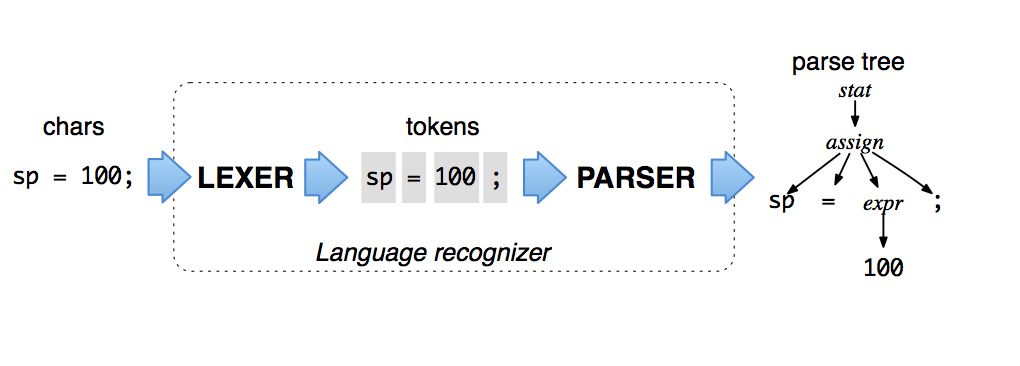
ANTLR生成的解析器构建一个称为解析树或语法树的数据结构，记录解析器如何识别输入句子和组件短语的结构。

[](https://github.com/antlr/antlr4/blob/master/doc/images/process.png)

ANTLR有一个ParseTreeWalker，它知道如何遍历这些解析树并触发您可以创建的侦听器实现对象中的事件。ANTLR工具也会为您生成侦听器接口，除非您使用命令行选项将其关闭。

public interface JavaListener extends ParseTreeListener<Token> {

void enterClassDeclaration(JavaParser.ClassDeclarationContext ctx);

void exitClassDeclaration(JavaParser.ClassDeclarationContext ctx);

void enterMethodDeclaration(JavaParser.MethodDeclarationContext ctx);

...

}

其中，解析器语法中的每个规则都有一个enter和exit方法。

假设您已经创建了一个调用的侦听器对象MyListener，这里是如何调用Java解析器并遍历解析树：

JavaLexer lexer = new JavaLexer(input);

CommonTokenStream tokens = new CommonTokenStream(lexer);

JavaParser parser = new JavaParser(tokens);

JavaParser.CompilationUnitContext tree = parser.compilationUnit(); // parse a compilationUnit

MyListener extractor = new MyListener(parser);

ParseTreeWalker.DEFAULT.walk(extractor, tree); // initiate walk of tree with listener in use of default walker