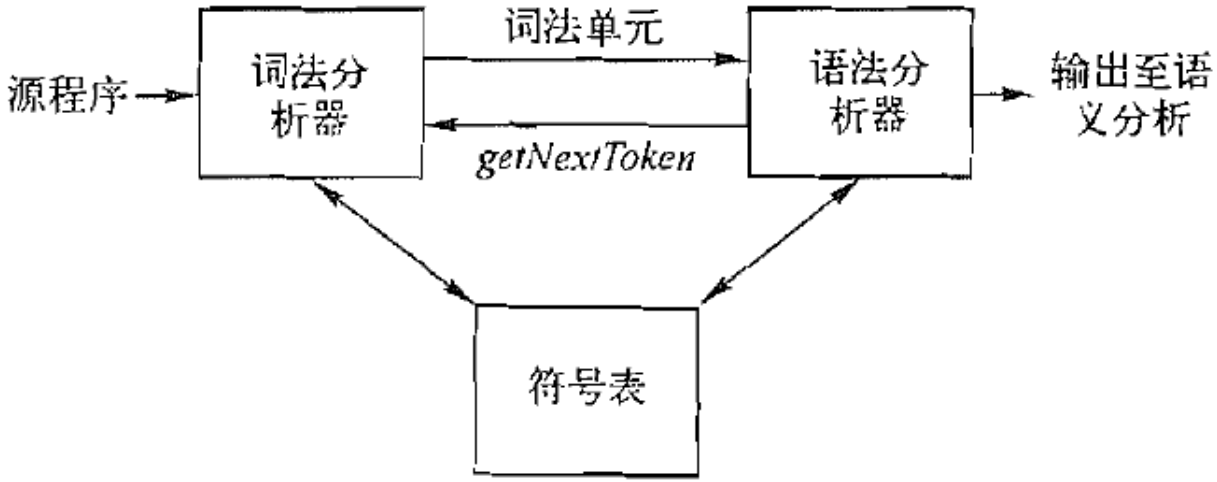
词法分析
(1. 词法分析器生成器 ANTLR v4)

魏恒峰

hfwei@nju.edu.cn

2023 年 03 月 01 日 (周三)

输入: 程序文本/字符串 s + **词法单元 (token) 的规约**

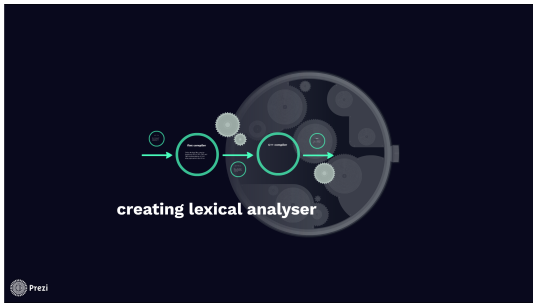


输出: 词法单元流

词法分析器的三种设计方法 (由易到难)

