解释器构造第一次作业

> 本次学习笔记参考的是 *The Definitive ANTLR 4 Reference* 第4.2章: “Building a calculator using a visitor"中描述的内容。

Antlr安装背景

- 操作系统：macOS 10.12 Sierra
- Antlr版本：4.5.3
- IDE：无

计算器的编写过程

g4文件

我们的语法在下列LabeledExpr.g4文件中进行描述。

```
grammar LabeledExpr; //grammar必须与文件名一致

prog: stat+; //程序是由多个语句组成

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
;

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

> 注意上面部分的label (#部分)不是注释，而是为每个选择(alternative)做的标签；如果不写这些标签的话，ANTLR只会为每条文法生成一个访客方法。而我们需要每个选择都拥有自己的访客方法，所以需要为每一个选择打上标签。

编译.g4文件

在.zshrc文件里应该已经写好了别名，这个在安装的时候讲的很清楚了。

```bash
alias antlr4='java -jar /usr/local/lib/antlr-4.5.3-complete.jar'
alias grun='java org.antlr.v4.gui.TestRig'
```

所以我们直接编译：

```bash
$ antlr4 -no-listener -visitor LabeledExpr.g4
```

> 这里-visitor参数很重要，千万别忘记了。

修改 Main.java

这里的Main.java函数在前一章的基础上进行修改。

```java
LabeledExprLexer lexer = new LabeledExprLexer(input);
CommonTokenStream tokens = new CommonTokenStream(lexer);
LabeledExprParser parser = new LabeledExprParser(tokens);
ParseTree tree = parser.prog();

/*待完成*/
EvalVisitor eval = new EvalVisitor();
eval.visit(tree);
```

现在还剩Visitor访客需要我们手动完成。

编写visitor

先看一下编译.g4文件产生的java/interface文件

```java
public interface LabeledExprVisitor<T> {
T visitId(LabeledExprParser.IdContext ctx); // from label id
T visitAssign(LabeledExprParser.AssignContext ctx); // from label assign
T visitMulDiv(LabeledExprParser.MulDivContext ctx); // from label MulDiv
...
}
```

这个接口定义用范型来代表访客函数的返回值类型。这样比较利于我们后续的自定义。

LabeledExprBaseVisitor.java

这个文件是antlr缺省生成的，我们可以通过继承这个类来编写我们的访客类。

LabeledExprBaseVisitor.java:

```java
// Generated from LabeledExpr.g4 by ANTLR 4.5.3
import org.antlr.v4.runtime.tree.AbstractParseTreeVisitor;

/**
 * This class provides an empty implementation of {@link LabeledExprVisitor},
 * which can be extended to create a visitor which only needs to handle a subset
 * of the available methods.
 *
 * @param <T> The return type of the visit operation. Use {@link Void} for
 * operations with no return type.
 */
public class LabeledExprBaseVisitor<T> extends AbstractParseTreeVisitor<T> implements LabeledE

    @Override public T visitProg(LabeledExprParser.ProgContext ctx) { return visitChildren(ctx

    @Override public T visitPrintExpr(LabeledExprParser.PrintExprContext ctx) { return visitCh

    @Override public T visitAssign(LabeledExprParser.AssignContext ctx) { return visitChildren

    @Override public T visitBlank(LabeledExprParser.BlankContext ctx) { return visitChildren(c

    @Override public T visitParens(LabeledExprParser.ParensContext ctx) { return visitChildren

    @Override public T visitMulDiv(LabeledExprParser.MulDivContext ctx) { return visitChildren

    @Override public T visitAddSub(LabeledExprParser.AddSubContext ctx) { return visitChildren

    @Override public T visitId(LabeledExprParser.IdContext ctx) { return visitChildren(ctx); }

    @Override public T visitInt(LabeledExprParser.IntContext ctx) { return visitChildren(ctx);
}
```

编写EvalVisitor.java

文件内容具体见附件，这里列出一小块。

```java
    @Override
    public Integer visitMulDiv(LabeledExprParser.MulDivContext ctx) {
       int left = visit(ctx.expr(0)); // get value ofleft subexpression
       int right = visit(ctx.expr(1)); // get value ofright subexpression
       if ( ctx.op.getType() == LabeledExprParser.MUL ) return left * right;
       return left / right; // must be DIV
    }
```

总之就是反复的使用visit函数，以及从父类中继承方法，其他函数类似。

测试

在非IDE环境中，

```Bash
$ javac Main.java LabeledExpr*.java
```

如果没有报错的话，写一个测试文件t.expr:

```
193
a = 5
b = 6
a + b*2
(1+2)*3
```

最后运行Main函数，得到的结果为：

```
193
17
9
```

说明程序运行正确。

总结

学习过程中遇到了以下问题

- g4文件的语法问题
- visitor模式的理解问题

最终通过查资料的方式解决了上述问题，完成了本次实验。

2016-10-14

2014302580341 余璞轩