

The `eventB` package*

Thai Son Hoang
ETH-Zurich
<htson at inf dot ethz dot ch>

January 15, 2014

Abstract

This class provides a template for typesetting Event-B models. It was developed at the Swiss Federal Institute of Technology Zurich (ETH-Zurich).

Contents

1	Introduction	1
2	Usage	1
2.1	Package Options	2
3	Implementation	2
3.1	Package Loading	2
3.2	Internal Helper Macros	2
3.3	Declaration of Options for the Package	6
3.3.1	Option for bounding boxes	7
3.3.2	Options for font size and spacing	8
3.3.3	Options for colouring	9
3.4	Meta-macros for creating macros for modelling elements	11
3.5	Commands for Pretty-Print Event-B Models	11

1 Introduction

This package was developed in order to ease the typesetting of Event-B models in \LaTeX .

2 Usage

See `sample-eventB.tex` for an example of how to use the package.

*This document corresponds to `eventB` v2.0, dated 2014/01/14.

2.1 Package Options

The package offers the following options:

- `nobox`: to disable to bounding boxes for the Event-B modelling elements,
- `small`, `compact`, `tiny`: options for font size,
- `colour` (or `color`): to colour several modelling elements.

3 Implementation

3.1 Package Loading

We begin by loading the required package `xspace`, `xcolor`, and `etoolbox`.

```
1 \RequirePackage{xspace}
2 \RequirePackage{xcolor}
3 \RequirePackage{etoolbox}
```

3.2 Internal Helper Macros

We define same basic internal helper macros that will be used to defined other macros.

<code>\B@ifstrequal</code>	A utility wrapper for ake sure that the first argument is properly expanded. 4 <code>\newcommand{\B@ifstrequal}{\expandafter\ifstrequal\expandafter}</code>
<code>\B@keywordbase</code>	Basic macro for Event-B keywords. 5 <code>\newcommand{\B@keywordbase}[1]{\mathsf{#1}}</code>
<code>\B@identifierbase</code>	Basic macro for Event-B identifiers. 6 <code>\newcommand{\B@identifierbase}[1]{\mathit{#1}}</code>
<code>\B@labelbase</code>	Basic macro for Event-B labels. 7 <code>\newcommand{\B@labelbase}[2][]{\%</code> 8 <code>\ifstrequal{#1}{\}{\%</code> 9 <code>\mathsf{#2}</code> 10 <code>}\%</code> 11 <code>\mathit{#2}</code> 12 <code>}</code> 13 <code>}</code>
<code>\B@constructbase</code>	Basic macro for Event-B constructs. 14 <code>\newcommand{\B@constructbase}[1]{\mathsf{#1}}</code>
<code>\B@pobase</code>	Basic macro for Event-B proof obligations. 15 <code>\newcommand{\B@pobase}[1]{\mathsf{#1}}</code>
<code>\B@declarationbase</code>	Basic macro for Event-B declarations (e.g., constants, variables, etc.). 16 <code>\newcommand{\B@declarationbase}[2]{\%</code> 17 <code>\begin{array}{@{}l@{\B@tab}l@{}}</code> 18 <code>\B@keyword{#1:} & #2</code> 19 <code>\end{array}</code> 20 <code>}</code>

`\B@sectionbase` Basic macro for Event-B sections (e.g., axioms, invariants, etc.). The optional argument is to produce the keywords or not

```

21 \newcommand{\B@sectionbase}[3] [] {%
22   \ifstrequal{#1}{%{
23     \begin{array}{@{}l@{}}
24       \B@keyword{#2:} \\
25       \begin{array}{@{\B@tab}l@{\B@tab}l@{\B@tab}}
26         #3
27       \end{array}
28     \end{array}
29   }{
30     \begin{array}{@{\B@tab}l@{\B@tab}l@{\B@tab}}
31       #3
32     \end{array}
33   }
34 }

```

`\B@eventbase` Basic macro for pretty-print Event-B events.

```

35 \newcommand{\B@eventbase}[7] [] {%
36   { % BEGIN group

```

We first save the arguments to local variables.

```

37   \newcommand\evt@sts{#1}% Event status
38   \newcommand\evt@label{#2}% Event label
39   \newcommand\evt@absevt{#3}% Abstract event
40   \newcommand\evt@pars{#4}% Event parameters
41   \newcommand\evt@grds{#5}% Event guards
42   \newcommand\evt@wits{#6}% Event witnesses
43   \newcommand\evt@acts{#7}% Event actions

```

The convergence status is skipped if empty.

```

44   \B@ifstrequal{\evt@sts}{%{
45     \newcommand\pretty@sts{}
46   }{
47     \newcommand\pretty@sts{\B@tab\Bstatus \B@tab \evt@sts \\\}
48   }

```

The refines clause is skipped if there are no abstract events.

```

49   \B@ifstrequal{\evt@absevt}{%{
50     \newcommand\pretty@absevt{}
51   }{
52     \newcommand\pretty@absevt{\B@tab \Brefines \B@tab \evt@absevt{} \\\}
53   }

