ANTLR

In computer-based language recognition, **ANTLR** (pronounced *antler*), or **ANother Tool for Language Recognition**, is a parser generator that uses LL(*) for parsing. ANTLR is the successor to the **Purdue Compiler Construction Tool Set** (**PCCTS**), first developed in 1989, and is under active development. Its maintainer is Professor Terence Parr of the University of San Francisco.

ANTLR

Original author(s)	Terence Parr and others
Initial release	February 1992
Stable release	4.9.2 / 11 March 2021
Repository	github.com/antlr/antlr4 (https://github.com/antlr/antlr4)
Written in	Java
Platform	Cross-platform
License	BSD License
Website	www.antlr.org (http://www.antlr.org)

Contents

Usage

ANTLR takes as input a grammar that specifies a language and generates as output source code for a recognizer of that language. While Version 3 supported generating code in the programming languages Ada95, ActionScript, C, C#, Java, JavaScript, Objective-C, Perl, Python, Ruby, and Standard ML,[1] Version 4 at present targets C#, C++, Dart,[2][3] Java, JavaScript, Go, PHP, Python (2 and 3), and Swift.

A language is specified using a context-free grammar expressed using Extended Backus–Naur Form (EBNF).[4] ANTLR can generate lexers, parsers, tree parsers, and combined lexer-parsers. Parsers can automatically generate parse trees or abstract syntax trees, which can be further processed with tree parsers. ANTLR provides a single consistent notation for specifying lexers, parsers, and tree parsers.

By default, ANTLR reads a grammar and generates a recognizer for the language defined by the grammar (i.e., a program that reads an input stream and generates an error if the input stream does not conform to the syntax specified by the grammar). If there are no syntax errors, the default action is to simply exit without printing any message. In order to do something useful with the language, actions can be attached to grammar elements in the grammar. These actions are written in the programming language in which the

recognizer is being generated. When the recognizer is being generated, the actions are embedded in the source code of the recognizer at the appropriate points. Actions can be used to build and check symbol tables and to emit instructions in a target language, in the case of a compiler.[4]

Other than lexers and parsers, ANTLR can be used to generate tree parsers. These are recognizers that process abstract syntax trees, which can be automatically generated by parsers. These tree parsers are unique to ANTLR and help processing abstract syntax trees.[4]

Licensing

ANTLR 3 and ANTLR 4 are free software, published under a three-clause BSD License.[5] Prior versions were released as public domain software.[6] Documentation, derived from Parr's book *The Definitive ANTLR 4 Reference*, is included with the BSD-licensed ANTLR 4 source.[5][7]

Various plugins have been developed for the Eclipse development environment to support the ANTLR grammar, including ANTLR Studio, a proprietary product, as well as the "ANTLR 2"[8] and "ANTLR 3"[9] plugins for Eclipse hosted on SourceForge.

ANTLR 4

ANTLR 4 deals with direct left recursion correctly, but not with left recursion in general, i.e., grammar rules *x* that refer to *y* that refer to *x*.[10]

Development

As reported on the tools[11] page of the ANTLR project, plug-ins that enable features like syntax highlighting, syntax error checking and code completion are freely available for the most common IDEs (Intellij IDEA, NetBeans, Eclipse, Visual Studio[12] and Visual Studio Code).

Projects

Software built using ANTLR includes:

- Groovy.[13]
- Jython.[14]
- Hibernate[15]
- OpenJDK Compiler Grammar project experimental version of the javac compiler based upon a grammar written in ANTLR.[16]
- Apex, Salesforce.com's programming language.
- The expression evaluator in Numbers, Apple's spreadsheet.
- Twitter's search query language.
- Weblogic server.
- Apache Cassandra.
- Processing.
- JabRef.
- Presto (SQL query engine)
- MySQL Workbench

Over 200 grammars implemented in ANTLR 4 are available on GitHub.[17] They range from grammars for a URL to grammars for entire languages like C, Java and Go.

Example

In the following example, a parser in ANTLR describes the sum of expressions can be seen in the form of "1 + 2 + 3":

```
// Common options, for example, the target language
options
{
 language = "CSharp";
}
// Followed by the parser
class SumParser extends Parser;
options
{
  k = 1; // Parser Lookahead: 1 Token
}
// Definition of an expression
statement: INTEGER (PLUS^ INTEGER)*;
// Here is the Lexer
class SumLexer extends Lexer;
options
{
  k = 1; // Lexer Lookahead: 1 characters
}
PLUS: '+';
DIGIT: ('0'..'9');
INTEGER: (DIGIT)+;
```

The following listing demonstrates the call of the parser in a program:

```
TextReader reader;
// (...) Fill TextReader with character
SumLexer lexer = new SumLexer(reader);
SumParser parser = new SumParser(lexer);

parser.statement();
```

See also

- Coco/R
- DMS Software Reengineering Toolkit
- JavaCC
- Modular Syntax Definition Formalism
- Parboiled (Java)
- Parsing expression grammar
- SableCC

References

1. SML/NJ Language Processing Tools: User Guide (http://www.classes.cs.uchicago.edu/archive/2007/winter/22610-1/docs/lpt-manual.pdf)
2. "Runtime Libraries and Code Generation Targets" (https://github.com/antlr/antlr4/blob/master/doc/targets.md). *github*. 6 January 2022.

