

Simple L^AT_EX Documents

R. Banach
School of Computer Science,
University of Manchester,
Manchester, M13 9PL, U.K.

The L^AT_EX system can produce the most elaborate scientific documents you can imagine. Here are a few basics which are sufficient for the reports needed in CS60110. Such reports can be produced by editing the `Template.tex` source mentioned in the exercises and downloadable from the CS60110 web site.

Basics: Open `Template.tex` in a text editor, with the corresponding `Template.pdf` to hand for comparison. Ignore everything from the beginning till you see the comment block saying:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%%  
%%          Edit the stuff below here !!          %%  
%%  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Next you see the `\title{...}` command, the contents of which you can edit to create your own title. You also see the `\author{...}` command, and can edit this as well. Note that it also contains address and email information, and you can edit these too. Observe that `\\` generates a newline (and `\newpage` does what you'd expect). Note that the email information is set in teletype font via the `\texttt{...}` command. You will have gathered that in L^AT_EX, more or less everything is bracketed/nested using `{` and `}` brackets, and commands always start with `\`. This is correct.

Next you see the *Abstract* opener (`\begin{abstract}`), *Abstract* body text, and *Abstract* closer (`\end{abstract}`). Make use of this if you need to.

Next, the document proper starts. You see a section heading `\section{Introduction}`... so that's how you do those.

Body text is just typed, as you'd expect. There are two paragraphs. Paragraphs are separated by a blank line.

Next, you see a subsection heading `\subsection{subIntroduction}`... so that's how you do those. Then there are another seven paragraphs. Then there are many consecutive blank lines in the source, but you see that they have no effect on the output.

The next paragraph contains examples of style variations: something in *italics*, using `\textit{...}`, something **bold**, using `\textbf{...}`, something in upright font, using `\textsf{...}`, something in teletype font, using `\texttt{...}`, and two simple maths formulae using `$. . . $`. The italics, bold and font attributes may be arbitrarily combined, simply by nesting the relevant commands. Unfortunately the ascii strings for maths symbols in Event-B AMN and L^AT_EX are different, so you will have to consult the web for the latter if you need to — though hopefully you won't. If you do, *google* 'latex maths symbols' to access <http://web.ift.uib.no/Fysisk/Teori/KURS/WRK/TeX/symALL.html>, which is shadowed on the CS60110 web site. There is unimaginably more to setting fancy maths in L^AT_EX, but that's plenty.

To highlight 'Here is how you write a requirement ...' some vertical space has been added via `\vspace{2ex}`. Here `ex` is a unit of length, and `2` is a multiplier. The command `\hspace{-2ex}` would do a similar job for horizontal spacing (in this case showing that the amount can be negative too).

Formal requirements, formatted in a manner similar to that used in the course slides/notes are created using the command `\REQUIREMENT{...}{...}` which takes two parameters. The first is the text of the requirement, and the second is its label. Note, in the first example in `Template.tex`, the presence of the `\noindent` to suppress a paragraph break caused by the newline between the `{` and the start of the requirement text. If the text started right after the `{` the `\noindent` would be unnecessary, as in the second example.

Final point for general source text: the special characters `#` `$` `&` `_` `%` `{` `}` `~` `\` are interpreted as commands, so cannot be typed directly. They have to be escaped: `\#` `\$` `\&` `_` `\%` `\{` `\}`. The tilde character `~` cannot be escaped in that way. Whatever you do, beware of including the character `\` in your text!

A useful general workaround if you can't get something to come out properly is via L^AT_EX's verbatim facilities, which cause the source stream to be copied to the output in teletype font, without any processing. So if something will not come out properly but you know how to make it look approximately OK in plain ascii, you can put `\verb+...+` into the source stream. This causes the content of the command, the `'...'`, to be reproduced exactly. In this, `\verb` is the command name, and the `+` is any character chosen by you that does not occur in the `'...'`, its two occurrences thus acting as a pair of matching brackets. The `\verb` facility is the most rapid way of getting a special character into the output stream if you don't know any better means of generating it in the current font you are using. It works for the tilde and backslash.

The inline `\verb` command has an extended paragraph oriented counterpart (good for program fragments etc.):

```
\begin{verbatim}
...
...
\end{verbatim}
```

Rodin components: The point of using L^AT_EX at all is to be able to include L^AT_EX source generated by Rodin for contexts and machines. This can be inserted into the `Template.tex` source as follows.

The `Template.tex` source already contains the opening and closing declarations that are produced by Rodin, so these have to be stripped out from the Rodin L^AT_EX source. To do this open such a source and delete from the beginning up to and including the lines that say:

```
%-----
\begin{document}
\thispagestyle{empty}
```

The first line of the file should now read:

```
\begin{description}
```

[N.B. If you don't want the 'component title in a box' that Rodin produces by default at the top of the component source, look just below for the line:

```
\Btitle{<componentname>}{<date>}{<time>}
```

and delete it.]

Now go to the end and delete the

```
\end{document}
```

line too. The content of the file as a whole can now be inserted into the `Template.tex` source in place of a body paragraph.

Screenshots: Rodin screenshots and other graphics, provided they are in `.png` format (the `.png` suffix must be in lower case), can be incorporated via the `\includegraphics[...]{...}` command, illustrated in Section 3 of `Template.tex`. The first parameter `[width=1.0\linewidth]` is self-explanatory, and the second is the filename.¹

Creating pdf: Finally, the source you have created can be compiled via:

```
pdflatex Template.tex
```

typed from the command line in `linux`. It produces `Template.pdf`. It's usually a good policy to compile small changes to the source and to check the resulting pdf, `Template.pdf`, relatively frequently, as L^AT_EX error messages can be somewhat uninformative.

1. N.B. The `'1.0'` is the default and could be omitted. I left it in to indicate how to write multipliers other than `1.0`.