



eolang: L^AT_EX Package for Formulas and Graphs of EO Programming Language and φ -calculus*

Yegor Bugayenko
yegor256@gmail.com

2022-10-27, 0.1.0

NB! You must run T_EX processor with `--shell-escape` option and you must have [Perl](#) installed. This package doesn't work on Windows.

1 Introduction

This package helps you print formulas of φ -calculus, which is a formal foundation of [EO](#) programming language. The calculus was introduced by Bugayenko (2021) and later formalized by Kudasov et al. (2022). Here is how you render a simple expression:

$ \begin{aligned} a &\mapsto \llbracket \\ &\quad \rho \# \mapsto \xi.b, \\ &\quad b \mapsto \llbracket c \mapsto \text{fn}(56), \\ &\quad \quad \varphi \mapsto \text{hello}(\xi), \\ &\quad \Delta \mapsto 01\text{-FE-C3} \rrbracket \rrbracket, \\ x &\mapsto \llbracket \alpha_0 \mapsto \emptyset \rrbracket. \end{aligned} $	<pre> 1 \documentclass{article} 2 \pagestyle{empty} 3 \usepackage{eolang} 4 \begin{document} 5 \begin{phiquestion*} 6 a -> [[7 ^ !-> \$.b, 8 b -> [[c -> fn (56), 9 @ -> hello (\$), 10 \Delta ..> 01-FE-C3]]],\ 11 x -> [[\alpha_0 -> ?]]. 12 \end{phiquestion*} 13 \end{document} </pre>
--	--

`phiquestion (env)` The environment `phiquestion` lets you write a φ -calculus expressions using simple plain-text notation, where:

*The sources are in GitHub at [objectionary/eolang.sty](https://github.com/objectionary/eolang.sty)

- “@” maps to “ φ ” (`\varphi`),
- “^” maps to “ ρ ” (`\rho`),
- “\$” maps to “ ξ ” (`\xi`),
- “&” maps to “ σ ” (`\sigma`),
- “?” maps to “ \emptyset ” (`\varnothing`),
- “->” maps to “ \mapsto ” (`\mapsto`),
- “!->” maps to “ \vdash ” (`\vdash`),
- “..>” maps to “ \vdash ” (`\vdash`),
- “[[” maps to “ \llbracket ” (`\llbracket`),
- “]]” maps to “ \rrbracket ” (`\rrbracket`),
- “|abc|” maps to “abc” (`\texttt{abc}`).

Also, a few symbols are supported for φ PU architecture:

- “-abc>” maps to “ \xrightarrow{ABC} ” (`\xrightarrow{\text{\sf family\scshape abc}}`),
- “:=” maps to “ \models ” (`\models`).

`\phiiq` The command `\phiiq` lets you inline a φ -calculus expressions using the same simple plain-text notation:

<p>A simple object $x \mapsto \llbracket \varphi \mapsto y \rrbracket$ is a decorator of the data object $y \mapsto \llbracket \Delta \vdash 42 \rrbracket$.</p>	<pre> 1 \documentclass{article} 2 \pagestyle{empty} 3 \usepackage{eolang} 4 \begin{document} 5 A simple object \\\ 6 \phiiq{x -> [[@ -> y]]} \\\ 7 is a decorator of \\\ 8 the data object \\\ 9 \phiiq{y -> [[\Delta ..> 42]]}. 10 \end{document} </pre>
--	---

`sodg (env.)` The environment `sodg` allows you to draw a [SODG](#) graph:

	<pre> 1 \documentclass{article} 2 \pagestyle{empty} 3 \usepackage{eolang} 4 \begin{document} 5 \begin{sodg} 6 v0 7 v1 xy:v0,-2,+1 data: 42 8 v0->v1 a:\$x\$ rho 9 v2 xy:v0,+1,+1 atom:\$\xi.x+1\$ 10 v1->v2 a: hi bend:-15 11 v2->v0 pi bend:10 12 \end{sodg} 13 \end{document} </pre>
--	---

The content of the environment is parsed line by line. Markers in each line are separated by a single space. The first marker is either a unique name of a vertex, like

v_1 in the example above, or an edge, like $v_0 \rightarrow v_1$. All other markers are either unary like ρ or binary like $\text{atom}:\$ \backslash x_i . x+1 \$$. Binary markers have two parts, separated by colon. The following markers are supported for a vertex:

