

# The `lstEventB` package\*

Thai Son Hoang and Chenyang Zhu  
ECS, University of Southampton

<{T dot S dot Hoang, C dot Zhu} at ecs dot soton dot ac dot uk>

April 28, 2018

## Abstract

This package provides macros for listing Event-B code. It was developed at the University of Southampton.

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Usage</b>	<b>1</b>
2.1	Package Options . . . . .	2
2.2	Typesetting Event-B Code . . . . .	2
<b>3</b>	<b>Implementation</b>	<b>3</b>
3.1	Package Options . . . . .	3
3.1.1	Colouring option . . . . .	3
3.1.2	Execution of options . . . . .	4
3.2	Typesetting of the Event-B language . . . . .	4
3.2.1	Defining the Event-B language . . . . .	4
3.2.2	Typesetting Event-B Code . . . . .	8

## 1 Introduction

This package was developed in order to ease the listing of Event-B code in `LATEX`.

## 2 Usage

Just like any other package, you need to request this package with a `\usepackage` command in the preamble. So in the simpler case (i.e., without any options), one just types

```
\usepackage{lstEventB}
```

to load the package.

---

\*This document corresponds to `lstEventB` v1.0, dated 2018/05/14.

## 2.1 Package Options

**Colouring Option.** Loading package with the `colour` or `color` options will enable the various colouring of the Event-B code.

## 2.2 Typesetting Event-B Code

The current supported syntax are from XEvent-B (cite XEvent-B paper). In particular, the Event-B mathematical symbols can be typeset using Unicode symbols. Alternatively, the mathematical symbols can be typeset using ASCII combinations (similar to the Event-B Summary (cite Ken Robinson)), with the exception that the *text* combinations must be prefixed by `!` (this is to prevent unintended translation of text in longer words). Some other symbols, e.g. `.` and `|` also need to be `!`-prefixed.

Table 1 shows the ASCII input for typesetting predicate-related symbols.

ASCII	Symbols	Explanation
<code>!false</code>	$\perp$	False
<code>!true</code>	$\top$	True
<code>&amp;</code>	$\wedge$	Conjunction
<code>!or</code>	$\vee$	Disjunction
<code>=&gt;</code>	$\Rightarrow$	Implication
<code>&lt;=&gt;</code>	$\Leftrightarrow$	Equivalence
<code>!not</code>	$\neg$	Negation
<code>!</code>	$\forall$	Universal quantification
<code>#</code>	$\exists$	Existential quantification
<code>!. </code>	$\cdot$	Quantification dot
<code>=</code>	$=$	Equality
<code>/=</code>	$\neq$	Inequality

Table 1: Predicates

Table 2 shows the ASCII inputs for typesetting set-related symbols.

ASCII	Symbols	Explanation
<code>{}</code>	$\emptyset$	Empty set
<code> </code>	$ $	Vertical bar, e.g., in set comprehension
<code>\ /</code>	$\cup$	Union
<code>\ \</code>	$\cap$	Intersection
<code>\</code>	$\setminus$	Set difference
<code> -&gt;</code>	$\mapsto$	Ordered pair
<code>**</code>	$\times$	Cartesian product
<code>!POW</code>	$\mathbb{P}$	Powerset
<code>!POW1</code>	$\mathbb{P}_1$	Non-empty subsets
<code>!card</code>	<code>card</code>	Cardinality
<code>!union</code>	<code>union</code>	Generalised union
<code>!inter</code>	<code>inter</code>	Generalised intersection
<code>!UNION</code>	$\bigcup$	Quantified union
<code>!INTER</code>	$\bigcap$	Quantified intersection

Table 2: Sets

## 3 Implementation

Our implementation is based on the `listings` package. Additionally, we also require `xspace` for spacing, `xcolor` for colouring, `bsymb` for typesetting Event-B mathematical symbols, and `xargs` for defining commands with argument lists.

```
\RequirePackage{listings}
\RequirePackage{xspace}
\RequirePackage{xcolor}
\RequirePackage{bsymb}
\RequirePackage{xargs}
\RequirePackage{mdframed}
```

### 3.1 Package Options

We define some options for customising the listing of Event-B code.

#### 3.1.1 Colouring option

We first declare some internal macros that can be updated when accordingly to the option for colouring.

