

naive-ebnf: L^AT_EX Package for EBNF in Plain Text*

Yegor Bugayenko
yegor256@gmail.com

2023-08-08, 0.0.14

NB! Large ENBF snippets may take too long to render!

1 Introduction

This package helps render an [Extended Backus-Naur Form](#) using plain text notation:

$\begin{array}{l} \langle \lambda\text{-Expr} \rangle \rightarrow \langle \text{Var} \rangle \\ \quad \text{ "}\lambda\text{" } \langle \text{Var} \rangle \text{ "}. " \langle \text{Expr} \rangle \\ \quad \text{ "(" } \langle \text{Expr} \rangle \langle \text{Expr} \rangle \text{ ")" } \end{array}$	<pre>1 \documentclass{minimal} 2 \usepackage{naive-ebnf} 3 \usepackage{mathtools} 4 \begin{document} 5 \begin{ebnf} 6 <\$\lambda\$-Expr> := <Var> \\ 7 "\$\lambda\$" <Var> "." <Expr> \\ 8 "\char{'\" <Expr> <Expr> "\char{'\" 9 \end{ebnf} 10 \end{document}</pre>
---	---

ebnf The `ebnf` environment *doesn't* add any formatting to the paragraph, but only replaces the plain text symbols, such as “:=” and “<Var>” with proper L^AT_EX commands. The following syntax is understood inside the `ebnf` environment:

- := separates the left-hand side from the right-hand side of the production rule;
- < . . . > denotes a non-terminal (variable);
- " . . . " denotes a terminal symbol;
- ' . . . ' denotes a special non-printable terminal symbol, like 'EOL';
- (. . . | . . .) denotes a series of options to choose from;
- / . . . / denotes a regular expression, like / [a-z] + /;
- [. . .] denotes an optional substitution;
- { . . . } denotes a zero or more times repetition;
- { . . . }+ denotes one or more times repetition;

*The sources are in GitHub at [yegor256/naive-ebnf](https://github.com/yegor256/naive-ebnf)

- `||` denotes an indented vertical bar at the beginning of the string.

Attention: The usage of some symbols is prohibited inside terminals. Instead, the following substitutions are recommended:

- `$(\lparen$` and `$(\rparen$` instead of “(” and “)” (from the [mathtools](#) package);
- `$(\langle$` and `$(\rangle$` instead of “<” and “>”;
- `$(\lbrace$` and `$(\rbrace$` instead of “{” and “}” (also [mathtools](#));
- `$(\lbrack$` and `$(\rbrack$` instead of “[” and “]” (also [mathtools](#));
- `$(\vert$` instead of “|”.

They would look even better, if the following notation is used:

- `\char‘\ (` and `\char‘\)` instead of “(” and “)”;
- `\char‘\ <` and `\char‘\ >` instead of “<” and “>”;
- `\char‘\ {` and `\char‘\ }` instead of “{” and “}”;
- `\char‘\ [` and `\char‘\]` instead of “[” and “]”.

width There is an optional argument of `ebnf` environment, which sets the width of the left-hand side of each rule (the default width is `6em`):

This EBNF has a larger width of the left hand side than usual: $\langle \text{VeryLongVariable} \rangle \rightarrow \langle X \rangle \mid \langle Y \rangle$ $\langle X \rangle \rightarrow \text{"X"} \text{ EOL}$ $\langle Y \rangle \rightarrow \text{"Y"}$	<pre> 4 This EBNF has a larger width of \ 5 the left hand side than usual: \par 6 \begin{ebnf}[1.5in] 7 <VeryLongVariable> := <X> <Y> \ 8 <X> := "X" 'EOL' \ 9 <Y> := "Y" \ 10 \end{ebnf} </pre>
--	---

\EbnfTerminal Inside the text, terminals, non-terminals, and special terminals may be formatted using three supplementary commands:
\EbnfNonTerminal
\EbnfSpecial

The non-terminal $\langle \text{Var} \rangle$ in λ -calculus may be equal to v_1, v_2, \dots . Application starts with “(” and ends with “)”.	<pre> 6 The non-terminal \EbnfNonTerminal{Var} 7 in \$(\lambda\$-calculus may be equal 8 to \$v_1, v_2, \dots\$. Application 9 starts with \EbnfTerminal{ (} and ends 10 with \EbnfTerminal{) }. </pre>
---	--