- “ $\text{data}:[\langle \text{box} \rangle]$ ” makes it a data vertex with an optional attached $\langle \text{box} \rangle$,
- “ $\text{atom}:[\langle \text{box} \rangle]$ ” makes it an atom with an optional attached $\langle \text{box} \rangle$,
- “ $\text{box}:\langle \text{txt} \rangle$ ” attaches a $\langle \text{box} \rangle$ to it,
- “ $\text{xy}:\langle v \rangle, \langle r \rangle, \langle d \rangle$ ” places this vertex in a position relative to the vertex $\langle v \rangle$, shifting it right by $\langle r \rangle$ and down by $\langle d \rangle$ centimetres.

The following markers are supported for an edge:

- “ ρ ” places a backward snake arrow to the edge,
- “ \rrho ” places a reverse ρ ,
- “ $\text{bend}:\langle \text{angle} \rangle$ ” bend it right by the amount of $\langle \text{angle} \rangle$,
- “ $\text{a}:\langle \text{txt} \rangle$ ” attaches label $\langle \text{txt} \rangle$ to it,
- “ pi ” makes it dotted, with π label.

`\eolang` There is also a no-argument command `\eolang` to help you print the name of EO
`\phic` language. It understands anonymous mode of [acmart](#) and prints itself differently, to double-blind your paper. There is also `\phic` command to print the name of φ -calculus, also sensitive to anonymous mode.

In our research we use XYZ,
an experimental object-oriented
dataflow language, and α -calculus,
its formal foundation.

```

1 \documentclass[anonymous]{acmart}
2 \thispagestyle{empty}
3 \usepackage{eolang}
4 \begin{document}
5 In our research we use \eolang{, \\\
6 an experimental object-oriented \\\
7 dataflow language, and \phic{, \\\
8 its formal foundation.
9 \end{document}

```

`\phiConst` A simple commands is defined to help you render an arrow for a constant attribute. It is recommende not to use it directly, but use `!->` instead. However, if you want to use `\phiConst`, wrap it in `\mathrel` for better display:

$\llbracket x \mapsto y \rrbracket$

```

1 \documentclass{article}
2 \pagestyle{empty}
3 \usepackage{eolang}
4 \begin{document}
5 \phiq{\llbracket x \mathrel{\phiConst} y \rrbracket}
6 \end{document}

```

2 Package Options

`tmpdir` The default location of temp files is `_eolang`. You can change this using `tmpdir` option:

```
\usepackage[tmpdir=/tmp/foo]{eolang}
```

3 More Examples

The `phiquation` environment treats ends of line as signals to start new lines in the formula. If you don't want this to happen and want to parse the next line as the a continuation of the current line, you can use a single backslash as it's done here:

$\frac{x \mapsto [\varphi \mapsto y] \quad y \mapsto [z \mapsto 42]}{x.z \mapsto 42} R1$	<pre> 1 \documentclass{article} 2 \usepackage{amsmath} 3 \usepackage{eolang} 4 \pagestyle{empty} 5 \begin{document} 6 \begin{phiquation*} 7 \dfrac \{ 8 x->[[@->y]] \quad y->[[z-> 42]] \} \backslash 9 \{x.z -> 42 \} \backslash 10 \text{\sffamily R1} 11 \end{phiquation*} 12 \end{document}</pre>
--	--

The `phiquation` environment may be used together with [acmart](#):

$\begin{array}{l} x \mapsto [\\ \quad y \mapsto [\\ \quad \quad z \mapsto \xi, f \mapsto \emptyset]], \\ \beta_1 \models [\psi \xrightarrow{\text{WAIT}} \emptyset]. \end{array}$	<pre> 1 \documentclass{acmart} 2 \usepackage{eolang} 3 \thispagestyle{empty} 4 \begin{document} 5 \begin{phiquation*} 6 x -> [[7 y -> [[8 z !-> \$, f ..> ?]]], \backslash 9 \beta_1 := [\psi -wait> ?]. 10 \end{phiquation*} 11 \end{document}</pre>
---	---

The `phiquation` environment will automatically align formulas by the first arrow, if there are only left-aligned formulas:

$\begin{array}{l} x(\pi) \mapsto [\lambda \mapsto f_1], \\ x(a, b, c) \mapsto [\alpha_0 \mapsto \emptyset, \varphi \mapsto \text{hello}(\xi)], \\ \Delta = 43-09. \end{array}$	<pre> 1 \documentclass{acmart} 2 \usepackage{eolang} 3 \thispagestyle{empty} 4 \begin{document} 5 \begin{phiquation*} 6 x(\pi) -> [[\lambda ..> f_1]], \backslash 7 x(a,b,c) -> [[\alpha_0 -> ?, \backslash 8 @ -> hello (\$)]], \backslash 9 \Delta = 43-09 . 10 \end{phiquation*} 11 \end{document}</pre>
--	--

4 Implementation

First, we include a few packages. We need [stmaryrd](#) for `\llbracket` and `\rrbracket` commands:

```
1 \RequirePackage{stmaryrd}
```

We need [amsmath](#) for `equation*` environment:

```
2 \RequirePackage{amsmath}
```

We need [amssymb](#) for `\varnothing` command. We disable `\Bbbk` because it may conflict with some packages from [acmart](#):

```
3 \let\Bbbk\relax\RequirePackage{amssymb}
```

We need [fancyvrb](#) for `\VerbatimEnvironment` command:

```
4 \RequirePackage{fancyvrb}
```

We need [iexec](#) for executing Perl scripts:

```
5 \RequirePackage{iexec}
```

Then, we process package options:

```
6 \RequirePackage{pgfopts}
7 \RequirePackage{ifluatex}
8 \RequirePackage{ifxetex}
9 \pgfkeys{
10  /eolang/.cd,
11  tmpdir/.store in=\eolang@tmpdir,
12  tmpdir/.default=_eolang\ifxetex-xe\else\ifluatex-lua\fi\fi,
13  tmpdir
14 }
15 \ProcessPgfOptions{/eolang}
```

Then, we make a directory where all temporary files will be kept:

```
16 \iexec[null]{mkdir -p "\eolang@tmpdir/\jobname"%}
```

`\eolang@mdfive` Then, we define a command for MD5 hash calculating of a file:

```
17 \RequirePackage{pdftexcmds}
18 \makeatletter\newcommand\eolang@mdfive[1]{\pdf@filemdfivesum{#1}}\makeatother
```

`eolang-phi.pl` Then, we create a Perl script for phi-quation processing:

```
19 \makeatletter
20 \begin{VerbatimOut}{\eolang@tmpdir/eolang-phi.pl}
21 $env = $ARGV[0];
22 open(my $fh, '<', $ARGV[1]);
23 my $tex; { local $/; $tex = <$fh>; }
24 print '% This file is auto-generated', "\n";
25 print '% There are ', length($tex),
26   ' chars in the input: ', $ARGV[1], "\n";
27 print '% ---', "\n";
28 if (index($tex, "\t") > 0) {
29   print "TABS are prohibited!";
30   exit 1;
31 }
32 my @lines = split (/\\n/g, $tex);
33 foreach my $t (@lines) {
34   print '% ', $t, "\n";
35 }
```

```

36 print '% ---', "\n";
37 if ($env eq 'phiq') {
38   print '$';
39 } else {
40   print '\begin{', $env, '}\begin{split}';
41 }
42 $tex =~ s/^\s+|\s+$//g;
43 if ($env ne 'phiq') {
44   $tex =~ s/\s+\\\n\s*//g;
45   $tex =~ s/\\\n\\\n\\\n/g;
46 }
47 $tex =~ s/\?/\varnothing{/g;
48 $tex =~ s/@/\varphi{/g;
49 $tex =~ s/&/\sigma{/g;
50 $tex =~ s/\^/\rho{/g;
51 $tex =~ s/\$/\xi{/g;
52 $tex =~ s/-([a-z]+)>/\mathrel{\xrightarrow{\text{\sfamily\scshape \1}}}/g;
53 $tex =~ s/!->/\mathrel{\phiConst}/g;
54 $tex =~ s/->/\mathrel{\mapsto}/g;
55 $tex =~ s/:=/\mathrel{\vDash}/g;
56 $tex =~ s/..>/\mathrel{\phiDotted}/g;
57 $tex =~ s/|([^\|]+)|/\texttt{\1}/g;
58 $tex =~ s/\[/\llbracket\mathrel{/g;
59 $tex =~ s/\]/\rrbracket\mathrel{/g;
60 if ($env ne 'phiq') {
61   $tex =~ s/\n\n/\\\&/g;
62   $tex =~ s/\n/\\\[-4pt]\&/g;
63   $tex =~ s/([^\s])\s{2}([^\s])/1 \2/g;
64   $tex =~ s/\s{2}/ \quad{/g;
65   my @leads = $tex =~ /&([^\s])\s/g;
66   my @eols = $tex =~ /\&/g;
67   $tex = '&' . $tex;
68   if (0+@leads == 0+@eols && 0+@eols > 0) {
69     $tex =~ s/&([^\s])\s/1\&/g;
70   }
71 }
72 print $tex;
73 if ($env eq 'phiq') {
74   print '$';
75 } else {
76   print '\end{split}\end{', $env, '}';
77 }
78 print '\endinput', "\n";
79 \end{VerbatimOut}
80 \message{eolang: File with Perl script
81   '\eolang@tmpdir/eolang-phi.pl' saved^^J}%
82 \iexec[trace,null]{perl -pi -e 's/(\\[a-zA-Z])\\s+/\1/g'
83   "\eolang@tmpdir/eolang-phi.pl"}
84 \makeatother