```

The parameters is skipped if there are none.

```

54   \B@ifstrequal{\evt@pars}{%{
55     \newcommand\pretty@pars{}
56   }{
57     \newcommand\pretty@pars{\B@tab \Bany \B@tab \evt@pars \B@tab \Bwhere \\\}
58   }

```

The keywords for guards also depends on if there are parameters or not.

```

59   \B@ifstrequal{\evt@grds}{%{
60     \newcommand\pretty@grds{}
61   }{
62     \newcommand\pretty@grds@tmp{

```

```

63     \begin{array}{@{\B@tab\B@tab}l@{\B@tab}l}
64     \evt@grds
65     \end{array}\\
66   }
67   \B@ifstrequal{\evt@pars}{}{
68     \newcommand\pretty@grds{
69       \B@tab \Bwhen \\
70       \pretty@grds@tmp
71     }
72   }{
73     \newcommand\pretty@grds{\pretty@grds@tmp}
74   }
75 }

```

The witnesses are skipped if there are none.

```

76   \B@ifstrequal{\evt@wits}{}{
77     \newcommand\pretty@wits{
78   }{
79     \newcommand\pretty@wits{
80       \B@tab\Bwith\\
81       \begin{array}{@{\B@tab\B@tab}l}
82         \evt@wits
83       \end{array}\\
84     }
85   }

```

When there are no actions, SKIP is used. The keyword is changed depending on whether the event has parameters or not.

```

86   \B@ifstrequal{\evt@acts}{}{
87     \renewcommand\evt@acts{\SKIP}
88   }{}
89   \newcommand\pretty@acts@tmp{
90     \begin{array}{@{\B@tab\B@tab}l@{\B@tab}l}
91       \evt@acts
92     \end{array}\\
93   }
94   \newcommand\pretty@acts@keyword{\B@tab\Bthen \\}
95   \B@ifstrequal{\evt@pars}{}{
96     \B@ifstrequal{\evt@grds}{}{
97       \renewcommand\pretty@acts@keyword{\B@tab\Bbegin \\}
98     }{}
99   }{}
100  \newcommand\pretty@acts{
101    \pretty@acts@keyword
102    \pretty@acts@tmp
103  }

```

Finally we put all the pretty-print pieces together.

```

104  \begin{array}{@{}l@{}}
105    \Bevt{\evt@label} \\
106    \pretty@sts
107    \pretty@absepts
108    \pretty@pars
109    \pretty@grds
110    \pretty@wits

```

```

111     \pretty@acts
112     \B@tab\Bend
113   \end{array}
114 } % END group
115 }

```

`\B@inlineeventbase` Basic macro for pretty-print Event-B events inline.

```

116 \newcommand{\B@inlineeventbase}[7] [] {
117   { % BEGIN group
118     We first save the arguments to local variables.
119     \newcommand\evt@sts{#1}% Event status
120     \newcommand\evt@label{#2}% Event label
121     \newcommand\evt@absevt{#3}% Abstract event
122     \newcommand\evt@pars{#4}% Event parameters
123     \newcommand\evt@grds{#5}% Event guards
124     \newcommand\evt@wits{#6}% Event witnesses
125     \newcommand\evt@acts{#7}% Event actions
126
127     The convergence status is skipped if empty.
128     \B@ifstrequal{\evt@sts}{}{
129       \newcommand\pretty@sts{}
130     }{
131       \newcommand\pretty@sts{(\evt@sts)}
132     }
133
134     The refines clause is skipped if there are no abstract events.
135     \B@ifstrequal{\evt@absevt}{}{
136       \newcommand\pretty@absevt{}
137     }{
138       \newcommand\pretty@absevt{~\Brefines~\evt@absevt}
139     }
140
141     The parameters is skipped if there are none.
142     \B@ifstrequal{\evt@pars}{}{
143       \newcommand\pretty@pars{}
144     }{
145       \newcommand\pretty@pars{\Bany~\evt@pars~\Bwhere~}
146     }
147
148     The keywords for guards also depends on if there are parameters or not.
149     \B@ifstrequal{\evt@grds}{}{
150       \newcommand\pretty@grds{}
151     }{
152       \newcommand\pretty@grds@tmp{
153         \evt@grds~
154       }
155       \B@ifstrequal{\evt@pars}{}{
156         \newcommand\pretty@grds{\Bwhen~\pretty@grds@tmp}
157       }{
158         \newcommand\pretty@grds{\pretty@grds@tmp}
159       }
160     }
161
162     The witnesses are skipped if there are none.
163     \B@ifstrequal{\evt@wits}{}{

```

```

153     \newcommand\pretty@wits{}
154   }{
155     \newcommand\pretty@wits{
156       \Bwith~
157       \evt@wits~
158     }
159   }

