

Ragel

Ragel is a finite-state machine compiler and a parser generator. Initially Ragel supported output for C, C++ and Assembly source code,^[1] and was expanded to support several other languages including Objective C, D, Go, Ruby, and Java.^[2] Additional language support is also in development.^[3] It supports the generation of table or control flow driven state machines from regular expressions^[4] and/or state charts and can also build lexical analysers via the longest-match method. Ragel specifically targets text parsing and input validation.^[5]

Contents

Overview

Syntax


See also

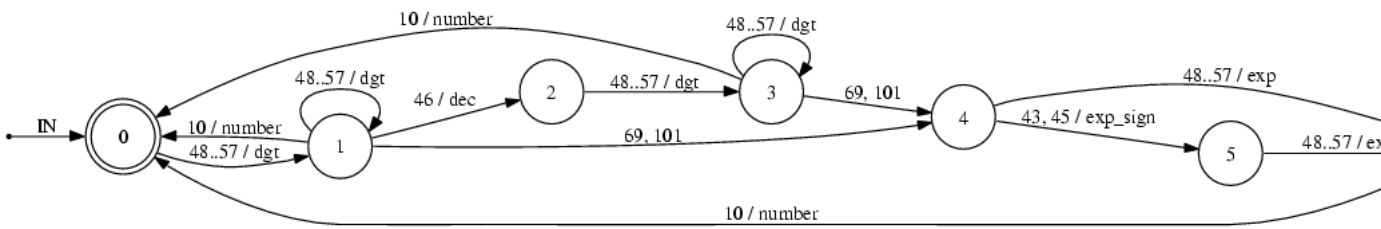
References

External links

Overview

Ragel supports the generation of table or control flow driven state machines from regular expressions and/or state charts and can also build lexical analysers via the longest-match method. A unique feature of Ragel is that user actions can be associated with arbitrary state machine transitions using operators that are integrated into the regular expressions. Ragel also supports visualization of the generated machine via graphviz.

Ragel	
Developer(s)	Adrian Thurston ^[6]
Stable release	6.10 / <div>March 24, 2017</div>
Preview release	7.0.4 / <div>February 16, 2021</div>
Repository	<div>github.com</div> <div>/adrian-thurston/ragel</div> <div>(https://github.com/adrian-thurston/ragel) </div>
Written in	<u>C++</u>
Operating system	<u>Unix-like</u> , <u>Windows</u>
Type	<u>State machine compiler</u>
License	"Ragel 6 remains under <u>GPL v2</u> [generated code] covered by the MIT (or <u>GPL v2</u>)". ^[7] Ragel 7: <u>MIT License</u>
Website	<div>www.colm.net/open-source/ragel/ (http://www.colm.net/open-source/ragel/)</div>



The above graph represents a state-machine that takes user input as a series of bytes representing ASCII characters and control codes. 48..57 is equivalent to the regular expression `[0-9]` (i.e. any digit), so only sequences beginning with a digit can be recognised. If 10 (line feed) is encountered, we're done. 46 is the decimal point ('.'), 43 and 45 are positive and negative signs ('+', '-') and 69/101 is uppercase/lowercase 'e' (to indicate a number in scientific format). As such it will recognize the following properly:

```
2
45
055
78.1
2e5
78.3e12
69.0e-3
3e+3
```

but not:

```
.3
46.
-5
3.e2
2e5.1
```

Syntax

Ragel's input is a regular expression only in the sense that it describes a regular language; it is usually not written in a concise regular expression, but written out into multiple parts like in Extended Backus–Naur form. For example, instead of supporting POSIX character classes in regex syntax, Ragel implements them as built-in production rules. As with usual parser generators, Ragel allows for handling code for productions to be written with the syntax.^[8] The code yielding the above example from the official website is:

```
action dgt      { printf("DGT: %c\n", fc); }
action dec      { printf("DEC: .\n"); }
action exp      { printf("EXP: %c\n", fc); }
action exp_sign { printf("SGN: %c\n", fc); }
action number   { /*NUMBER*/ }

# A floating-point number literal.
number = (
    [0-9]+ $dgt ( '.' @dec [0-9]+ $dgt )?
    ( [eE] ( [+\\-] $exp_sign )? [0-9]+ $exp )?
) %number;

main := ( number '\\n' )*;
```

See also

- Comparison of parser generators
- Executable UML
- Finite-state machine
- Regular expression
- Thompson's construction - the algorithm used by Ragel
- Umple
- Helsinki Finite-State Technology (HFST)

References

- Adrian D. Thurston. "Parsing Computer Languages with an Automaton Compiled from a Single Regular Expression. (http://www.complang.org/thurston/thurston_CIAA_06_sing_regex.pdf) Archived (https://web.archive.org/web/20120907061836/http://www.complang.org/thurston/thurston_CIAA_06_sing_regex.pdf) 2012-09-07 at the Wayback Machine" In: *11th International Conference on Implementation and Application of Automata (CIAA 2006), Lecture Notes in Computer Science, volume 4094*, p. 285-286, Taipei, Taiwan, August 2006.
- "Ragel User Guide" (<http://www.colm.net/files/ragel/ragel-guide-6.10.pdf>) (PDF). March 2017.
- "Additional Target Languages Return to Ragel 7" (<http://www.colm.net/news/2018/05/18/new-target-langs.html>). 18 May 2018.
- Liqun Chen, Chris J. Mitchell, Andrew Martin (2009) *Trusted Computing: Second International Conference, Trust 2009 Oxford, UK, April 6–8, 2009, Proceedings*. p. 111
- Omar Badreddin (2010) "**Umple**: a model-oriented programming language." *Software Engineering, 2010 ACM/IEEE 32nd International Conference on*. Vol. 2. IEEE, 2010.
- Dr. Adrian D. Thurston (<http://www.complang.org/thurston/>) at *complang.org* Last changed: Jul 14, 2013
- "Ragel State Machine Compiler" (<http://www.colm.net/open-source/ragel/>). *www.colm.net*. Retrieved 2019-11-19.
- "Ragel User Guide" (<http://www.colm.net/files/ragel/ragel-guide-6.10.pdf>) (PDF). March 2017.

External links

- Media related to Ragel at Wikimedia Commons
- Official website (<https://www.colm.net/open-source/ragel/>)

Retrieved from "<https://en.wikipedia.org/w/index.php?title=Ragel&oldid=1061552273>"

This page was last edited on 22 December 2021, at 10:55 (UTC).

Text is available under the Creative Commons Attribution-ShareAlike License 3.0; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.