```

phiquation Then, we define phiquation and phiquation* environments through a supplementary `\eolang@process` command:

```

85 \makeatletter\newcommand\eolang@process[1]{
86   \def\hash{\eolang@mdfive

```

```

87   {\eolang@tmpdir/\jobname/phiuation.tex}}%
88   \iexec[null]{cp "\eolang@tmpdir/\jobname/phiuation.tex"
89     "\eolang@tmpdir/\jobname/\hash.tex"}%
90   \iexec[trace,stdout=\eolang@tmpdir/\jobname/\hash-post.tex]{
91     perl "\eolang@tmpdir/eolang-phi.pl"
92     '#1'
93     "\eolang@tmpdir/\jobname/\hash.tex"}%
94 }
95 \newenvironment{phiuation*}%
96 {\VerbatimEnvironment\begin{VerbatimOut}
97   {\eolang@tmpdir/\jobname/phiuation.tex}}
98 {\end{VerbatimOut}\eolang@process{equation*}}
99 \newenvironment{phiuation}%
100 {\VerbatimEnvironment\begin{VerbatimOut}
101   {\eolang@tmpdir/\jobname/phiuation.tex}}
102 {\end{VerbatimOut}\eolang@process{equation}}
103 \makeatother

```

\phiq Then, we define \phiq command:

```

104 \makeatletter\newcommand\phiq[1]{
105   \iexec[trace,quiet,stdout=\eolang@tmpdir/\jobname/phiq.tex]{
106     /bin/echo '\detokenize{#1}'}
107   \def\hash{\eolang@mdfive
108     {\eolang@tmpdir/\jobname/phiq.tex}}%
109   \iexec[null]{cp "\eolang@tmpdir/\jobname/phiq.tex"
110     "\eolang@tmpdir/\jobname/\hash.tex"}%
111   \iexec[trace,stdout=\eolang@tmpdir/\jobname/\hash-post.tex]{
112     perl \eolang@tmpdir/eolang-phi.pl '\phiq'
113     "\eolang@tmpdir/\jobname/\hash.tex"}%
114 }\makeatother

```

eolang-sodg.pl Then, we create a Perl script for sodg graphs processing:

```

115 \makeatletter
116 \begin{VerbatimOut}{\eolang@tmpdir/eolang-sodg.pl}
117 open(my $fh, '<', $ARGV[0]);
118 my $tex; { local $/; $tex = <$fh>; }
119 print '% This file is auto-generated', "\n";
120 print '% There are ', length($tex),
121   ' chars in the input: ', $ARGV[0], "\n";
122 print '% ---', "\n";
123 if (index($tex, "\t") > 0) {
124   print "TABS are prohibited!";
125   exit 1;
126 }
127 $tex =~ s/^\s+|\s+$//g;
128 $tex =~ s/(\\[a-zA-Z]+)\s+/\\1/g;
129 $tex =~ s/\n\s+/\n/g;
130 $tex =~ s/\\|([^\|]+)\\|/\\texttt{\1}/g;
131 my @cmds = split (/\\n/g, $tex);
132 foreach my $t (@cmds) {
133   print '% ', $t, "\n";
134 }
135 print '% ---', "\n";
136 print '\begin{picture}', "\n";