It’s possible to use them in math-mode too, for example:

If “($f_1 \langle \lambda\text{-Var} \rangle$)” is always true, then f_1 is a tautology.	<pre> 6 If \$(\EbnfTerminal{ (} f_1 7 \EbnfNonTerminal{ \$(\lambda\$-Var} 8 \EbnfTerminal{) }\$ is always true, then 9 \$f_1\$ is a tautology. </pre>
--	---

\EbnfRegex A regular expression is possible too:

$\langle \text{data} \rangle \rightarrow \langle \text{bool} \rangle \mid \langle \text{integer} \rangle \mid \langle \text{byte} \rangle$ $\langle \text{bool} \rangle \rightarrow \text{TRUE} \mid \text{FALSE}$ $\langle \text{integer} \rangle \rightarrow / (+ -)?[0-9]+/$ $\langle \text{byte} \rangle \rightarrow /[0-9a-f]{2}/$ $\langle \text{number} \rangle \rightarrow /[1-9]+/ \ / [0-9]+/$	<pre> 6 \begin{ebnf} 7 <data> := <bool> <integer> <byte> \\ 8 <bool> := "TRUE" "FALSE" \\ 9 <integer> := /(+\char'\'-)?[0-9]+/ \\ 10 <byte> := /[0-9a-f]\char'\'{2}\char'\'/ \\ 11 <number> := /[1-9]+/ \ / [0-9]+/ 12 \end{ebnf} </pre>
--	--

Special symbols are interpreted correctly, if they stay inside quotes:

$\langle X \rangle \rightarrow \text{EOL} \text{ " ' " }$ $\langle Y \rangle \rightarrow ">" "<" "[" "]" " / " " / " "$ $\langle Z \rangle \rightarrow \text{\LaTeX} \text{ "\textdollar" }$	<pre> 5 \begin{ebnf} 6 <X> := 'EOL' " ' " " \\ 7 <Y> := ">" "<" "[" "]" " / " " / " " \\ 8 <Z> := "\LaTeX" "\textdollar" \\ 9 \end{ebnf} </pre>
---	--

Nested brackets work fine too:

$\langle x \rangle \rightarrow ("x" ("y" ("z" \langle z \rangle)))$ $\langle y \rangle \rightarrow [["x1"] \{ / [a-z]+ / \}]$ $\langle z \rangle \rightarrow \{ \{ \{ \langle x \rangle \} \langle y \rangle \}^+ \langle z \rangle \}$ $\langle t \rangle \rightarrow [\langle x \rangle] [\langle y \rangle]$	<pre> 5 \begin{ebnf} 6 % There is no meaning in this: 7 <x> := ("x" ("y" ("z" <z>))) \\ 8 <y> := [["x1"] { / [a-z]+ / }] \\ 9 <z> := { { { <x> }+ <y> } <z> } \\ 10 <t> := [<x>] [<y>] \\ 11 \end{ebnf} </pre>
--	--

2 Package Options

It's possible to configure the behavior of the package with the help of a few package options:

bw By default, some colors are used in the rendered grammar. However, the **bw** package option disables any colors and makes sure the gammar is black-and-white:

```
\usepackage[bw]{naive-ebnf}
```

trail The **ebnf** environment is doing pre-processing of the \TeX commands provided and then let \TeX render them. It may be useful to see the output generated by the pre-processing. The **trail** option (with a file name) asks the package to save the content of the environment after the pre-processing into the file:

```
\usepackage[trail=log.tex]{naive-ebnf}
```

3 Implementation

First, we process package options:

```

1 \RequirePackage{pgfopts}
2 \pgfkeys{
3   /ebnf/.cd,
4   bw/.store in=\ebnf@bw,
5   trail/.store in=\ebnf@trail,

```

```

6  trail/.default=naive-ebnf.tmp.tex,
7 }
8 \ProcessPgfPackageOptions{/ebnf}

```

Then, we include a few packages, mostly to deal with L^AT_EX3 expressions:

```

9 \RequirePackage{expl3}