```

When there are no actions, SKIP is used. The keyword is changed depending on whether the event has parameters or not.

```

160   \B@ifstrequal{\evt@acts}{}{
161     \renewcommand\evt@acts{\SKIP}
162   }{}
163   \newcommand\pretty@acts@tmp{
164     \evt@acts
165   }
166   \newcommand\pretty@acts@keyword{\Bthen}
167   \B@ifstrequal{\evt@pars}{}{
168     \B@ifstrequal{\evt@grds}{}{
169       \renewcommand\pretty@acts@keyword{\Bbegin}
170     }{}
171   }{}
172   \newcommand\pretty@acts{
173     \pretty@acts@keyword~
174     \pretty@acts@tmp~
175   }

```

Finally we put all the pretty-print pieces together.

```

176   \Bevt{\evt@label}\pretty@sts\pretty@absepts~\widehat{=}~
177   \pretty@pars
178   \pretty@grds
179   \pretty@wits
180   \pretty@acts
181   \Bend
182 } % END group
183 }

```

\B@makebox A wrapper macro to make a fbox with the boundary adjusted.

```

184 \newlength{\B@tmp@length}
185 \newcommand{\B@makebox}[1]{
186   {
187     \setlength{\B@tmp@length}{\fboxsep}
188     \setlength{\fboxsep}{2ex}
189     \fbox{#1}
190     \setlength{\fboxsep}{\B@tmp@length}
191   }
192 }

```

3.3 Declaration of Options for the Package

In this part various options for the package are defined.

3.3.1 Option for bounding boxes

By default, Event-B modelling elements, e.g., invariants, events, etc., are displayed in a bounding box. This `nobox` option enables them to be displayed without the bounding box.

`\B@event` Default definition displays Event-B events in a box.

```
193 \newcommand{\B@event}[7] [] {
194   \B@makebox{
195     \ensuremath{
196       \B@eventbase[#1]{#2}{#3}{#4}{#5}{#6}{#7}
197     }
198   }
199 }
```

`\B@declaration` Default definition displays Event-B declarations in a box.

```
200 \newcommand{\B@declaration}[2] {
201   \B@makebox{
202     \ensuremath{
203       \B@declarationbase[#1]{#2}
204     }
205   }
206 }
```

`\B@section` Default definition displays Event-B sections in a box

```
207 \newcommand{\B@section}[3] [] {
208   \B@makebox{
209     \ensuremath{
210       \B@sectionbase[#1]{#2}{#3}
211     }
212   }
213 }
```

Option “nobox” The above commands are redefined accordingly when option `nobox` is enabled.

```
214 \DeclareOption{nobox}{
```

`\B@event` Redefine the definition without the bounding box.

```
215   \renewcommand{\B@event}[7] [] {
216     \B@eventbase[#1]{#2}{#3}{#4}{#5}{#6}{#7}
217   }
```

`\B@declaration` Redefine the definition without the bounding box.

```
218   \renewcommand{\B@declaration}[2] {
219     \B@declarationbase[#1]{#2}
220   }
```

`\B@section` Redefine the definition without the bounding box.

```
221   \renewcommand{\B@section}[3] [] {
222     \B@sectionbase[#1]{#2}{#3}
223   }
224 }
```

3.3.2 Options for font size and spacing

We define the default values for font size and some spacing commands, and how they are redefined according to options `small`, `compact`, and `tiny`. In particular, option `compact` and `tiny` implies option `nobox`.

`\B@fontsize` The font size used in the `Bcode` environment (defined later).

```
225 \newcommand{\B@fontsize}{\normalsize}
```

`\B@vspace` A vertical rule for spacing, defaulted to be `2ex`.

```
226 \newcommand{\B@vspace}[1][2ex]{\[#1]}
```

`\B@hspace` A horizontal rule for spacing, defaulted to be `2em`.

```
227 \newcommand{\B@hspace}[1][2em]{\hspace{#1}}
```

`\B@tab` A small tab for spacing, defaulted to be `\quad`.

```
228 \newcommand{\B@tab}{\quad} % A small separation space
```

We subsequently redefined the above spacing commands when one of the options `small`, `compact`, `tiny` is enabled.

Option “small” For option `small` they are adjusted as follows.

```
229 \DeclareOption{small}{
```

`\B@fontsize` Redefine to be `\small` for option `small`.

```
230 \renewcommand{\B@fontsize}{\small}
```

`\B@vspace` Redefine to be `1ex` for option `small`.

```
231 \renewcommand{\B@vspace}[1][1ex]{\[#1]}
```

`\B@hspace` Redefine to be `1em` for option `small`.

```
232 \renewcommand{\B@hspace}[1][1em]{\hspace{#1}}
```

`\B@tab` Redefine to be `\` for option `small`.

```
233 \renewcommand{\B@tab}{\ }
```

```
234 }
```

Option “compact” For option `compact` the commands are adjusted as follows.

```
235 \DeclareOption{compact}{
```

