An Eight Point Intoduction to LaTeX

Calvin Houser

Beginning

The traditional word processor puts the formation of written content in opposition with the content's formatting. The effort spent in articulation is compounded by an unneeded focus on layout revision. As a document is assembled, one must reconcile the maunal placement of objects, figures, and text with the algorithmic document formatting procedures governing text flow. These procedures must be adjusted for every document but are represented in a seperate interface of buttons, menus, and windows. Nonlinear editing of a document typically requires examining all succeeding parts of the document to fix any new formatting errors.

LaTeXprovides a system for document preparation which puts document-specific formatting commands directly in the file text. As explained on latex-project.org, LaTeXis not a word processor, but a markup language[2]. Commands are written alongside document content, which are interpreted to guide document formatting.

For instance, this document is formatted using the `article` document class, using the command `\documentclass[11pt,twocolumn]{article}` which also specifies the font size and the two-column text formatting. The title is created using the command `\title{An Eight Point Introduction to \LaTeX}` which also includes the `\LaTeX` command, specifying use of the LaTeX logo symbol. Each one of these commands are written out in the document file, in the same window as this text. The commands in this paragraph are specified to be interpreted as printed text by the `\verb!\LaTeX!` command.

This introduction will familiarize you with everything necessary to start writing in LaTeX, and function as a reference for commonly used features of the language.

1 Document Classes

The base of any LaTeXdocument is its document class. The document class specifies a collection of formatting standards applied to the document. The command `\documentclass[options]{class}` is used at the very top of the document, with the type of document specified in curly braces. These types are contained in class files, designated by the .cls extension. One can write their own class file, but there are a variety which come standard with LaTeX. These include:

- `article` - Designed for scientific publications, short reports, and documentation.
- `letter` - A traditional US letter form, provides commands for typical letter fields.
- `beamer` - A set of commands for creating slide presentations

The `options` field is optional and can be used for specifying any additional parameters important to the layout of the document. The options for this document are `11pt`, specifying an 11pt font size; and `twocolumn`, which specifies that the document should be typeset in two columns. Other options specify paper type, landscape formatting, and whether to add a title page. The exact options available will depend on the document class being used. As seen here, when multiple options are included, they are seperated by a comma.

2 Packages

Much of what LaTeXcan do is distributed into packages, or collections of commands for customizing parts of your document. Many come with the basic LaTeX distribution, including[4]:

- `graphicx` - Allows graphics to be loaded in from file.
- `mathtools` - A collection of packages for a broad selection of mathmematical typesetting applications including symbols, options for equation numbering, and matrix formatting.
- `setspace` - Provides options for changing standard line spacing within a document.
- `url` - Contains a command for specifying URLs, and manages the special characters that can be found in them automatically.
- `xcolor` - Gives support for colored text.

Each package is included using the command `\usepackage[options]{package}` which references a file with the .sty extension. The again optional square brackets can include parameters for the implementation of the package. Like document classes these can also be written by scratch or provided by a third party, but unlike document classes any number of them can be used in a document.

3 Titling

Each LaTeXdocument has fields for the title, author, and date. These properties are each specified using commands such as `\title{ }`, `\author{ }`, and `\date{ }`. When a document has multiple authors, using the command `\and` between each name formats each authors' names individually. When the date command is omitted, the current date is printed. Left blank, no date is printed. A subtitle is optionally specified using the `\subtitle{ }` command, and `\thanks{footnote text}` can be used inline within these commands to include a footnote reference at the bottom of the title page.

Each of these commands is placed before the document body, which is specified using the commands `\begin{document}` and `\end{document}`. Between the document commands, the title information is placed using the command `\maketitle`. If the `\thanks{ }` command was used, the `\blindtext` command should also be added here to insert the footnote.

In scientific publication, there of often many authors contributing to a single paper. Each of these authors may have seperate affiliations, which should be printed next to their names. Typically authors' affiliations are specified using the form `\author{name \\ affiliation}`, where the `\\` specifies a newline. One would then specify multiple authors using the form `\author{name1 \\ affiliation1 \and name2 \\ affiliation2 \and ...}`

4 Subdividing Text

As a document grows in length, it becomes important to break the content into meaningful sections. LaTeXprovides several levels of this organization. For instance, within the `article` document class, there are six levels of division with titling commands.