```

`\ebnf@color` Then, we include `xcolor` to colorize the output a bit:

```

10 \makeatletter\ifdefined\ebnf@bw\else
11   \RequirePackage{xcolor}
12 \fi
13 \newcommand\ebnf@color[2]
14   {\ifdefined\ebnf@bw#2\else\textcolor{#1}{#2}\fi}
15 \makeatother

```

`\EbnfTerminal` Then, we define a command to render a single terminal:

```

16 \makeatletter
17 \newcommand\EbnfTerminal[1]{\%
18   \relax\ifmmode\else\ttfamily\fi%
19   \ebnf@color{gray}{\relax\ifmmode\textsf{''}\else\sffamily{''}\fi}%
20   #1%
21   \ebnf@color{gray}{\relax\ifmmode\textsf{''}\else\sffamily{''}\fi}}
22 \makeatother

```

`\EbnfTerminal` Then, we define a command to render a single non-terminal:

```

23 \makeatletter
24 \newcommand\EbnfNonTerminal[1]{\%
25   \ebnf@color{gray}{\relax\ifmmode\langle\else\(\langle\)\fi}%
26   \relax\ifmmode\textsf{#1}\else\sffamily#1\fi%
27   \ebnf@color{gray}{\relax\ifmmode\rangle\else\(\rangle\)\fi}}
28 \makeatother

```

`\EbnfSpecial` Then, we define a command to render a single non-terminal:

```

29 \makeatletter
30 \newcommand\EbnfSpecial[1]{\relax\ifmmode\else\ttfamily\fi#1}%
31 \makeatother

```

`\EbnfRegex` Then, we define a command to render a regular expression:

```

32 \makeatletter
33 \newcommand\EbnfRegex[1]{\relax\ifmmode\else\ttfamily\fi/#1/}%
34 \makeatother

```

Then, we define supplementary commands:

```

35 \makeatletter
36 \newcommand\ebnf@optional[1]
37   {\ebnf@color{gray}{[ ]#1\ebnf@color{gray}{ ]}}
38 \newcommand\ebnf@repetition[1]
39   {\ebnf@color{gray}{\{ \}#1\ebnf@color{gray}{ \}}}
40 \newcommand\ebnf@iteration[1]
41   {\ebnf@color{gray}{\{ \}#1\ebnf@color{gray}{ \}}\scriptscriptstyle +)}
42 \newcommand\ebnf@grouping[1]
43   {\ebnf@color{gray}{( )#1\ebnf@color{gray}{ )}}
44 \ExplSyntaxOn
45 \newcommand\ebnf@terminal[1]{

```

```

46 \tl_set:Nn \l_ebnf_tl {}
47 \tl_set_rescan:Nnn \l_ebnf_tl {} { #1 }
48 \EbnfTerminal{\l_ebnf_tl}
49 }
50 \newcommand\ebnf@special[1]{
51 \tl_set:Nn \l_ebnf_tl {}
52 \tl_set_rescan:Nnn \l_ebnf_tl {} { #1 }
53 \EbnfSpecial{\l_ebnf_tl}
54 }
55 \newcommand\ebnf@nonterminal[1]{
56 \tl_set:Nn \l_ebnf_tl {}
57 \tl_set_rescan:Nnn \l_ebnf_tl {} { #1 }
58 \EbnfNonTerminal{\l_ebnf_tl}
59 }
60 \newcommand\ebnf@regex[1]{
61 \tl_set:Nn \l_ebnf_tl {}
62 \tl_set_rescan:Nnn \l_ebnf_tl {} { #1 }
63 \EbnfRegex{\l_ebnf_tl}
64 }
65 \ExplSyntaxOff
66 \newcommand\ebnf@to
67 {\ebnf@color{gray}{\(\to\)}}
68 \newcommand\ebnf@alternation
69 {\ebnf@color{gray}{\(\vert\)}}
70 \makeatother

