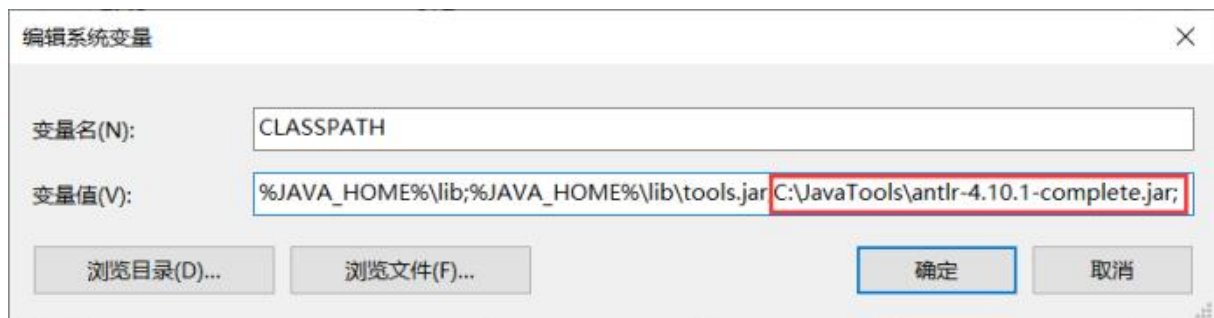


Antlr 生成 AST

1. 安装 antlr 运行环境

(参考文档 <https://github.com/antlr/antlr4/blob/master/doc/getting-started.md>) 在 ANTLR 官网 (<https://www.antlr.org/download.html>) 点击红框内的链接下载 antlr-4.10.1-complete.jar。

