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>

June 6, 2017

Abstract

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

Contents

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 v0.1, dated 2017/08/10.

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}
```

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 `EventB@SetKeywordColour` is used to set the colour of the Event-B keywords, by default, it is set to `black`.

```
\newcommand{\EventB@SetKeywordColour}[1]{%
  \colorlet{EventB@keywordcolour}{#1}%
}
\EventB@SetKeywordColour{black}
```

`EventB@SetNdKeywordColour` Command `EventB@SetNdKeywordColour` is used to set the colour of the secondary Event-B keywords, by default, it is set to `black`.

```
\newcommand{\EventB@SetNdKeywordColour}[1]{%
  \colorlet{EventB@ndkeywordcolour}{#1}%
}
\EventB@SetNdKeywordColour{black}
```

`EventB@SetIdentifierColour` Command `EventB@SetIdentifierColour` is used to set the colour of Event-B identifiers, by default, it is set to `black`.

```
\newcommand{\EventB@SetIdentifierColour}[1]{%
  \colorlet{EventB@identifiercolour}{#1}%
}
\EventB@SetIdentifierColour{black}
```

`EventB@SetCommentColour` Command `EventB@SetCommentColour` is used to set the colour of Event-B comments, by default, it is set to `black`.

```
\newcommand{\EventB@SetCommentColour}[1]{%
  \colorlet{EventB@commentcolour}{#1}%
}
\EventB@SetCommentColour{black}
```

`EventB@SetFormulaColour` Command `EventB@SetFormulaColour` is used to set the colour of Event-B formulae, by default, it is set to `black`.

```
\newcommand{\EventB@SetFormulaColour}[1]{%
  \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`. The formula colour is `blue`.

```
\DeclareOption{colour}{
  \EventB@SetKeywordColour{red}
  \EventB@SetNdKeywordColour{red}
  \EventB@SetIdentifierColour{purple}
  \EventB@SetCommentColour{green}
  \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 `lstdefinelanguage`.

```
\lstdefinelanguage{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}\bf\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}\bf\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=false, % 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
{}{{$\bfalse$}}1%
{}{{$\btrue$}}1%
{}{{$\land$}}1%
{}{{$\lor$}}1%
{}{{$\limp$}}1%
{}{{$\leqv$}}1%
{}{{$\lnot$}}1%
{}{{$\forall$}}1%
{}{{$\exists$}}1%
{}{{$\qdot$}}1%
{}{{$\neq$}}1%
{}{{$\emptyset$}}1%
{}{{$\bunion$}}1%
{}{{$\binter$}}1%
{}{{$\setminus$}}1%
{}{{$\mapsto$ }}1%
{}{{$\cprod$ }}1%
{}{{$\pow$ }}1%
{1}{{$\pown$ }}1%
{}{{$\in$}{ }}2%
{}{{$\notin$ }}1%
{}{{$\subseteq$}}1%
{}{{$\nsubseteq$ }}1%
{}{{$\subset$ }}1%
{}{{$\nsubset$ }}1%
{}{{$\intg$ }}1%
{}{{$\nat$}}1%
{1}{{$\natn$ }}1%
{}{{$\geq$ }}1%
{}{{$\leq$ }}1%
{}{{$\rel$ }}1%
{}{{$\circ$ }}1%
{}{{$\domres$ }}1%
{}{{$\domsub$}}1%
{}{{$\ranres$ }}1%
{}{{$\ransub$ }}1%
{}{{$\sim$}}1%
{}{{$\ovl$ }}1%
{}{{$\dprod$ }}1%
{}{{$\pprod$ }}1%
```

```
  {}{{$\pfun$ }}1%
  {}{{$\tfun$}}1%
  {}{{$\pinj$ }}1%
  {}{{$\tinj$ }}1%
  {}{{$\psur$ }}1%
  {}{{$\tsur$ }}1%
  {}{{$\tbij$ }}1%
  {}{{$\lambda$ }}1%
  {}{{$\bcmeq$}{ }}2%
  {:}{{$\bcmin$}{ }}2%
  {:}{{$\bcmsuch$}{ }}2%
  , % 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}}{}
```

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=false, % 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,
}

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

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

Change History

v0.1

General: Initial version 1

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the definition; numbers in roman refer to the pages where the entry is used.