```

```

137 foreach my $c (@cmds) {
138   my ($head, $tail) = split (/ /, $c, 2);
139   my %opts = {};
140   foreach my $p (split (/ /, $tail)) {
141     my ($q, $t) = split (/:/, $p);
142     $opts{$q} = $t;
143   }
144   if (index($head, '->') == -1) {
145     print '\node[';
146     if (exists $opts{'xy'}) {
147       my ($v, $right, $down) = split(/,/, $opts{'xy'});
148       print ',below right=';
149       print $down;
150       print 'cm and ';
151       print $right;
152       print 'cm of ';
153       print $v;
154     }
155     if (exists $opts{'data'}) {
156       print ',phi-data';
157       if (not $opts{'data'} eq '') {
158         $opts{'box'} = $opts{'data'};
159       }
160     } elsif (exists $opts{'atom'}) {
161       print ',phi-atom';
162       if (not $opts{'atom'} eq '') {
163         $opts{'box'} = $opts{'atom'};
164       }
165     } else {
166       print ',phi-object';
167     }
168     print ']';
169     print ' (', $head, ')';
170     print ' {';
171     if ($head eq 'v0') {
172       print '\Phi';
173     } else {
174       print 'v_', substr($head, 1);
175     }
176     print '$}';
177     if (exists $opts{'box'}) {
178       print ' node[phi-box] at (';
179       print $head, '.south east) {';
180       print $opts{'box'}, ')';
181     }
182   } else {
183     print '\draw[';
184     if (exists $opts{'pi'}) {
185       print ',phi-pi';
186       if (not exists $opts{'a'}) {
187         $opts{'a'} = '$\pi$';
188       }
189     }
190     print ']';

```



```

191 my ($from, $to) = split (/>/, $head);
192 print ' (' , $from, ') ';
193 if (exists $opts{'bend'}) {
194     print 'edge [bend right=', $opts{'bend'}, ']';
195 } else {
196     print '--';
197 }
198 if (exists $opts{'rho'} or exists $opts{'rrho'}) {
199     print ' pic[sloped,phi-rho]{parallel arrow={';
200     print '-' if not exists $opts{'rrho'};
201     print '0.3,-0.15}}';
202 }
203 if (exists $opts{'a'}) {
204     print ' node [phi-attr] {' , $opts{'a'}, '}';
205 }
206 print ' (' , $to, ')';
207 }
208 print ";\n";
209 }
210 print '\end{phicture}', "\n", '\endinput';
211 \end{VerbatimOut}
212 \message{eolang: File with Perl script
213 '\eolang@tmpdir/eolang-sodg.pl' saved^^J}%
214 \iexec[trace,null]{perl -pi -e 's/(\\[a-zA-Z])\\s+/\1/g'
215 "\eolang@tmpdir/eolang-sodg.pl"}
216 \makeatother

```

tikz Then, we include tikz package and its libraries:

```

217 \RequirePackage{tikz}
218 \usetikzlibrary{arrows}
219 \usetikzlibrary{shapes}
220 \usetikzlibrary{decorations}
221 \usetikzlibrary{decorations.pathmorphing}
222 \usetikzlibrary{intersections}
223 \usetikzlibrary{positioning}
224 \usetikzlibrary{calc}
225 \usetikzlibrary{shapes.arrows}

```

phicture Then, we define internal environment phicture:

```

226 \newenvironment{phicture}%
227 {\noindent\begin{tikzpicture}[
228   ->,>=stealth',node distance=0,thick,
229   pics/parallel arrow/.style={
230     code={\draw[-latex,phi-rho] (##1) -- (-##1);}}}%
231 {\end{tikzpicture}}
232 \tikzstyle{transforms} = [fill=white!80!black, single arrow,
233   minimum height=0.5cm, minimum width=0.5cm,
234   single arrow head extend=2mm]
235 \tikzstyle{phi-thing} = [thick,inner sep=0pt,minimum height=2.4em,
236   draw,font={\small}]
237 \tikzstyle{phi-object} = [phi-thing,circle]
238 \tikzstyle{phi-data} = [phi-thing,regular polygon,
239   regular polygon sides=8]
240 \tikzstyle{phi-empty} = [phi-object]