```

ebnf Then, we define the ebnf environment:

```

71 \ExplSyntaxOn
72 \cs_generate_variant:Nn \tl_replace_all:Nnn {Nx}
73 \makeatletter
74 \NewDocumentEnvironment{ebnf}{0{4em}+b}
75 {\tl_set:Nn\ebnf_tmp{#2}}
76 {%
77 \regex_replace_all:nnN
78 { ([^s]) / ([^s]) } {\1\slash{} \2} \ebnf_tmp%
79 \regex_replace_all:nnN
80 { ([^s]) < } {\1\textless{} } \ebnf_tmp%
81 \regex_replace_all:nnN
82 { > ([^s]) } {\1\textgreater{} } \ebnf_tmp%
83 \regex_replace_all:nnN
84 { ([^s]) ' ([^s]) } {\1\textquotesingle{} \2} \ebnf_tmp%
85 \regex_replace_all:nnN
86 { ([^s]) \ | ([^s]) } {\1\textbar{} \2} \ebnf_tmp%
87 %
88 \regex_replace_all:nnN
89 { / (.+?) / }%
90 {\c{ebnf@regex}{\1}} \ebnf_tmp%
91 \cs_new:Npn\ebnf_iterated{%
92 \regex_replace_all:nnNT
93 { \{s([^\s]*([^\s\{\}\|\s\(\)\|\{)[^\s])?\)*\s\}+ }%
94 {\c{ebnf@iteration}{\1}} \ebnf_tmp \ebnf_iterated}%
95 \ebnf_iterated%
96 \cs_new:Npn\ebnf_curled{%
97 \regex_replace_all:nnNT

```

```

98   { \[ \s((([^\s]*(\s[^\]\[\]| \s(\)|\{)[^\s]))?)*)\s\} }%
99   {\c{ebnf@repetition}{\1}} \ebnf_tmp \ebnf_curled}%
100 \ebnf_curled%
101 \cs_new:Npn\ebnf_brackets{%
102   \regex_replace_all:nnNT
103   { \[ \s((([^\s]*(\s[^\]\[\]| \s(\)|\{)[^\s]))?)*)\s\} }%
104   {\c{ebnf@grouping}{\1}} \ebnf_tmp \ebnf_brackets}%
105 \ebnf_brackets%
106 \cs_new:Npn\ebnf_squares{%
107   \regex_replace_all:nnNT
108   { \[ \s((([^\s]*(\s[^\]\[\]| \s(\)|\{)[^\s]))?)*)\s\] }%
109   {\c{ebnf@optional}{\1}} \ebnf_tmp \ebnf_squares}%
110 \ebnf_squares%
111 \regex_replace_all:nnN { (<[^\>]+>\s:=) }%
112   {\c{makebox}[#1][r]{\1}} \ebnf_tmp%
113 \regex_replace_all:nnN { <(.+?)> }%
114   {\c{ebnf@nonterminal}{\1}} \ebnf_tmp%
115 \regex_replace_all:nnN { "(.+?)" }%
116   {\c{ebnf@terminal}{\1}} \ebnf_tmp%
117 \regex_replace_all:nnN { '(.+?)' }%
118   {\c{ebnf@special}{\1}} \ebnf_tmp%
119 \regex_replace_all:nnN { \|(\|) }%
120   {\c{makebox}[#1][r]{ \| }} \ebnf_tmp%
121 \regex_replace_all:nnN { \| }%
122   {\c{ebnf@alternation}{}} \ebnf_tmp%
123 \regex_replace_all:nnN { := }%
124   {\c{ebnf@to}{}} \ebnf_tmp%
125 \tl_put_left:Nn \ebnf_tmp {\noindent}
126 \tl_put_right:Nn \ebnf_tmp {}
127 \ifdefined\ebnf@trail%
128   \newwrite\ebnf@write%
129   \immediate\openout\ebnf@write\ebnf@trail\relax%
130   \immediate\write\ebnf@write{\unexpanded\expandafter{\ebnf_tmp}}%
131   \immediate\closeout\ebnf@write%
132   \message{naive-ebnf:\space pre-processed\space TeX
133     \space saved\space to\space "\ebnf@trail"^^J}%
134 \fi%
135 \ebnf_tmp}
136 \makeatother
137 \ExplSyntaxOff

138 \endinput