编辑系统变量 CLASSPATH，添加 antlr-4.10.1-complete.jar 所在的路径。



检查是否安装成功，在命令行中输入指令 `java org.antlr.v4.Tool`，出现以下输出说明安装成功。

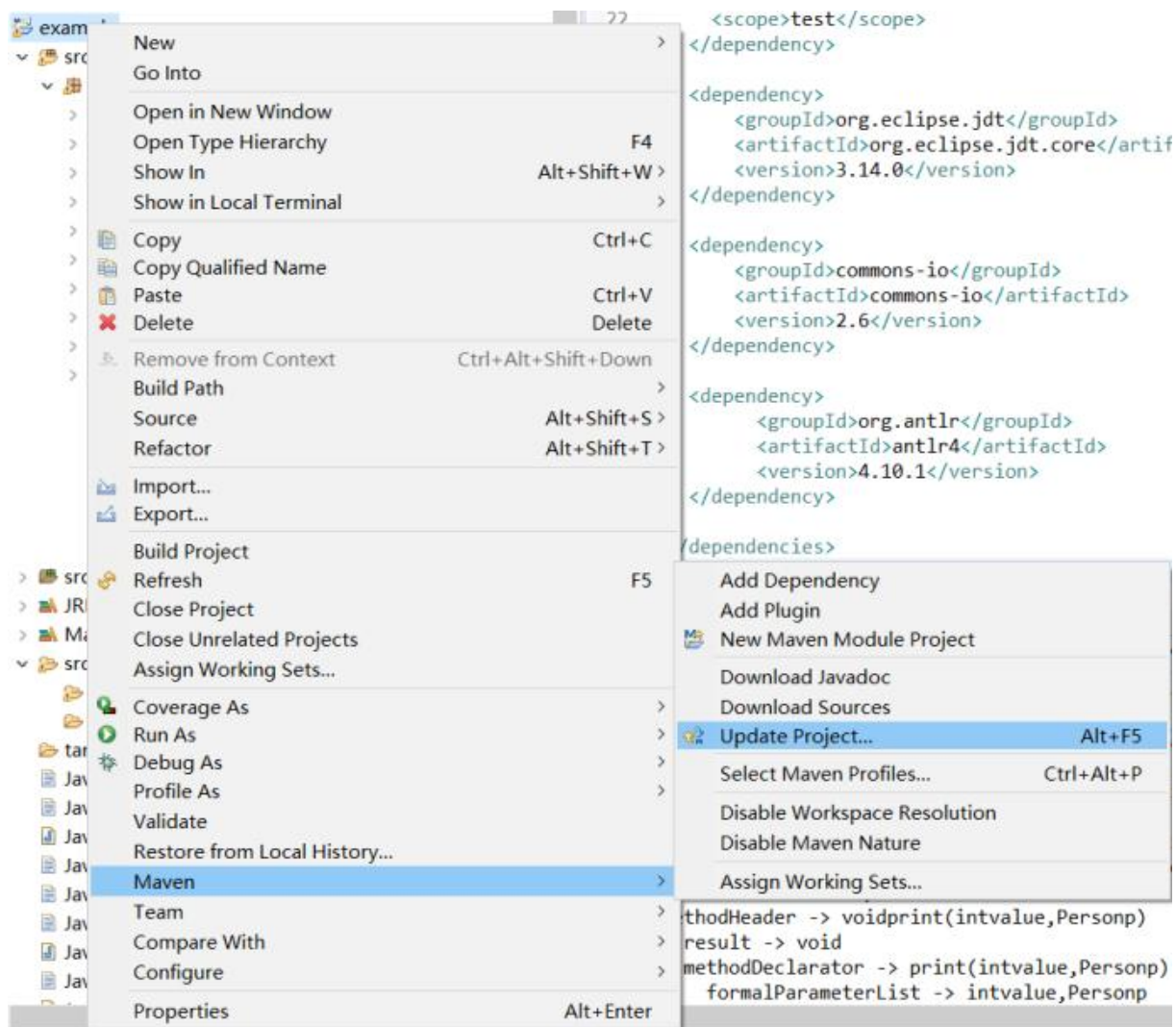
```
ANTLR Parser Generator Version 4.10.1
-o _____ specify output directory where all output is generated
-lib _____ specify location of grammars, tokens files
-atn _____ generate rule augmented transition network diagrams
-encoding _____ specify grammar file encoding: e.g., euc-jp
-message-format _____ specify output style for messages in antlr, gnu, vs2005
-long-messages show exception details when available for errors and warnings
-listener generate parse tree listener (default)
-no-listener don't generate parse tree listener
-visitor generate parse tree visitor
-no-visitor don't generate parse tree visitor (default)
-package _____ specify a package/namespace for the generated code
-depend generate file dependencies
-D<option>=value set/override a grammar-level option
-Werror treat warnings as errors
-XdbgST launch StringTemplate visualizer on generated code
-XdbgSTWait wait for STViz to close before continuing
-Xforce-atn use the ATN simulator for all predictions
-Xlog dump lots of logging info to antlr-timestamp.log
-Xexact-output-dir all output goes into -o dir regardless of paths/package
```

2. 使用 antlr

在 eclipse 中新建一个 maven 项目，在 pom 文件中添加依赖：

```
<dependency>
  <groupId>org.antlr</groupId>
  <artifactId>antlr4</artifactId>
  <version>4.10.1</version>
</dependency>
```

右键项目—>【Maven】—>【Update Project】，下载 jar 包。



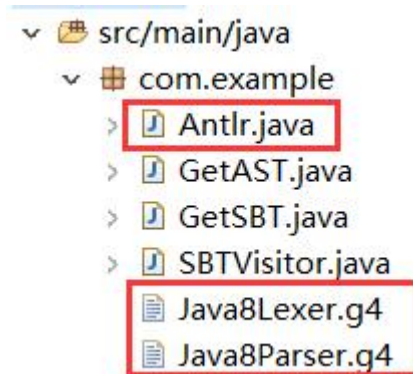
在 <https://github.com/antlr/grammars-v4> 下载 grammar 文件。以下载 java8 的 grammar 文件为例：在 <https://github.com/antlr/grammars-v4/tree/master/java/java8> 下载红框中的两个文件。

(update: 用 java8 的 grammar 文件生成的 AST 不太好，最好下载 java 的 grammar 文件 `JavaLexer/JavaParser`，地址：