`\B@fontsize` Redefine to be `\footnotesize` for option `compact`.

```
236 \renewcommand{\B@fontsize}{\footnotesize}
```

`\B@vspace` Redefine to be `0ex` for option `compact`.

```
237 \renewcommand{\B@vspace}[1][0ex]{\[#1]}
```

`\B@hspace` Redefine to be `0.5em` for option `compact`.

```
238 \renewcommand{\B@hspace}[1][0.5em]{\hspace{#1}}
```

`\B@tab` Redefine to be `\` for option `compact`.

```
239 \renewcommand{\B@tab}{\ }
```

Option `nobox` is enabled.

```
240 \ExecuteOptions{nobox}
```

```
241 }
```


Option “tiny” For option `tiny` the commands are adjusted as follows.

```

242 \DeclareOption{tiny}{
\B@fontsize  Redefine to be \scriptsize for option tiny.
243   \renewcommand{\B@fontsize}{\scriptsize}

\B@vspace  Redefine to be -0.5ex for option tiny.
244   \renewcommand{\B@vspace}[1]{-0.5ex}{\[#1]}

\B@hspace  Redefine to be 0.5em for option compact.
245   \renewcommand{\B@hspace}[1]{0.5em}{\hspace{#1}}

\B@etab  Redefine to be \ for option compact.
246   \renewcommand{\B@etab}{\ }

  Option nobox is enabled.
247   \ExecuteOptions{nobox}
248 }
```

3.3.3 Options for colouring

Keywords, labels and identifiers in Event-B can be coloured. We define several commands and redefine them accordingly for colouring. When `colour` (or `color`) option is enabled, one can customise the colours for Event-B keywords, labels or identifier or proof obligation labels. We proceed with some definitions that can be redefined by these options.

```

\B@keyword  Macro for Event-B keywords.
249 \newcommand{\B@keyword}[1]{\ensuremath{\B@keywordbase{#1}}\xspace}

\B@identifier  Macro for Event-B identifiers.
250 \newcommand{\B@identifier}[1]{\ensuremath{\B@identifierbase{#1}}\xspace}

\B@label  Macro for Event-B labels.
251 \newcommand{\B@label}[2][ ]{\ensuremath{\B@labelbase{#1}{#2}}\xspace}

\B@construct  Macro for Event-B constructs.
252 \newcommand{\B@construct}[1]{\ensuremath{\B@constructbase{#1}}\xspace}

\B@po  Macro for Event-B proof obligations.
253 \newcommand{\B@po}[1]{\ensuremath{\B@pobase{#1}}\xspace}
```

We redefine the above commands if option `colour` or `color` is enabled. Furthermore, we define some commands for setting colour for various modelling elements.

Option 'colour' The option colour is declared as follows.

```
254 \DeclareOption{colour}{
```

`\setBKeywordColour` Utility macro for defining Event-B keywords colour.

```
255 \newcommand{\setBKeywordColour}[1]{\colorlet{B@keywordcolor}{#1}}
```

```
256 \setBKeywordColour{blue}
```

`\setBIdentifierColour` Utility macro for defining Event-B identifiers colour.

```
257 \newcommand{\setBIdentifierColour}[1]{\colorlet{B@identifiercolor}{#1}}
```

```
258 \setBIdentifierColour{blue!50!red}
```

`\setBLabelColour` Utility macro for defining Event-B labels colour.

```
259 \newcommand{\setBLabelColour}[1]{\colorlet{B@labelcolor}{#1}}
```

```
260 \setBLabelColour{green!50!black}
```

`\setBConstructColour` Utility macro for defining Event-B constructs colour.

```
261 \newcommand{\setBConstructColour}[1]{\colorlet{B@constructcolor}{#1}}
```

```
262 \setBConstructColour{orange!75!black}
```

`\setBP0Colour` Utility macro for defining Event-B proof obligations colour.

```
263 \newcommand{\setBP0Colour}[1]{\colorlet{B@pocolor}{#1}}
```

```
264 \setBP0Colour{red}
```

We redefine the commands with colours.

```
265 \renewcommand{\B@keyword}[1]{% Remove the leading space
266 \ensuremath{\textcolor{B@keywordcolor}{\B@keywordbase{#1}}}\xspace
267 }
268 \renewcommand{\B@identifier}[1]{% Remove the leading space
269 \ensuremath{\textcolor{B@identifiercolor}{\B@identifierbase{#1}}}\xspace
270 }
271 \renewcommand{\B@label}[2][1]{% Remove the leading space
272 \ensuremath{\textcolor{B@labelcolor}{\B@labelbase{#1}{#2}}}\xspace
273 }
274 \renewcommand{\B@construct}[1]{% Remove the leading space
275 \ensuremath{\textcolor{B@constructcolor}{\B@constructbase{#1}}}\xspace
276 }
277 \renewcommand{\B@po}[1]{% Remove the leading space
278 \ensuremath{\textcolor{B@pocolor}{\B@pobase{#1}}}\xspace
279 }
280 }
```