```

```

241 \tikzstyle{phi-rho} = [draw,decorate,decoration={
242   snake,amplitude=.4mm,segment length=2mm,post length=1mm}]
243 \tikzstyle{phi-pi} = [draw,dotted]
244 \tikzstyle{phi-atom} = [phi-object,double]
245 \tikzstyle{phi-box} = [xshift=-5pt,yshift=3pt,draw,fill=white,
246   rectangle,thin,minimum width=1.2em,anchor=north west,
247   font={\scriptsize}]
248 \tikzstyle{phi-attr} = [midway,sloped,inner sep=0pt,
249   above=2pt,sloped/.append style={transform shape},
250   font={\scriptsize},color=black]

```

sodg Then, create a new environment sodg, as suggested [here](#):

```

251 \makeatletter\newenvironment{sodg}{%
252 {\VerbatimEnvironment\begin{VerbatimOut}
253 {\eolang@tmpdir/\jobname/sodg.tex}}
254 {\end{VerbatimOut}}%
255 \def\hash{\eolang@mdfive
256   {\eolang@tmpdir/\jobname/sodg.tex}}%
257 \iexec[null]{cp "\eolang@tmpdir/\jobname/sodg.tex"
258   "\eolang@tmpdir/\jobname/\hash.tex"}}%
259 \iexec[trace,stdout=\eolang@tmpdir/\jobname/\hash-post.tex]{
260   perl "\eolang@tmpdir/eolang-sodg.pl"
261   "\eolang@tmpdir/\jobname/\hash.tex"}}%
262 }\makeatother

```

\eolang

```

263 \makeatletter\newcommand\eolang{%
264   \ifdefined\anon%
265     \anon[XYZ]{\sffamily EO}%
266   \else%
267     {\sffamily EO}%
268   \fi%
269 }\makeatother

```

\phic

```

270 \makeatletter\newcommand\phic{%
271   \ifdefined\anon%
272     \anon[ $\alpha$ -calculus]{ $\varphi$ -calculus}%
273   \else%
274      $\varphi$ -calculus%
275   \fi%
276 }\makeatother

```

\phiConst Then, we define a command to render an arrow for a constant attribute, as suggested [here](#):

```

277 \newcommand\phiConst{%
278   \mathrel{\hspace{.15em}}\mapstochar\mathrel{\hspace{-.15em}}\mapsto

```

\phiDotted Then, we define a command to render an arrow for a special attribute, as suggested [here](#):

```

279 \RequirePackage{trimclip}
280 \RequirePackage{amsfonts}
281 \makeatletter
282 \newcommand{\phiDotted}{\mapstochar\mathrel{\mathpalette\phiDotted@\relax}}

```

```

283 \newcommand{\phiDotted@}[2]{%
284   \begingroup
285   \settowidth{\dimen\z@}{\${m@th#1\rightarrow$}%
286   \settoheight{\dimen\tw@}{\${m@th#1\rightarrow$}%
287   \sbox\z@{%
288     \makebox[\dimen\z@][s]{%
289       \clipbox{0 0 {0.4\width} 0}%
290       {\resizebox{\dimen\z@}{\height}%
291        {\${m@th#1\rightarrow$}}}%
292       \hss%
293       \clipbox{{0.69\width} {-0.1\height} 0 {-\height}}{\${m@th#1\rightarrow$}%
294       }%
295     }%
296   \ht\z@=\dimen\tw@ \dp\z@=\z@%
297   \box\z@%
298   \endgroup}\makeatother

```

References

- Bugayenko, Yegor (2021). *EOLANG and φ -calculus*. arXiv: [2111.13384](#) [cs.PL].
- Kudasov, Nikolai et al. (2022). *φ -calculus: a purely object-oriented calculus of decorated objects*. arXiv: [2204.07454](#) [cs.PL].