<https://github.com/antlr/grammars-v4/tree/master/java/java>，其他步骤一样)

kaby76 Renaming "emptyStatement" to eliminate symbol conflict in Go target.			442bale on 13 Dec 2021	History
examples	Give examples for java instance of parsing correctly.		14 months ago	
CodepointRangeScanner.java	swift java asm into modules		2 years ago	
Java8Lexer.g4	swift java asm into modules		2 years ago	
Java8Parser.g4	Renaming "emptyStatement" to eliminate symbol conflict in Go target.		6 months ago	
Test.java	Fix Test.java for java8/9 grammars		2 years ago	
pom.xml	Fix for #2051 - All parent entries in the pom files using relativePat...		15 months ago	

放到项目对应的文件夹中。(Antlr.java 使用 antlr 输出 AST，代码在步骤 3)

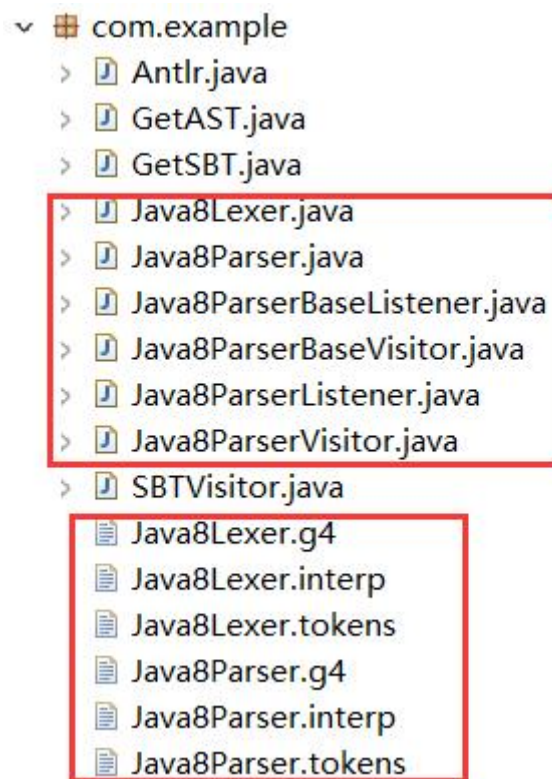


打开命令行，输入下面的 3 个指令，会生成几个新的文件

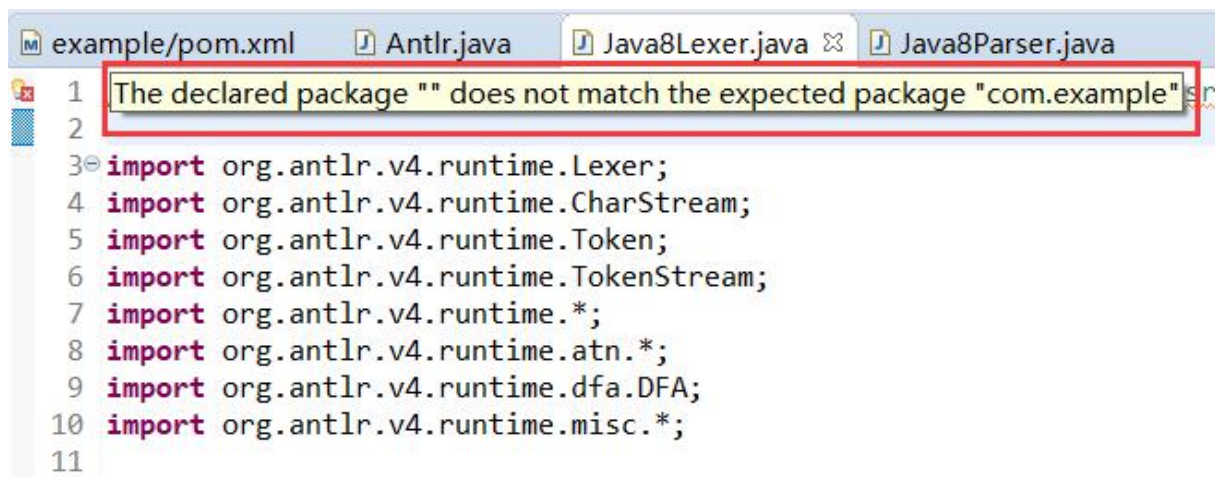
```
doskey antlr4=java org.antlr.v4.Tool $*
```

```
antlr4 -visitor 【Java8Lexer.g4 文件的路径】
```

```
antlr4 -visitor 【Java8Parser.g4 文件的路径】
```