Option 'color' This option is a pointer to colour.

```
281 \DeclareOption{color}{
```

```
282 \ExecuteOptions{colour}
```

```
283 }
```

After declaration of options, we execute them accordingly.

```
284 \ProcessOptions
```

3.4 Meta-macros for creating macros for modelling elements

We define meta-macros to create macros for different modelling elements.

```

285 \newcommand{\B@newmacro}[3][]{
286   \ifstrequal{#1}{}{
287     \expandafter\def\csname #2\endcsname{#3{#2}}
288   }{
289     \expandafter\def\csname #1\endcsname{#3{#2}}
290   }
291 }

292 \newcommand{\newBctx}[2][]{\B@newmacro[#1]{#2}{\Bctx}}
293 \newcommand{\newBset}[2][]{\B@newmacro[#1]{#2}{\Bset}}
294 \newcommand{\newBcst}[2][]{\B@newmacro[#1]{#2}{\Bcst}}
295 \newcommand{\newBaxm}[2][]{\B@newmacro[#1]{#2}{\Baxm}}
296 \newcommand{\newBthm}[2][]{\B@newmacro[#1]{#2}{\Bthm}}
297 \newcommand{\newBmch}[2][]{\B@newmacro[#1]{#2}{\Bmch}}
298 \newcommand{\newBvrb}[2][]{\B@newmacro[#1]{#2}{\Bvrb}}
299 \newcommand{\newBinu}[2][]{\B@newmacro[#1]{#2}{\Binu}}
300 \newcommand{\newBevt}[2][]{\B@newmacro[#1]{#2}{\Bevt}}
301 \newcommand{\newBpar}[2][]{\B@newmacro[#1]{#2}{\Bpar}}
302 \newcommand{\newBgrd}[2][]{\B@newmacro[#1]{#2}{\Bgrd}}
303 \newcommand{\newBact}[2][]{\B@newmacro[#1]{#2}{\Bact}}
304 \newcommand{\newBkeyword}[2][]{\B@newmacro[#1]{#2}{\Bkeyword}}

```

3.5 Commands for Pretty-Print Event-B Models

We start with the definition of the `\eventB` macro.

```

305 \newcommand{\eventB}{Event-B\xspace}

```

The `Bcode` environment for displaying Event-B models. The environment has an optional argument for specifying the font size. By default, it is the same as the `\B@fontsize` controlled by the package option.

```

306 \newenvironment{Bcode}[1][\B@fontsize]{\begin{center}#1{\end{center}}

```

Declarations and Collections Event-B modelling elements are organised into declarations (e.g., variables, constants, etc.) or collections (e.g., invariants, axioms). For each declaration, the input is a comma-separated list of elements. For each collection, the input is a newly(`\`)-separated list of elements.

```

307 \newcommand{\carriersets}[1]{%
308   \B@declaration{sets}{#1}
309 }

310 \newcommand{\constants}[1]{%
311   \B@declaration{constants}{#1}
312 }

```

```

313 \newcommand{\axioms}[2][]{\%
314   \B@section[#1]{axioms}{#2}
315 }

316 \newcommand{\variables}[1]{
317   \B@declaration{variables}{#1}
318 }

319 \newcommand{\invariants}[2][]{
320   \B@section[#1]{invariants}{#2}
321 }

322 \newcommand{\variant}[1]{
323   \B@declaration{variant}{#1}
324 }

```

Event-B keywords We define the keywords for pretty-print Event-B models.

```

325 \newBkeyword[Bcontext]{context}
326 \newBkeyword[Bsets]{sets}
327 \newBkeyword[Bconstants]{constants}
328 \newBkeyword[Baxioms]{axioms}
329
330 \newBkeyword[Bmachine]{machine}
331 \newBkeyword[Brefines]{refines}
332 \newBkeyword[Bsees]{sees}
333 \newBkeyword[Bvariables]{variables}
334 \newBkeyword[Binvariants]{invariants}
335 \newBkeyword[Bevents]{events}
336
337 \newBkeyword[Bevent]{event}
338 \newBkeyword[Bextends]{extends}
339 \newBkeyword[Bany]{any}
340 \newBkeyword[Bbegin]{begin}
341 \newBkeyword[Bstatus]{status}
342 \newBkeyword[Bthen]{then}
343 \newBkeyword[Bwhen]{when}
344 \newBkeyword[Bwhere]{where}
345 \newBkeyword[Bwith]{with}
346 \newBkeyword[Bend]{end}