词法分析器生成器

手写词法分析器



自动化词法分析器

生产环境下的编译器 (如 gcc) 通常选择**手写词法分析器**

master ▾ **gcc** / gcc / c-family / **c-lex.c**

iains Objective-C/C++ : Improve '@' keyword locations. ···

19 contributors +7

1435 lines (1278 sloc) | 38.8 KB

master ▾ **gcc** / libcpp / **lex.c**

urnathan cpplib: EOF in pragmas ···

25 contributors +13

4364 lines (3825 sloc) | 119 KB

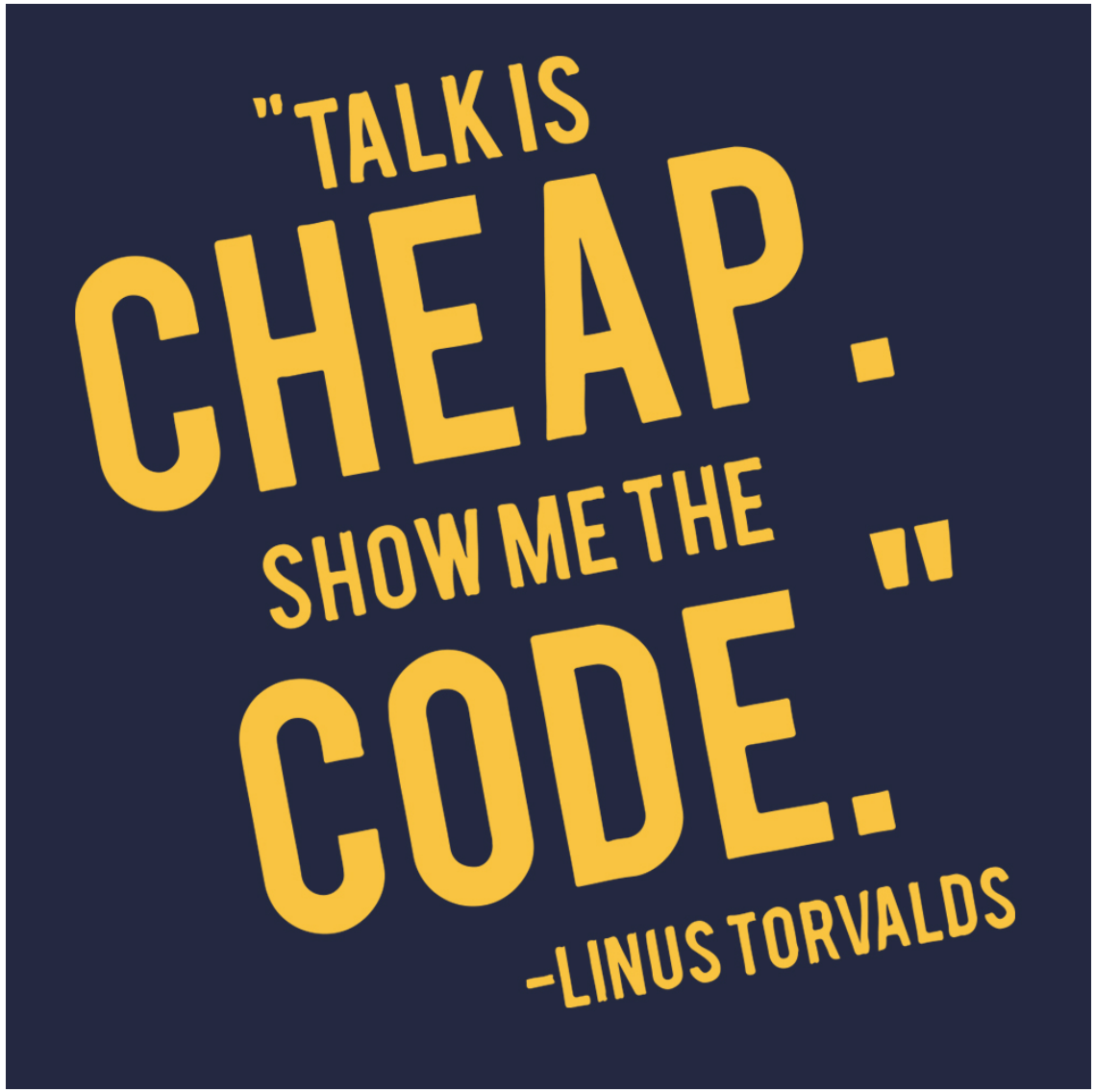
词法分析器生成器

输入： 词法单元的规约

SimpleExpr.g4

输出： 词法分析器

- SimpleExpr.tokens
- SimpleExprLexer.java


"TALK IS
CHEAP.
SHOW ME THE
CODE."
-LINUS TORVALDS

ANTLR v4 中的**冲突解决**规则

最前优先匹配: 关键字 *vs.* 标识符

最长优先匹配: `1.23`, `>=`, `ifhappy`

非贪婪匹配: `*?`, `+?`, `{n,}?`

Let Us Ask ChatGPT to Write a Lexer Using ANTLR 4

以编程的方式使用 ANTLR 4 生成的 xxxLexer.java

```
@header {
package simpleexpr;
}
```

```
CharStream input = CharStreams.fromStream(is);
SimpleExprLexer lexer = new SimpleExprLexer(input);

lexer.getAllTokens().forEach(System.out::println);
```

lexer grammar

Section 4.1 (1) of “ANTLR 4 权威指南”

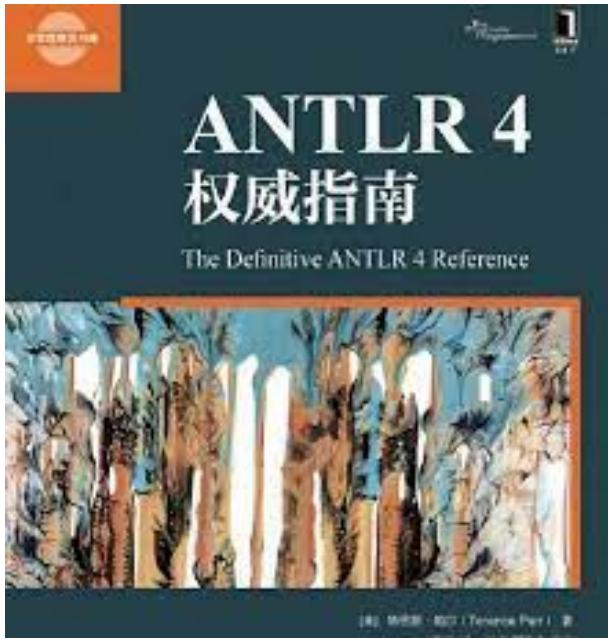
```
lexer grammar SimpleExprLexerRules;

// Comment out the following lines
// Otherwise, there will be duplicate package statements
// @header {
// package simpleexpr;
// }
```

```
grammar SimpleExpr;
import SimpleExprLexerRules;

@header {
package simpleexpr;
}
```

You can learn a lot from grammars-v4/c.



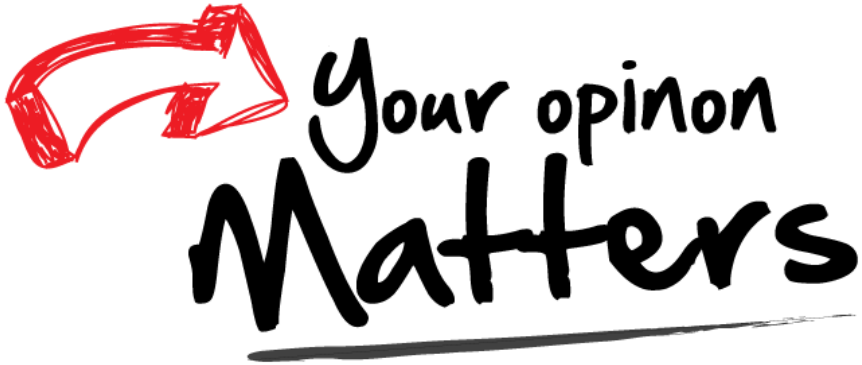
5.5: 识别常见的语法结构

15.5: 词法规则

15.6: 通配符与非贪婪子规则

12: 掌握词法分析的“黑魔法”


Thank
You!



Office 926

hfwei@nju.edu.cn