```

Change History

0.0.1		text and math modes.	4	
	General: First draft.	3	0.0.4	
0.0.11	<code>ebnf</code> : Many bugs fixed in the area of regular expression matching.	5	<code>ebnf</code> : Any symbols are allowed inside <code>\EbnfNonTerminal</code> commands and inside the <code>ebnf</code> environment, where non-terminals are mentioned.	5
0.0.14	<code>ebnf</code> : One-or-more repetition introduced with <code>{...}+</code> syntax.	5	0.0.5	
0.0.2	General: Proper parsing of grouping.	3	General: New package option <code>trail</code> added, to enable saving of the generated <code>T_EX</code> content to a file, for debugging purposes.	3
	Substitutions suggested for special symbols.	3	0.0.6	
	<code>\EbnfTerminal</code> : New command <code>\EbnfNonTerminal</code> added, to enable rendering non-terminal symbols outside of the <code>ebnf</code> environment.	4	<code>\EbnfSpecial</code> : New command <code>\EbnfSpecial</code> added, to enable rendering of special non-printable terminal symbols outside of the <code>ebnf</code> environment.	4
	New command <code>\EbnfTerminal</code> added, to enable rendering terminal symbols outside of the <code>ebnf</code> environment.	4	0.0.8	
0.0.3	<code>\EbnfTerminal</code> : Quotes fixed in both		<code>\EbnfRegex</code> : New command <code>\EbnfRegex</code> added, to enable rendering of regular expressions outside of the <code>ebnf</code> environment.	4

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols		\ebnf@terminal 45	\noindent 125
\(. . . 25, 27, 41, 67, 69, 103		\ebnf@to 66	
\) . . . 25, 27, 41, 67, 69, 103		\ebnf@trail 5, 127, 129, 133	O
\+ 93		\ebnf@write	\openout 129
\[. 108		. . . 128, 129, 130, 131	
\{ 39, 41, 93, 98		\EbnfNonTerminal . 24, 58	P
\} 39, 41, 93, 98		\EbnfRegex 32, 63	\pgfkeys 2
\] 108		\EbnfSpecial 29, 53	\ProcessPgfPackageOptions 8
\ 86, 119, 121		\EbnfTerminal . . 16, 23, 48	
		\endinput 138	
Numbers		\expandafter 130	R
\2 78, 84, 86		\ExplSyntaxOff . . 65, 137	\rangle 27
		\ExplSyntaxOn 44, 71	\regex 77, 79, 81,
			83, 85, 88, 92, 97,
C			102, 107, 111, 113,
\c 90, 94, 99,		I	115, 117, 119, 121, 123
104, 109, 112, 114,		\ifdefined . . . 10, 14, 127	\relax 18, 19, 21,
116, 118, 120, 122, 124		\ifmmode 18, 19,	25, 26, 27, 30, 33, 129
\closeout 131		21, 25, 26, 27, 30, 33	\RequirePackage . 1, 9, 11
\cs 72, 91, 96, 101, 106		\immediate . 129, 130, 131	
			S
E			\scriptscriptstyle . 41
\ebnf . . . 71, 75, 78, 80,		L	\sffamily 19, 21, 26
82, 84, 86, 90, 91,		\l . 46, 47, 48, 51, 52, 53,	\space 132, 133
94, 95, 96, 99, 100,		56, 57, 58, 61, 62, 63	
101, 104, 105, 106,		\langle 25	
109, 110, 112, 114,			T
116, 118, 120, 122,		M	\textcolor 14
124, 125, 126, 130, 135		\makeatletter . . . 10,	\textsf 19, 21, 26
\ebnf@alternation . . 68		16, 23, 29, 32, 35, 73	\tl . 46, 47, 51, 52, 56, 57,
\ebnf@bw 4, 10, 14		\makeatother . . . 15,	61, 62, 72, 75, 125, 126
\ebnf@color		22, 28, 31, 34, 70, 136	\to 67
10, 19, 21, 25, 27,		\message 132	\ttfamily 18, 30, 33
37, 39, 41, 43, 67, 69			
\ebnf@grouping 42		N	U
\ebnf@iteration 40		\newcommand	\unexpanded 130
\ebnf@nonterminal . . 55		. . . 13, 17, 24, 30,	
\ebnf@optional 36		33, 36, 38, 40, 42,	V
\ebnf@regexp 60		45, 50, 55, 60, 66, 68	\vert 69
\ebnf@repetition . . . 38		\NewDocumentEnvironment	
\ebnf@special 50	 74	W
		\newwrite 128	\write 130