如果新生成的文件出现下图的错误，原因是没有添加 package 语句。



```
example/pom.xml Antlr.java Java8Lexer.java Java8Parser.java
1 The declared package "" does not match the expected package "com.example"
2
3 import org.antlr.v4.runtime.Lexer;
4 import org.antlr.v4.runtime.CharStream;
5 import org.antlr.v4.runtime.Token;
6 import org.antlr.v4.runtime.TokenStream;
7 import org.antlr.v4.runtime.*;
8 import org.antlr.v4.runtime.atn.*;
9 import org.antlr.v4.runtime.dfa.DFA;
10 import org.antlr.v4.runtime.misc.*;
11
```

添加 package 语句



```
2 package com.example;
3
4 import org.antlr.v4.runtime.Lexer;
5 import org.antlr.v4.runtime.CharStream;
6 import org.antlr.v4.runtime.Token;
7 import org.antlr.v4.runtime.TokenStream;
8 import org.antlr.v4.runtime.*;
9 import org.antlr.v4.runtime.atn.*;
10 import org.antlr.v4.runtime.dfa.DFA;
11 import org.antlr.v4.runtime.misc.*;
```

3. 测试程序

```
package com.example;
```

```
import java.io.File;
```

```
import java.io.IOException;
```

```
import java.nio.charset.Charset;
```

```
import java.nio.file.Files;
```

```
import org.antlr.v4.runtime.*;
```

```
import org.antlr.v4.runtime.tree.*;
```

```
public class Antlr {
```

```

public static void main(String args[]) throws IOException {

    String inputString = "class Test {public String extractFor(Integer
id){LOG.debug(\"Extracting method with ID:{}\", id);return requests.remove(id);}}";

    ANTLRInputStream input = new ANTLRInputStream(inputString);

    Java8Lexer lexer = new Java8Lexer(input);

    CommonTokenStream tokens = new CommonTokenStream(lexer);

    Java8Parser parser = new Java8Parser(tokens);

    ParserRuleContext ctx = parser.classDeclaration();

    printAST(ctx, false, 0);
}

private static void printAST(RuleContext ctx, boolean verbose, int indentation) {

    boolean toBelgnored = !verbose && ctx.getChildCount() == 1 && ctx.getChild(0)
instanceof ParserRuleContext;
//      boolean toBelgnored = false ;

    if (!toBelgnored) {

        String ruleName = Java8Parser.ruleNames[ctx.getRuleIndex()];

        for (int i = 0; i < indentation; i++) {

            System.out.print(" ");

        }

        System.out.println(ruleName + " -> " + ctx.getText());

    }

    for (int i = 0; i < ctx.getChildCount(); i++) {

```

```

ParseTree element = ctx.getChild(i);

if (element instanceof RuleContext) {

    printAST((RuleContext) element, verbose, indentation + (toBelgnored ? 0 :
1));

}

else if(element instanceof TerminalNode)

{

TerminalNode tn = (TerminalNode)element;

for (int j = 0; j <= indentation; j++) {

    System.out.print(" ");

}

    System.out.println(Java8Parser.tokenNames[tn.getSymbol().getType()] + " ->
" + tn.getText());

}

}

}

}