EventB@SetKeywordColour	Command <code>EventB@SetKeywordColour</code> is used to set the colour of the Event-B keywords, by default, it is set to <code>black</code> . <pre>\newcommand{\EventB@SetKeywordColour}[1]{%   \colorlet{EventB@keywordcolour}{#1}% } \EventB@SetKeywordColour{black}</pre>
EventB@SetNdKeywordColour	Command <code>EventB@SetNdKeywordColour</code> is used to set the colour of the secondary Event-B keywords, by default, it is set to <code>black</code> . <pre>\newcommand{\EventB@SetNdKeywordColour}[1]{%   \colorlet{EventB@ndkeywordcolour}{#1}% } \EventB@SetNdKeywordColour{black}</pre>
EventB@SetIdentifierColour	Command <code>EventB@SetIdentifierColour</code> is used to set the colour of Event-B identifiers, by default, it is set to <code>black</code> . <pre>\newcommand{\EventB@SetIdentifierColour}[1]{%   \colorlet{EventB@identifiercolour}{#1}% } \EventB@SetIdentifierColour{black}</pre>
EventB@SetCommentColour	Command <code>EventB@SetCommentColour</code> is used to set the colour of Event-B comments, by default, it is set to <code>black</code> . <pre>\newcommand{\EventB@SetCommentColour}[1]{%   \colorlet{EventB@commentcolour}{#1}% } \EventB@SetCommentColour{black}</pre>
EventB@SetFormulaColour	Command <code>EventB@SetFormulaColour</code> is used to set the colour of Event-B formulae, by default, it is set to <code>black</code> . <pre>\newcommand{\EventB@SetFormulaColour}[1]{%</pre>

```

\colorlet{EventB@formulacolour}{#1}%
}
\EventB@SetFormulaColour{black}

```

We now define the **colour** option and set the different colours accordingly. The keywords colour (both first primary and secondary keywords) is **red**. The identifier colour is **purple**. The comment colour is **green** $\forall 50\forall$ **black** (dark green). The formula colour is **blue**.

```

\DeclareOption{colour}{
  \EventB@SetKeywordColour{red}
  \EventB@SetNdKeywordColour{red}
  \EventB@SetIdentifierColour{purple}
  \EventB@SetCommentColour{green!50!black}
  \EventB@SetFormulaColour{blue}
}

```

Additionally, we define the **color** option as an alias of **colour**.

```

\DeclareOption{color}{
  \ExecuteOptions{colour}
}

```

### 3.1.2 Execution of options

```

\ProcessOptions

```

## 3.2 Typesetting of the Event-B language

In this section, we define how to typesetting Event-B code.

### 3.2.1 Defining the Event-B language

We first define the Event-B language using **lstdefinlanguage**.

```

\def\lst@visiblespace{\hspace{0.2em}}
\lstdefinlanguage{Event-B}{%
  basicstyle=\rmfamily\footnotesize,

```

Subsequently, we define the keywords of Event-B and how to typeset them. Note that the keywords are insensitive.

```

keywords={%
  % Keywords for contexts
  context,extends,sets,constants,axioms,theorem,end,%
  % Keywords for machines
  machine,sees,refines,variables,invariants,variant,events,%
},%
keywordstyle=\color{EventB@keywordcolour}\bfseries\sffamily,%
sensitive=false,

```

We also define the secondary keywords of Event-B and how to typeset them.

```

ndkeywords={%
  % Keywords for events
  extended,theorem,any,where,when,with,begin,then%
},%
ndkeywordstyle=\color{EventB@ndkeywordcolour}\bfseries\sffamily,%

```

Next, we define how to typeset Event-B identifiers.

```

identifierstyle=\color{EventB@identifiercolour}\sffamily,

```

We define how comments are typeset.

```
comment=[l]{//},%
morecomment=[s]{/*}{*/},%
commentstyle=\color{EventB@commentcolour}\rmfamily,%
```

Furthermore, we define the appearance of formulae (which are typeset strings).

```
stringstyle=\color{EventB@formulacolour}\sffamily,
string=[b]",
showstringspaces=true, % Do not show the space in formulae
```