```

Event-B modelling elements We define several macros for pretty-print Event-B modelling elements.

```

347 \newcommand{\Bctx}[1]{\B@construct{#1}}
348 \newcommand{\Bset}[1]{\B@identifier{#1}}
349 \newcommand{\Bcst}[1]{\B@identifier{#1}}
350 \newcommand{\Baxm}[1]{\B@label{#1}}
351 \newcommand{\Bthm}[1]{\B@label{thm}{#1}}
352
353 \newcommand{\Bmch}[1]{\B@construct{#1}}
354 \newcommand{\Bvrb}[1]{\B@identifier{#1}}
355 \newcommand{\Binv}[1]{\B@label{#1}}
356 \newcommand{\Bevt}[1]{\B@label{#1}}
357 \newcommand{\Bpar}[1]{\B@identifier{#1}}
358 \newcommand{\Bact}[1]{\B@label{#1}}
359 \newcommand{\Bgrd}[1]{\B@label{#1}}

```

```

360 \newcommand{\Bbap}[1]{\hbox{\sl\bfseries #1}}
361
362 %%%% Theorem Proof Obligation
363 %%%% Print the theorem proof obligation, given the theorem label.
364 %%%% Arguments:
365 %%%% 1. Theorem label
366 %%%%
367 %%%% Usage:
368 %%%% - \thmpo{thm} will produce "thm/THM"
369 \newcommand{\thmpo}[1]{\Bthm{#1}/\B@po{THM}}
370
371 %%%% Axiom Well-definedness Proof Obligation
372 %%%% Print the axiom well-definedness proof obligation, given the
373 %%%% axiom label.
374 %%%% Arguments:
375 %%%% 1. Axiom label
376 %%%%
377 %%%% Usage:
378 %%%% - \axmwdpo{axm} will produce "axm/WD"
379 \newcommand{\axmwdpo}[1]{\Baxm{#1}/\B@po{WD}}
380
381 %%%% Invariant Proof Obligation
382 %%%% Print the invariant proof obligation, given the event name and
383 %%%% invariant label
384 %%%% Arguments:
385 %%%% 1. Event name
386 %%%% 2. Invariant label
387 %%%%
388 %%%% Usage:
389 %%%% - \invpo{evt}{inv} will produce "evt/inv/INV"
390 \newcommand{\invpo}[2]{\Bevt{#1}/\Binv{#2}/\B@po{INV}}
391
392 %%%% Theorem (in guard) Proof Obligation
393 %%%% Print the simulation proof obligation, given the event name and
394 %%%% the theorem (in guard) label.
395 %%%% Arguments:
396 %%%% 1. Event name
397 %%%% 2. Theorem (in guard) label
398 %%%%
399 %%%% Usage:
400 %%%% - \grdthmpo{evt}{thm} will produce "evt/thm/THM"
401 \newcommand{\grdthmpo}[2]{\Bevt{#1}/\Bthm{#2}/\B@po{THM}}
402
403 %%%% Feasibility Proof Obligation
404 %%%% Print the feasibility proof obligation, given the event name and
405 %%%% the action label
406 %%%% Arguments:
407 %%%% 1. Event name
408 %%%% 2. Action label
409 %%%%
410 %%%% Usage:
411 %%%% - \fispo{evt}{act} will produce "evt/act/FIS"

```

```

412 \newcommand{\fispo}[2]{\Bevt{#1}/\Bact{#2}/\B@po{FIS}}
413
414 %%%% Variant finiteness Proof Obligation
415 %%%% Print the Variant finiteness proof obligation
416 %%%% Arguments: No arguments
417 %%%%
418 %%%% Usage:
419 %%%% - \finpo will produce "FIN"
420 \newcommand{\finpo}{\B@po{FIN}}
421
422 %%%% Variant Proof Obligation
423 %%%% Print the guard strengthen proof obligation, given the event name
424 %%%% Arguments:
425 %%%% 1. Event name
426 %%%%
427 %%%% Usage:
428 %%%% - \grdpo{evt} will produce "evt/VAR"
429 \newcommand{\varpo}[1]{\Bevt{#1}/\B@po{VAR}}
430
431 %%%% Simulation Proof Obligation
432 %%%% Print the simulation proof obligation, given the event name and
433 %%%% the action label.
434 %%%% Arguments:
435 %%%% 1. Event name
436 %%%% 2. Action label
437 %%%%
438 %%%% Usage:
439 %%%% - \simpo{evt}{act} will produce "evt/act/SIM"
440 \newcommand{\simpo}[2]{\Bevt{#1}/\Bact{#2}/\B@po{SIM}}
441
442 %%%% Guard Strengthen Proof Obligation
443 %%%% Print the guard strengthen proof obligation, given the event
444 %%%% name and the guard label
445 %%%% Arguments:
446 %%%% 1. (Abstract) Event name
447 %%%% 2. (Abstract) Guard label
448 %%%%
449 %%%% Usage:
450 %%%% - \grdpo{evt}{grd} will produce "evt/grd/GRD"
451 \newcommand{\grdpo}[2]{\Bevt{#1}/\Bgrd{#2}/\B@po{GRD}}
452
453 %%%% Variant Natural Number Proof Obligation
454 %%%% Print the Variant Natural Number proof obligation, given the event name
455 %%%% Arguments:
456 %%%% 1. Event name
457 %%%%
458 %%%% Usage:
459 %%%% - \natpo{evt} will produce "evt/NAT"
460 \newcommand{\natpo}[1]{\Bevt{#1}/\B@po{NAT}}
461