```

4. 节点的 label

在生成的 JavaParser.java 文件中可以看到所有节点 label 的集合。

```

84 private static String[] makeRuleNames() {
85     return new String[] {
86         "compilationUnit", "packageDeclaration", "importDeclaration", "typeDeclaration",
87         "modifier", "classOrInterfaceModifier", "variableModifier", "classDeclaration",
88         "typeParameters", "typeParameter", "typeBound", "enumDeclaration", "enumConstants",
89         "enumConstant", "enumBodyDeclarations", "interfaceDeclaration", "classBody",
90         "interfaceBody", "classBodyDeclaration", "memberDeclaration", "methodDeclaration",
91         "methodBody", "typeTypeOrVoid", "genericMethodDeclaration", "genericConstructorDeclaration",
92         "constructorDeclaration", "fieldDeclaration", "interfaceBodyDeclaration",
93         "interfaceMemberDeclaration", "constDeclaration", "constantDeclarator",
94         "interfaceMethodDeclaration", "interfaceMethodModifier", "genericInterfaceMethodDeclaration",
95         "interfaceCommonBodyDeclaration", "variableDeclarators", "variableDeclarator",
96         "variableDeclaratorId", "variableInitializer", "arrayInitializer", "classOrInterfaceType",
97         "typeArgument", "qualifiedNameList", "formalParameters", "receiverParameter",
98         "formalParameterList", "formalParameter", "lastFormalParameter", "lambdaLVTIList",
99         "lambdaLVTIPParameter", "qualifiedName", "literal", "integerLiteral",
100        "floatLiteral", "altAnnotationQualifiedName", "annotation", "elementValuePairs",
101        "elementValuePair", "elementValue", "elementValueArrayInitializer", "annotationTypeDeclaration",
102        "annotationTypeBody", "annotationTypeElementDeclaration", "annotationTypeElementRest",
103        "annotationMethodOrConstantRest", "annotationMethodRest", "annotationConstantRest",
104        "defaultValue", "moduleDeclaration", "moduleBody", "moduleDirective",
105        "requiresModifier", "recordDeclaration", "recordHeader", "recordComponentList",
106        "recordComponent", "recordBody", "block", "blockStatement", "localVariableDeclaration",
107        "identifier", "localTypeDeclaration", "statement", "catchClause", "catchType",
108        "finallyBlock", "resourceSpecification", "resources", "resource", "switchBlockStatementGroup",
109        "switchLabel", "forControl", "forInit", "enhancedForControl", "parExpression",
110        "expressionList", "methodCall", "expression", "pattern", "lambdaExpression",
111        "lambdaParameters", "lambdaBody", "primary", "switchExpression", "switchLabeledRule",
112        "guardedPattern", "switchRuleOutcome", "classType", "creator", "createdName",
113        "innerCreator", "arrayCreatorRest", "classCreatorRest", "explicitGenericInvocation",
114        "typeArgumentsOrDiamond", "nonWildcardTypeArgumentsOrDiamond", "nonWildcardTypeArguments",
115        "typeList", "typeType", "primitiveType", "typeArguments", "superSuffix",
116        "explicitGenericInvocationSuffix", "arguments"
117     };
118 }

```

Java:

1. compilationUnit
2. packageDeclaration
3. importDeclaration
4. typeDeclaration
5. modifier
6. classOrInterfaceModifier
7. variableModifier
8. classDeclaration
9. typeParameters
10. typeParameter
11. typeBound
12. enumDeclaration
13. enumConstants
14. enumConstant
15. enumBodyDeclarations
16. interfaceDeclaration
17. classBody
18. interfaceBody
19. classBodyDeclaration
20. memberDeclaration
21. methodDeclaration
22. methodBody
23. typeTypeOrVoid
24. genericMethodDeclaration
25. genericConstructorDeclaration

- 26. constructorDeclaration
- 27. fieldDeclaration
- 28. interfaceBodyDeclaration
- 29. interfaceMemberDeclaration
- 30. constDeclaration
- 31. constantDeclarator
- 32. interfaceMethodDeclaration
- 33. interfaceMethodModifier
- 34. genericInterfaceMethodDeclaration
- 35. interfaceCommonBodyDeclaration
- 36. variableDeclarators
- 37. variableDeclarator
- 38. variableDeclaratorId
- 39. variableInitializer
- 40. arrayInitializer
- 41. classOrInterfaceType
- 42. typeArgument
- 43. qualifiedNameList
- 44. formalParameters
- 45. receiverParameter
- 46. formalParameterList
- 47. formalParameter
- 48. lastFormalParameter
- 49. lambdaLVTLList
- 50. lambdaLVTIPParameter
- 51. qualifiedName
- 52. literal
- 53. integerLiteral
- 54. floatLiteral
- 55. altAnnotationQualifiedName
- 56. annotation
- 57. elementValuePairs
- 58. elementValuePair
- 59. elementValue
- 60. elementValueArrayInitializer
- 61. annotationTypeDeclaration
- 62. annotationTypeBody
- 63. annotationTypeElementDeclaration
- 64. annotationTypeElementRest
- 65. annotationMethodOrConstantRest
- 66. annotationMethodRest
- 67. annotationConstantRest
- 68. defaultValue
- 69. moduleDeclaration
- 70. moduleBody