Finally, we define the Event-B mathematical symbols using the **bsymb** package as follows.

```
inputencoding=utf8, % Allow UTF-8 input encoding
extendedchars=true, % Use extended characters
literate= % Event-B mathematical symbols
% Short sequences should appear before long sequences containing them
% Predicates
{}{{\bfalse$}}1% False
{}{{\btrue$}}1% True
{}{{\land$}}1% Conjunction
{}{{\lor$}}1% Disjunction
{}{{\limp$}}1% Implication
{}{{\leqv$}}1% Equivalence
{}{{\lnot$}}1% Negation
{}{{\forall$}}1% Universal quantification
{}{{\exists$}}1% Existential quantification
{}{{\qdot$}}1% Quantification dot
{=}{\=$}}1% Equality
{}{{\neq$}}1% Inequality
% Sets
{}{{\forall$}}1% Universal quantification (This is moved here from ASCII perdicates)
{}{{\emptyset$}}1% Empty set
{}{{\mid$}}1% Vertical bar, e.g., in set comprehension
{}{{\bunion$}}1% Union
{}{{\binter$}}1% Intersection
{}{{\setminus$}}1% Set difference
{}{{\mapsto$}}1% Ordered pair
{}{{\cprod$}}1% Cartesian product
{}{{\pow$}}1% Powerset
{}{{\pown$}}1% Non-empty subsets
{}{{\card$}}1% Cardinality
{}{{\union$}}1% Generalised union
{}{{\inter$}}1% Generalised intersection
{}{{\Union$}}1% Quantified union
{}{{\Inter$}}1% Quantified intersection
% Set predicates
{}{{\in$}}1% Set membership
{}{{\notin$}}1% Set non-membership
{}{{\subseteq$}}1% Subset
{}{{\nsubseteq$}}1% Not a subset
{}{{\subset$}}1% Proper subset
{}{{\nsupset$}}1% Not a proper subset
{}{{\finite$}}1% Finite set
{}{{\partition$}}1% Partition
% Bool and bool
```

```

{!BOOL}{{{\$Bool$}}1% BOOL set
{!TRUE}{{{\$True$}}1% TRUE
{!FALSE}{{{\$False$}}1% FALSE
{!bool}{{{\$bool$}}1% bool predicate set
% Numbers
{{{\$intg$}}1% Set of integer numbers
{{{\$nat$}}1% Set of natural numbers
{1}{{{\$natn$}}1% Set of positive natural numbers
{!min}{{{\$min$}}1% Minimum
{!max}{{{\$max$}}1% Maximum
{+}{{{\$+$}}1% Sum
{-}{{{\$-$}}1% Difference
{{{\$times$}}1% Product
{{{\$div$}}1% Quotient
{!mod}{{{\$textrm{mod}$}}1% Remainder
{{{\$upto$}}1% Interval
% Number predicates
{{{\$geq$}}1% Greater or equal
{{{\$leq$}}1% Less or equal
% Relations
{{{\$rel$}}1% Relations
{!dom}{{{\$dom$}}1% Domain
{!ran}{{{\$ran$}}1% Range
{{{\$trel$}}1% Total relations
{{{\$srel$}}1% Surjective relations
{{{\$strel$}}1% Total surjective relations
{{{\$circ$}}1% Backward composition
{!id}{{{\$id$}}1% Identity
{{{\$domres$}}1% Domain restriction
{{{\$domsub$}}1% Domain subtraction
{{{\$ranres$}}1% Range restriction
{{{\$ransub$}}1% Range subtraction
{{{\$^{-1}$}}1% Inverse
{{{\$ovl$}}1% Overriding
{{{\$dprod$}}1% Direct product
{{{\$pprod$}}1% Parallel product
{!prj1}{{{\$prjone$}}1% First projection
{!prj2}{{{\$prjtwo$}}1% Second projection
% Functions
{{{\$pfun$}}1% Partial functions
{{{\$tfun$}}1% Total functions
{{{\$pinj$}}1% Partial injections
{{{\$tinj$}}1% Total injections
{{{\$psur$}}1% Partial surjections
{{{\$tsur$}}1% Total surjections
{{{\$tbij$}}1% Bijections
{{{\$lambda$}}1% Lambda abstraction
% Assignment
{{{\$bcmeq$}}1% Becomes equal to
{:}{{{\$bcm$}}1% Choice from a set
{:}{{{\$bcm$}}1% Choice by predicate
% ASCII Number predicates (This has to be before ASCII Predicates)
{>}{{{\$>$}}1% Greater
{<}{{{\$<$}}1% Less