```

\inlineevent

```

462 \newcommand{\inlineevent}[7] [] {
463   \B@inlineeventbase[#1]{#2}{#3}{#4}{#5}{#6}{#7}
464 }

465
466
467
468
469
470
471
472 %%%% (BEGIN) Macros for Pretty-Print Event-B Components %%%
473 \newcommand{\SKIP}{\textsc{skip}\xspace}
474 %

```

\INITIALISATION

```

475 %%%% INITIALISATION label
476 \newBevt{INITIALISATION}

477
478 %%%% Pretty print the initialisation: no ‘‘refines’’ clause. no parameters, no
479 %%%% guards
480 %%%% Arguments:
481 %%%% 1. (Newline(\\)-separated) list of assignments.
482 %%%%
483 %%%% Usage: \initialisation{S1(v,x,y)\\S2(w,x,y)}
484 %%%% will produce the following
485 %%%%
486 %%%% init
487 %%%% begin
488 %%%% S1(v, x, y)
489 %%%% S2(w, x, y)
490 %%%% end
491 %%%%
492 \newcommand{\initialisation}[1]{
493   \event{\INITIALISATION}{-}{-}{-}{-}{#1}
494 }
495
496 %\newcommand{\event}[7] [] {
497 % \B@event[#1]{#2}{#3}{#4}{#5}{#6}{#7}
498 %}
499
500 \let\event\B@event
501 \let\Bvspace\B@vspace
502 \let\Bhspace\B@hspace
503 \let\Bpo\B@po
504

```

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.

$\backslash \square$	233, 239, 246	$\backslash \text{B@po}$	<u>253</u> , 277, 369, 379, 390, 401, 412, 420, 429, 440, 451, 460, 503	$\backslash \text{constants}$	310
A		$\backslash \text{B@pobase}$	<u>15</u> , 253, 278	E	
$\backslash \text{axioms}$	313	$\backslash \text{B@section}$ <u>207</u> , <u>221</u> , 314, 320	$\backslash \text{event}$	493, 496, 500
$\backslash \text{axmwdpo}$	378, 379	$\backslash \text{B@sectionbase}$ <u>21</u> , 210, 222	$\backslash \text{eventB}$	305
B		$\backslash \text{B@tab}$	17, 25, 30, 47, 52, 57, 63, 69, 80, 81, 90, 94, 97, 112, <u>228</u> , <u>233</u> , <u>239</u> , <u>246</u>	$\backslash \text{evt@absevt}$	39, 49, 52, 120, 130, 133
$\backslash \text{B@construct}$ <u>252</u> , 274, 347, 353	$\backslash \text{B@vspace}$	<u>226</u> , <u>231</u> , <u>237</u> , <u>244</u> , 501	$\backslash \text{evt@acts}$ 43, 86, 87, 91, 124, 160, 161, 164
$\backslash \text{B@constructbase}$	<u>14</u> , 252, 275	$\backslash \text{Bact}$	303, 358, 412, 440	$\backslash \text{evt@grds}$ 41, 59, 64, 96, 122, 140, 144, 168
$\backslash \text{B@declaration}$	<u>200</u> , <u>218</u> , 308, 311, 317, 323	$\backslash \text{Bany}$	57, 138	$\backslash \text{evt@label}$ 38, 105, 119, 176
$\backslash \text{B@declarationbase}$	<u>16</u> , 203, 219	$\backslash \text{Baxm}$	295, 350, 379	$\backslash \text{evt@pars}$	40, 54, 57, 67, 95, 121, 135, 138, 146, 167
$\backslash \text{B@event}$ <u>193</u> , <u>215</u> , 497, 500	$\backslash \text{Bbap}$	360	$\backslash \text{evt@sts}$	37, 44, 47, 118, 125, 128
$\backslash \text{B@eventbase}$	<u>35</u> , 196, 216	$\backslash \text{Bbegin}$	97, 169	$\backslash \text{evt@wits}$	42, 76, 82, 123, 152, 157
$\backslash \text{B@fontsize}$	<u>225</u> , <u>230</u> , <u>236</u> , <u>243</u> , 306	$\backslash \text{Bcst}$	294, 349	F	
$\backslash \text{B@hspace}$	<u>227</u> , <u>232</u> , <u>238</u> , <u>245</u> , 502	$\backslash \text{Bctx}$	292, 347	$\backslash \text{finpo}$	419, 420
$\backslash \text{B@identifier}$ <u>250</u> , 268, 348, 349, 354, 357	$\backslash \text{Bend}$	112, 181	$\backslash \text{fispo}$	411, 412
$\backslash \text{B@identifierbase}$ <u>6</u> , 250, 269	$\backslash \text{Bevt}$	105, 176, 300, 356, 390, 401, 412, 429, 440, 451, 460	G	
$\backslash \text{B@ifstrequal}$ <u>4</u> , 44, 49, 54, 59, 67, 76, 86, 95, 96, 125, 130, 135, 140, 146, 152, 160, 167, 168	$\backslash \text{bfseries}$	360	$\backslash \text{grdpo}$	428, 450, 451
$\backslash \text{B@inlineeventbase}$ <u>116</u> , 463	$\backslash \text{Bgrd}$	302, 359, 451	$\backslash \text{grdthmpo}$	400, 401
$\backslash \text{B@keyword}$	18, 24, <u>249</u> , 265, 304	$\backslash \text{Bhspace}$	502	I	
$\backslash \text{B@keywordbase}$ <u>5</u> , 249, 266	$\backslash \text{Binv}$	299, 355, 390	$\backslash \text{INITIALISATION}$	<u>475</u> , 493
$\backslash \text{B@label}$	<u>251</u> , 271, 350, 351, 355, 356, 358, 359	$\backslash \text{Bmch}$	297, 353	$\backslash \text{initialisation}$	483, 492
$\backslash \text{B@labelbase}$	<u>7</u> , 251, 272	$\backslash \text{Bpar}$	301, 357	$\backslash \text{inlineevent}$	<u>462</u>
$\backslash \text{B@makebox}$ <u>184</u> , 194, 201, 208	$\backslash \text{Bpo}$	503	$\backslash \text{invariants}$	319
$\backslash \text{B@newmacro}$	285, 292–304	$\backslash \text{Brefines}$	52, 133	$\backslash \text{invpo}$	389, 390
		$\backslash \text{Bset}$	293, 348	L	
		$\backslash \text{Bstatus}$	47	$\backslash \text{let}$	500–503
		$\backslash \text{Bthen}$	94, 166	N	
		$\backslash \text{Bthm}$	296, 351, 369, 401	$\backslash \text{natpo}$	459, 460
		$\backslash \text{Bvrb}$	298, 354	$\backslash \text{newBact}$	303
		$\backslash \text{Bvspace}$	501	$\backslash \text{newBaxm}$	295
		$\backslash \text{Bwhen}$	69, 147	$\backslash \text{newBcst}$	294
		$\backslash \text{Bwhere}$	57, 138	$\backslash \text{newBctx}$	292
		$\backslash \text{Bwith}$	80, 156	$\backslash \text{newBevt}$	300, 476
		C			
		$\backslash \text{carrier}$	307		

