Project Description

In this project a Lexical Analyzer and Parser for the Java Language, to identify the language syntax and to create the parsing tree

The lexical analyzer scans the input file and tokenizes the program according to the rules rules defined in the .g4 file. The parser takes the tokens produced by the lexer and tries to determine if proper sentences are correct at the syntaxical level instead of the word/characater level.

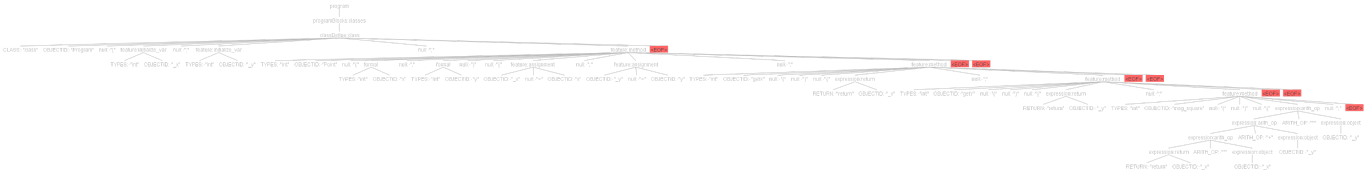
Important Notes

Order of lexical rules is of utmost importance when designing the parser, also. the rules must be consistent to avoid errors.

Test Cases

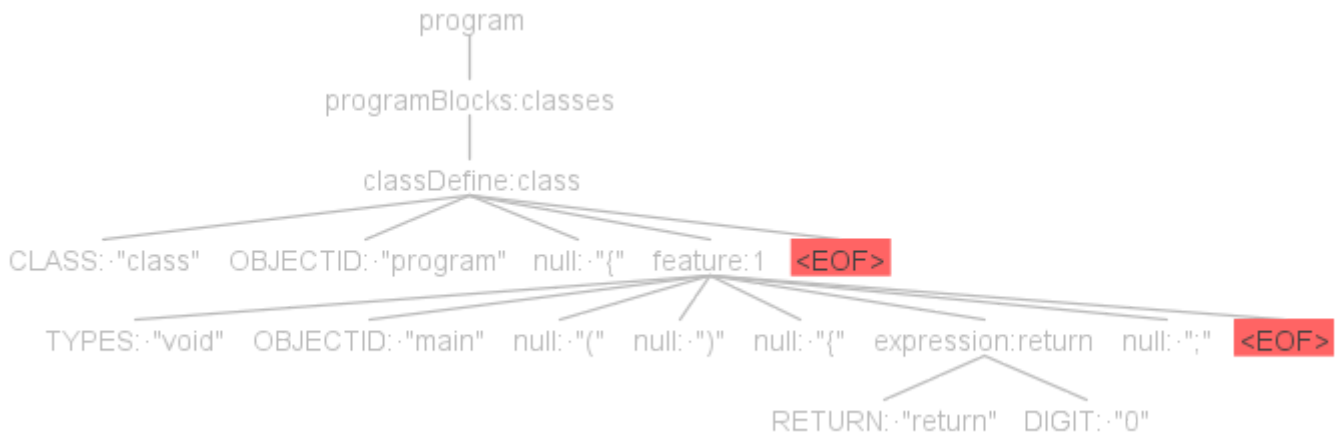
Program 1

```
class Program{

    //x coordinate
    int _x;

    //y coordinate
    int _y;

    int Point(int x, int y) { _x = x; _y = y; }

    int getX() { return _x; }
    int getY() { return _y; }

    int mag_square() {
      return _x*_x +
      _y*_y;
      }

}
```



Program 2

```
//Here is a comment
/* additional comment */ class program { void main() { return 0; } }
```

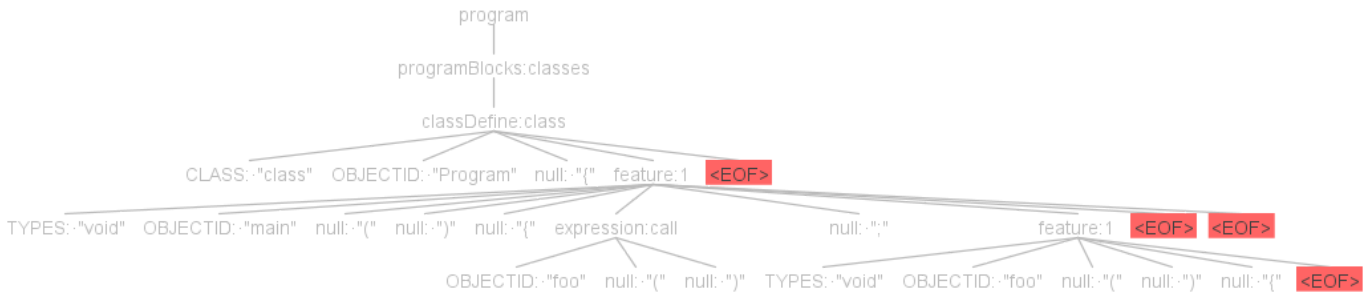


\

Program 3

(legal-18 in MIT test files)

```
class Program {
  void main() {
    foo();  // semanitcally bad, but gramatically ok
  }

  void foo() {}
}
```


program
programBlocks:classes
classDefine:class
CLASS:·"class"
OBJECTID:·"Program"
null:·"{"
feature:1
<EOF>
TYPES:·"void"
OBJECTID:·"main"
null:·"("
null:·")"
null:·"{"
expression:call
null:·";"
feature:1
<EOF>
<EOF>
OBJECTID:·"foo"
null:·"("
null:·")"
TYPES:·"void"
OBJECTID:·"foo"
null:·"("
null:·")"
null:·"{"
<EOF>