- 71. moduleDirective
- 72. requiresModifier
- 73. recordDeclaration
- 74. recordHeader
- 75. recordComponentList
- 76. recordComponent
- 77. recordBody
- 78. block
- 79. blockStatement
- 80. localVariableDeclaration
- 81. identifier
- 82. localTypeDeclaration
- 83. statement
- 84. catchClause
- 85. catchType
- 86. finallyBlock
- 87. resourceSpecification
- 88. resources
- 89. resource
- 90. switchBlockStatementGroup
- 91. switchLabel
- 92. forControl
- 93. forInit
- 94. enhancedForControl
- 95. parExpression
- 96. expressionList
- 97. methodCall
- 98. expression
- 99. pattern
- 100. lambdaExpression
- 101. lambdaParameters
- 102. lambdaBody
- 103. primary
- 104. switchExpression
- 105. switchLabeledRule
- 106. guardedPattern
- 107. switchRuleOutcome
- 108. classType
- 109. creator
- 110. createdName
- 111. innerCreator
- 112. arrayCreatorRest
- 113. classCreatorRest
- 114. explicitGenericInvocation
- 115. typeArgumentsOrDiamond

- 116. nonWildcardTypeArgumentsOrDiamond
- 117. nonWildcardTypeArguments
- 118. typeList
- 119. typeType
- 120. primitiveType
- 121. typeArguments
- 122. superSuffix
- 123. explicitGenericInvocationSuffix
- 124. arguments

Python:

- 1. root
- 2. single_input
- 3. file_input
- 4. eval_input
- 5. stmt
- 6. compound_stmt
- 7. suite
- 8. decorator
- 9. elif_clause
- 10. else_clause
- 11. finally_clause
- 12. with_item
- 13. except_clause
- 14. classdef
- 15. funcdef
- 16. typedargslist
- 17. args
- 18. kwargs
- 19. def_parameters
- 20. def_parameter
- 21. named_parameter
- 22. simple_stmt
- 23. small_stmt
- 24. testlist_star_expr
- 25. star_expr
- 26. assign_part
- 27. exprlist
- 28. import_as_names
- 29. import_as_name
- 30. dotted_as_names
- 31. dotted_as_name
- 32. test
- 33. varargslist
- 34. vardef_parameters
- 35. vardef_parameter

- 36. varargs
- 37. varkwargs
- 38. logical_test
- 39. comparison
- 40. expr
- 41. atom
- 42. dictorsetmaker
- 43. testlist_comp
- 44. testlist
- 45. dotted_name
- 46. name
- 47. number
- 48. integer
- 49. yield_expr
- 50. yield_arg
- 51. trailer
- 52. arguments
- 53. arglist
- 54. argument
- 55. subscriptlist
- 56. subscript
- 57. sliceop
- 58. comp_for
- 59. comp_iter

C++:

- 1. translationUnit
- 2. primaryExpression
- 3. idExpression
- 4. unqualifiedId
- 5. qualifiedId
- 6. nestedNameSpecifier
- 7. lambdaExpression
- 8. lambdaIntroducer
- 9. lambdaCapture
- 10. captureDefault
- 11. captureList
- 12. capture
- 13. simpleCapture
- 14. initcapture
- 15. lambdaDeclarator
- 16. postfixExpression
- 17. typeIdOfTheTypeId
- 18. expressionList
- 19. pseudoDestructorName
- 20. unaryExpression