<code>\newBgrd</code>	302	94, 97,	<u>261</u>
<code>\newBin</code>	299	101, 166, 169, 173	<code>\setBIdentifierColour</code>	
<code>\newBkeyword</code>		<code>\pretty@acts@tmp</code>	<u>257</u>
. 304, 325–328, 330–335, 337–346		. 89, 102, 163, 174	<code>\setBKeywordColour</code> .	<u>255</u>
<code>\newBmch</code>	297	<code>\pretty@grds</code>	<code>\setBLabelColour</code> . .	<u>259</u>
<code>\newBpar</code>	301	60, 68, 73, 109, 141, 147, 149, 178	<code>\setBP0Colour</code>	<u>263</u>
<code>\newBset</code>	293	<code>\pretty@grds@tmp</code> 62,	<code>\simpo</code>	439, 440
<code>\newBthm</code>	296	70, 73, 143, 147, 149	<code>\SKIP</code>	87, 161, 473
<code>\newBvrb</code>	298	<code>\pretty@pars</code> . 55, 57, 108, 136, 138, 177	<code>\sl</code>	360
P				
<code>\pretty@absepts</code> . . .		<code>\pretty@sts</code> . . 45, 47, 106, 126, 128, 176	T	
. 50, 52, 107, 131, 133, 176		<code>\pretty@wits</code> . 77, 79, 110, 153, 155, 179	<code>\thmpo</code>	368, 369
<code>\pretty@acts</code>			V	
. 100, 111, 172, 180		S	<code>\variables</code>	316
<code>\pretty@acts@keyword</code>		<code>\setBConstructColour</code>	<code>\variant</code>	322
			<code>\varpo</code>	429

Change History

v1.0		v1.1.1	
General: Initial version	1	General: Updated documentation . .	1
v1.0.1		v2.0	
<code>\B@declaration:</code> Ensure math-mode	7	General: Major re-implementation, use etoolbox instead of ifthen	1
<code>\B@event:</code> Ensure math-mode	7	Updated documentation, added DoNotIndex	1
<code>\B@section:</code> Ensure math-mode . .	7	<code>\B@makebox:</code> Added	6
v1.1		<code>\INITIALISATION:</code> Renamed from init	15
General: Re-implement how options are defined, added options 'nobox'	1	<code>\inlineevent:</code> Renamed from eventinline	14