Change History

0.0.1	0.1.0
General: First draft.	General: Parsing of package options introduced.
0.0.2	
sodg: The environment “phigure” renamed to “sodg” for the sake of better semantic. The graph in the picture is solely a SODG graph, that’s why the name “sodg” is better.	\eolang: New command “eolang” added to print the name of the language in both normal and anonymous mode of “acmart” . . .
eolang-phi.pl: New symbol added for basket slots	\eolang@mdfive: New supplementary command added to calculate MD5 sum of a file.
Parsing of symbols “@,” “^,” and “&” enabled (varphi, rho, and sigma) . .	eolang-phi.pl: A new Perl script “eolang-phi.pl” added for parsing of phi expressions.
The symbols “[” and “]” replaced with “[[” and “]]” for abstract object brackets, because they conflicted with normal square brackets	eolang-sodg.pl: There are two Perl scripts now: one for phiuation, another one for sodg.
eolang-sodg.pl: The Perl file now has a fixed name, which doesn’t depend on the name of the TeX job. This file may be shared among jobs, no need to make it uniquely named. 7	\phic: New command “phic” prints the name of φ -calculus in both normal and anonymous mode of “acmart”
\phiq: Parsing of additional symbols enabled	\phiConst: New command “phiConst” added to denote a link to a constant attribute.
	\phiDotted: New command “phiDotted” added to denote a link to a special attribute.

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		H	\phic 270
\\$ 51		\hash 86, 89, 90,	\phiConst 277
\? 47		93, 107, 110, 111,	\picture 226
\[..... 58		113, 255, 258, 259, 261	\phiDotted 279
\] 59		\height 290, 293	\phiDotted@ 282, 283
\^ 50		\hspace 278	\phiiq 104
\ 57, 130		\hss 292	\phiquation 85
		\ht 296	\pi 187
Numbers			\ProcessPgfoptions .. 15
\2 63		I	
A		\iexec 16, 82, 88, 90, 105,	R
\alpha 272		109, 111, 214, 257, 259	\relax 3, 282
\anon ... 264, 265, 271, 272		\ifdefined 264, 271	\RequirePackage ...
B		\ifluatex 12 1, 2, 3, 4, 5,
\Bbbk 3		\ifxetex 12	6, 7, 8, 17, 217, 279, 280
\begin 20, 40, 96,			\resizebox 290
100, 116, 136, 227, 252		J	\rightarrow . 285, 286, 293
\box 297		\jobname 16, 87,	
C		88, 89, 90, 93, 97,	S
\clipbox 289, 293		101, 105, 108, 109,	\sbox 287
		110, 111, 113, 253,	\scriptsize 247, 250
D		256, 257, 258, 259, 261	\settoheight 286
\dashrightarrow ... 291			\settowidth 285
\def 86, 107, 255		M	\sffamily 265, 267
\detokenize 106		\m@th ... 285, 286, 291, 293	\small 236
\dimen 285, 286, 288, 290, 296		\makeatletter	\sodg 251
\dp 296		.. 18, 19, 85, 104,	
\draw 183, 230		115, 251, 263, 270, 281	T
E		\makeatother 216, 262, 269, 276, 298	\t 28, 123
\end 76, 79, 98,		\makebox 288	\tikz 217
102, 210, 211, 231, 254		\mapsto 278	\tikzstyle 232,
\endinput 78, 210		\mapstochar 278, 282	235, 237, 238, 240,
\eolang 263		\mathpalette 282	241, 243, 244, 245, 248
\eolang-phi.pl 19		\mathrel 278, 282	\tw@ 286, 296
\eolang-sodg.pl ... 115		\message 80, 212	
\eolang@mdfive		N	U
.... 17, 86, 107, 255		\newcommand 18, 85, 104,	\usetikzlibrary ...
\eolang@process 85, 98, 102		263, 270, 277, 282, 283	... 218, 219, 220,
\eolang@tmpdir		\newenvironment ...	221, 222, 223, 224, 225
.... 11, 16, 20,	 95, 99, 226, 251	V
81, 83, 87, 88, 89,		\node 145	\varphi 272, 274
90, 91, 93, 97, 101,		\noindent 227	\VerbatimEnvironment
105, 108, 109, 110,		 96, 100, 252
111, 112, 113, 116,		P	
213, 215, 253, 256,		\pdf@filemdfivesum .. 18	W
257, 258, 259, 260, 261		\pgfkeys 9	\width 289, 293
		\Phi 172	Z
			\z@ 285, 287, 288, 290, 296, 297