21. unaryOperator
22. newExpression
23. newPlacement
24. newTypeId
25. newDeclarator
26. noPointerNewDeclarator
27. newInitializer
28. deleteExpression
29. noExceptExpression
30. castExpression
31. pointerMemberExpression
32. multiplicativeExpression
33. additiveExpression
34. shiftExpression
35. shiftOperator
36. relationalExpression
37. equalityExpression
38. andExpression
39. exclusiveOrExpression
40. inclusiveOrExpression
41. logicalAndExpression
42. logicalOrExpression
43. conditionalExpression
44. assignmentExpression
45. assignmentOperator
46. expression
47. constantExpression
48. statement
49. labeledStatement
50. expressionStatement
51. compoundStatement
52. statementSeq
53. selectionStatement
54. condition
55. iterationStatement
56. forInitStatement
57. forRangeDeclaration
58. forRangeInitializer
59. jumpStatement
60. declarationStatement
61. declarationseq
62. declaration
63. blockDeclaration
64. aliasDeclaration
65. simpleDeclaration

- 66.staticAssertDeclaration
- 67.emptyDeclaration
- 68.attributeDeclaration
- 69.declSpecifier
- 70.declSpecifierSeq
- 71.storageClassSpecifier
- 72.functionSpecifier
- 73.typedefName
- 74.typeSpecifier
- 75.trailingTypeSpecifier
- 76.typeSpecifierSeq
- 77.trailingTypeSpecifierSeq
- 78.simpleTypeLengthModifier
- 79.simpleTypeSignednessModifier
- 80.simpleTypeSpecifier
- 81.theTypeName
- 82.decltypeSpecifier
- 83.elaboratedTypeSpecifier
- 84.enumName
- 85.enumSpecifier
- 86.enumHead
- 87.opaqueEnumDeclaration
- 88.enumkey
- 89.enumbase
- 90.enumeratorList
- 91.enumeratorDefinition
- 92.enumerator
- 93.namespaceName
- 94.originalNamespaceName
- 95.namespaceDefinition
- 96.namespaceAlias
- 97.namespaceAliasDefinition
- 98.qualifiednamespacespecifier
- 99.usingDeclaration
- 100.usingDirective
- 101.asmDefinition
- 102.linkageSpecification
- 103.attributeSpecifierSeq
- 104.attributeSpecifier
- 105.alignmentspecifier
- 106.attributeList
- 107.attribute
- 108.attributeNamespace
- 109.attributeArgumentClause
- 110.balancedTokenSeq

111. balancedtoken
112. initDeclaratorList
113. initDeclarator
114. declarator
115. pointerDeclarator
116. noPointerDeclarator
117. parametersAndQualifiers
118. trailingReturnType
119. pointerOperator
120. cvqualifierseq
121. cvQualifier
122. refqualifier
123. declaratorid
124. theTypeId
125. abstractDeclarator
126. pointerAbstractDeclarator
127. noPointerAbstractDeclarator
128. abstractPackDeclarator
129. noPointerAbstractPackDeclarator
130. parameterDeclarationClause
131. parameterDeclarationList
132. parameterDeclaration
133. functionDefinition
134. functionBody
135. initializer
136. braceOrEqualInitializer
137. initializerClause
138. initializerList
139. bracedInitList
140. className
141. classSpecifier
142. classHead
143. classHeadName
144. classVirtSpecifier
145. classKey
146. memberSpecification
147. memberdeclaration
148. memberDeclaratorList
149. memberDeclarator
150. virtualSpecifierSeq
151. virtualSpecifier
152. pureSpecifier
153. baseClause
154. baseSpecifierList
155. baseSpecifier

156.classOrDeclType
157.baseTypeSpecifier
158.accessSpecifier
159.conversionFunctionId
160.conversionTypeId
161.conversionDeclarator
162.constructorInitializer
163.memInitializerList
164.memInitializer
165.meminitializerid
166.operatorFunctionId
167.literalOperatorId
168.templateDeclaration
169.templateparameterList
170.templateParameter
171.typeParameter
172.simpleTemplateId
173.templateId
174.templateName
175.templateArgumentList
176.templateArgument
177.typeNameSpecifier
178.explicitInstantiation
179.explicitSpecialization
180. tryBlock
181.functionTryBlock
182.handlerSeq
183.handler
184.exceptionDeclaration
185.throwExpression
186.exceptionSpecification
187.dynamicExceptionSpecification
188.typeIdList
189.noexceptSpecification
190.theOperator
191.literal