**Google翻译太乱，给个例子自己看**

MyGrammar.g4

grammar MyGrammar; // MyGrammarBaseVisitor Class

program // VisitProgram(ProgramContext context) Actiono

: expression\*

;

expression

: noValue

| hasValue

;

noValue

: Type Identifier ';' { System.Console.WriteLine("var name:"+$Identifier.text); }

;

hasValue

: Type Identifier '=' Value ';' { System.Console.WriteLine("var name:"+$Identifier.text); }

;

Type: 'int'|'void'|'string';

Identifier : [a-zA-Z\_$] [a-zA-Z\_$0-9]\* ; // simplified

Value : [a-zA-Z0-9]\* ;

WS : [\t\r\n\u0020\u3000]+ **->** skip ;

示例代码：

public static void Test1()

{

string input = "int a = 1 ;";

var stream = new AntlrInputStream(input);

// 解析词法

var lexer = new MyGrammarLexer(stream);

var tokens = new CommonTokenStream(lexer);

// 解析语法

var parser = new MyGrammarParser(tokens);

// 获取语法树

var tree = parser.hasValue();

// 访问语法树

var result = new MyGrammarVisitor().Visit(tree);

// 从 hasValue( 表达式为 Type Identifier '=' Value ';' ) 树生成一个解析树 int a = <Value> ;

ParseTreePattern p = parser.CompileParseTreePattern("int a = <Value> ;", MyGrammarParser.RULE\_hasValue);

// 判断解析树是否与 tree 返回的树匹配

ParseTreeMatch m = p.Match(tree);

if (m.Succeeded)

{

Console.WriteLine("匹配成功");

Console.WriteLine(m.Get("Value"));

}

else

{

Console.WriteLine("匹配不成功");

}

Console.WriteLine(tree.ToStringTree(parser));

Console.WriteLine(result);

}

public static void Test2() {

string input = "int a = 1 ;int a = 2 ;void a = 3 ;";

var stream = new AntlrInputStream(input);

// 解析词法

var lexer = new MyGrammarLexer(stream);

var tokens = new CommonTokenStream(lexer);

// 解析语法

var parser = new MyGrammarParser(tokens);

// 获取语法树

var tree = parser.program();

// 访问语法树

var result = new MyGrammarVisitor().Visit(tree);

// 从 hasValue( 表达式为 Type Identifier '=' Value ';' ) 树生成一个解析树 int a = <Value> ;

ParseTreePattern p = parser.CompileParseTreePattern("int a = <Value> ;", MyGrammarParser.RULE\_hasValue);

// 查找 tree 下的expression下的所有解析树，找出与 p 匹配的解析树

List<ParseTreeMatch> matches = p.FindAll(tree, "//expression/\*").ToList();

matches.ForEach(item =>

{

Console.WriteLine(item.Tree.ToStringTree());

});

Console.WriteLine(tree.ToStringTree(parser));

Console.WriteLine(result);

}

**使用XPath标识解析树节点集**

路径是一系列具有以下分隔符的节点名称。

| **表达** | **描述** |
| --- | --- |
| 节点名称 | 具有令牌或规则名称节点名称的节点 |
| / | 因为它看起来更好，所以使用/（或//以下）启动所有模式。 |
| // | 例如，//ID查找ID树中的所有令牌节点。 |
| ！ | 例如，/classdef/!field应该找到非子树的classdef根节点的所有子节点field。 |