3. https://soft-gems.net/the-antlr4-c-runtime-reached-home
4. Parr, Terence (2013-01-15). *The Definitive ANTLR 4 Reference* (https://books.google.com/books?id=gA9QDwAAQBAJ). Pragmatic Bookshelf. ISBN 978-1-68050-500-9.
5. "antlr4/LICENSE.txt" (https://github.com/antlr/antlr4/blob/master/LICENSE.txt). *GitHub*. 2017-03-30.
6. Parr, Terence (2004-02-05). "licensing stuff" (https://web.archive.org/web/20110718165021/http://www.antlr.org/pipermail/antlr-interest/2004-February/006340.html). *antlr-interest* (Mailing list). Archived from the original (http://www.antlr.org/pipermail/antlr-interest/2004-February/006340.html) on 2011-07-18. Retrieved 2009-12-15.
7. "ANTLR 4 Documentation" (https://github.com/antlr/antlr4/blob/master/doc/index.md). *GitHub*. 2017-03-30.
8. "ANTLR plugin for Eclipse" (http://antlreclipse.sourceforge.net).
9. "ANTLR IDE. An eclipse plugin for ANTLR grammars" (http://antlrv3ide.sourceforge.net).
10. What is the difference between ANTLR 3 & 4 (https://github.com/antlr/antlr4/blob/master/doc/faq/general.md#what-is-the-difference-between-antlr-3-and-4)
11. "ANTLR Development Tools" (http://www.antlr.org/tools.html).
12. "ANTLR Language Support - Visual Studio Marketplace" (https://marketplace.visualstudio.com/items?itemName=SamHarwell.ANTLRLanguageSupport).
13. "GroovyRecognizer (Groovy 2.4.0)" (http://docs.groovy-lang.org/2.4.0/html/api/org/codehaus/groovy/antlr/parser/GroovyRecognizer.html).
14. "Jython: 31d97f0de5fe" (https://hg.python.org/jython/rev/31d97f0de5fe).
15. Ebersole, Steve (2018-12-06). "Hibernate ORM 6.0.0.Alpha1 released" (https://in.relation.to/2018/12/06/hibernate-orm-600-alpha1-out/). *In Relation To, The Hibernate team blog on everything data*. Retrieved 2020-07-11.
16. "OpenJDK: Compiler Grammar" (https://openjdk.java.net/projects/compiler-grammar/).
17. *Grammars written for ANTLR v4; expectation that the grammars are free of actions.: antlr/grammars-v4* (https://github.com/antlr/grammars-v4), Antlr Project, 2019-09-25, retrieved 2019-09-25

Bibliography

- Parr, Terence (May 17, 2007), *The Definitive Antlr Reference: Building Domain-Specific Languages* (http://www.pragprog.com/titles/tpantlr/the-definitive-antlr-reference) (1st ed.), Pragmatic Bookshelf, p. 376, ISBN 978-0-9787392-5-6
- Parr, Terence (December 2009), *Language Implementation Patterns: Create Your Own Domain-Specific and General Programming Languages* (http://www.pragprog.com/titles/tpdsl/language-implementation-patterns) (1st ed.), Pragmatic Bookshelf, p. 374, ISBN 978-1-934356-45-6
- Parr, Terence (January 15, 2013), *The Definitive ANTLR 4 Reference* (http://pragprog.com/book/tpantlr2/the-definitive-antlr-4-reference) (1st ed.), Pragmatic Bookshelf, p. 328, ISBN 978-1-93435-699-9

Further reading

- Parr, T.J.; Quong, R.W. (July 1995). "ANTLR: A Predicated-LL(k) Parser Generator". *Software: Practice and Experience*. **25** (7): 789–810. CiteSeerX 10.1.1.54.6015 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.6015). doi:10.1002/spe.4380250705 (http

s://doi.org/10.1002%2Fspe.4380250705). S2CID 13453016 (https://api.semanticscholar.org/CorpusID:13453016).

External links

- Official website (http://www.antlr.org)
- ANTLR (mega) Tutorial (https://tomassetti.me/antlr-mega-tutorial/)
- Why Use ANTLR? (http://www.bearcave.com/software/antlr/antlr_expr.html)
- ANTLR Studio (http://www.placidsystems.com/antlrstudio.aspx)