```

```

{>=}{\geq$}1% Greater or equal
{<=}{\leq$}1% Less or equal
% ASCII Predicates
{!false}{\bfalse$}1% False
{!true}{\btrue$}1% True
{&}{\land$}1% Conjunction (note the backslash)
{!or}{\lor$}1% Disjunction
{=>}{\limp$}1% Implication
{<=>}{\leqv$}1% Equivalence
{!not}{\lnot$}1% Negation
{!#}{\exists$}1% Existential quantification (note the backslash)
{!.}{\qdot$}1% Quantification dot
{/=}{\neq$}1% Inequality
% ASCII Sets
{\{\}}{\emptyset$}1% Empty set (note the backslashes)
{\|}{\mid$}1% Vertical bar, e.g., in set comprehension (not the backslash)
{\setminus}{\setminus$}1% Difference
{\|/}{\bunion$}1% Union
{\/\|}{\binter$}1% Intersection
{|->}{\mapsto$}1% Ordered pair
{**}{\cprod$}1% Cartesian product
{!POW}{\pow$}1% Powerset
{!POW1}{\pown$}1% Non-empty subsets
{!UNION}{\Union$}1% Quantified union
% ASCII Set predicates
{/:}{\notin$}1% Set non-membership
{/<:}{\not\subsepeq$}1% Not a subset
{/<<:}{\not\subset$}1% Not a proper subset
{<<:}{\subset$}1% Proper subset
{<:}{\subsepeq$}1% Subset
{!:}{\in$}1% Set membership
% ASCII Numbers
{!INT}{\intg$}1% Set of integer numbers
{!INTER}{\Inter$}1% Quantified intersection (This is moved here from ASCII Sets)
{!NAT}{\nat$}1% Set of natural numbers
{!NAT1}{\natn$}1% Set of positive natural numbers
{-}{\-$}1% Difference
{!*}{\times$}1% Product (Note the !)
{!/}{\div$}1% Quotient (Note the !)
{..}{\upto$}1% Interval
% ASCII Relations
{<->}{\rel$}1% Relations
{<<->}{\trel$}1% Total relations
{<->>}{\srel$}1% Surjective relations
{<<->>}{\strel$}1% Total surjective relations
{!circ}{\circ$}1% Backward composition
{<|}{\domres$}1% Domain restriction
{<<|}{\domsub$}1% Domain subtraction
{|>}{\ranres$}1% Range restriction
{|>>}{\ransub$}1% Range subtraction
{~}{\^{-1}$}1% Inverse
{<+}{\ovl$}1% Overriding
{><}{\dprod$}1% Direct product
{||}{\pprod$}1% Parallel product

```

```

% ASCII Functions
{+>}{{\pfun$}}1% Partial functions
{-->}{{\tfun$}}1% Total functions
{>+>}{{\pinj$}}1% Partial injections
{>->}{{\tinj$}}1% Total injections
{+>>}{{\psur$}}1% Partial surjections
{->>}{{\tsur$}}1% Total surjections
{>->>}{{\tbij$}}1% Bijections
{\%}{{\lambda$}}1% Lambda abstraction
% ASCII Assignment
{:=}{{\bcmeq$}}1% Becomes equal to
{:}{{\bcmin$}}1% Choice from a set
{:|}{{\bcmsuch$}}1% Choice by predicate
, % End of Event-B mathematical symbols
}

```

### 3.2.2 Typesetting Event-B Code

We first create a short inline Event-B code with `|` using `\lstMakeShortInline` command.

```
\lstMakeShortInline[language=Event-B, breaklines=f, basicstyle=\rmfamily\normalsize]|
```

We then create a dedicated `EventBcode` environment using `\lstnewenvironment`.

```
\lstnewenvironment{EventBcode}{\lstset{language=Event-B}}{}
```

We then create a dedicated `EventBNoInline` environment using `\newenvironment`.

```

\newenvironment{EventBNoShortInline}
{
  \lstDeleteShortInline|
  \lstset{language=Event-B, breaklines=f, basicstyle=\rmfamily\normalsize}
}
{\lstMakeShortInline[language=Event-B, breaklines=f, basicstyle=\rmfamily\normalsize]}

```

Finally, we set some appearance parameters for display the code.

```

\lstset{%
  columns=fullflexible, % The columns are fully flexible.
  numberbychapter=false,
  frame=top,frame=bottom, % There are line (frame at top and bottom).
  stepnumber=1, % the step between two line-numbers. If it is 1 each line will be numbered
  numberstyle=\tiny,
  numbersep=5pt, % how far the line-numbers are from the code
  tabsize=2, % tab size in blank spaces
  breaklines=true, % sets automatic line breaking
  captionpos=b, % sets the caption-position to top
  mathescape=false,
  showspaces=true, % Do not show spaces
  showtabs=false, % Do not show tabs
  xleftmargin=10pt,
  framexleftmargin=10pt,
  framexrightmargin=0pt,
  framexbottommargin=5pt,
  framextopmargin=5pt,
  escapechar=\%,
  numbers=left, % where to put the line-numbers; possible values are (none, left, right)
  numbersep=5pt,

```



```

}
\newcommand{\EventBinputlisting}[2][1]{%
  \begin{mdframed}[backgroundcolor=yellow!10, rightline=false, leftline=false]
    \lstinputlisting[language=Event-B, mathescape, frame={}, #1]{#2}
  \end{mdframed}
}

```

Event@SetKeywordColour

```

\let\EventBSetKeywordColour\EventB@SetKeywordColour

\newcommand{\EventB}{Event-B\xspace}

```

## Change History

v0.1		combinations (Issue #3) . . . . .	1
	General: Initial version . . . . .	1	v1.0
v0.2	General: Added support for ASCII	Event@SetKeywordColour: Added . . .	9