1. `\section{Section Title}`
2. `\subsection{Subsection Title}`
3. `\subsubsection{Subsubsection Title}`
4. `\paragraph{Paragraph Title}`
5. `\subparagraph{Subparagraph Title}`

There is additionally a `\part{Part Title}` command, which has a depth of `-1`. Level 0 is reserved for `\chapter{ }`, which is only in book and report document classes. Each command provides formatting for a title, including an optional numbering system. Using a command without a title can still invoke a section number, and paragraphs can be specified without any titling, as they are in this document, by just inserting an extra blank line between each paragraph.

```
line in first paragaph
line which still is in first paragraph

line in second paragraph
```

LaTeXnumbers each section chronologically, using a decimal to denotate each subsection level. By default LaTeXdoesn't number the paragraphs, but this can be changed using the command `\setcounter{secnumdepth}{int}` where `int` represents the new depth of the numbering. The default value for instance is `3`. Parts are numbered using roman numerals by default. Any sectioning command can be specifically invoked without a number using the command with an asterisk. For example, `\section*{Beginning}` is the command designating the first section of this document.

5 Environments

An indispensible function of most word processors is the ability to format data and other information into tables and lists. These are each environments for content in which special formatting rules may be invoked; and in a word processor, transitioning in and out of these environments is often an uncertain task, defined implicitly by cursor selection.

In LaTeX, each environment is defined using the block command:

```
\begin{environment}!
...
\end{environment}
```

where `environment` specifies whichever ruleset is being invoked. Some of the commonly useful environments in standard LaTeXare described below.

5.1 `itemize` and `eunmerate`

Lists in LaTeXcan take two forms, bulleted or numbered. Each of these are handled in seperate environments, `itemize` and `eunmerate`. The items in each list both use the command `\item` to denote an object on the list. After this command, all text will be indented after the bullet point or number until the next `\item` command or the `\end{environment}` command. The first list used in this reference was formatted using the following text.

```
\begin{itemize}
  \item \verb!article! - Designed for s...
  \item \verb!letter! - A traditional U...
```

5.2 tabular

Tables in LaTeXare typically represented using the `tabular` environment. A table is declared in this way with the command `\begin{tabular}{table spec}`. the `table spec` argument takes an arbitrarily long set of characters, separated by a space, to describe the columns of the table.

char	column type
`l`	left-justified cells
`c`	center-justified cells
`r`	right-justified cells
`p{'width'}`	justified according to the paragraph formatting style with a fixed width set by any standard LaTeXlength unit.

These columns will by default be separated only by white space, but a dividing line can be inserted between them by inserting a `|` between two columns. `||` designates a double dividing line. The next table in this reference, for instance, will be declared using the expression `\begin{tabular}{ l | l | l }` which will create three columns, each with left justified text, and separated by a single dividing line.

Within the `tabular` environment, each row is separated by a newline `\\` command, and each cell is separated by an ampersand. The first few rows of the table above were made using the following text.

```
\begin{tabular}{ l p{1.75in} }
  char & column type \\
  \hline
  ...
```

This example also highlights the `\hline` command, which is used to insert a horizontal line between rows. A line which only divides the rows of some columns can be inserted using `\cline{i-j}`, where `i` and `j` are both integers representing the starting and ending columns for the line.

5.3 verbatim

The environment used most prevalently in this document and many other informative texts concerning software is the `verbatim` environment. This simple command creates a block of text formatted to resemble a console prompt or text editor and is often used to designate text which is meant to be run or output by a computer. Because of this, any text within the verbatim environment will not be executed as LaTeXcommands or formatted beyond what is seen in the editor. This is how earlier examples of LaTeXcommands were formatted into this document.

```
\begin{verbatim}
  \begin{tabular}{ l p{1.75in} }
    char & column type \\
    ...
```

It is important to remember with this environment that because the enclosed text is not formatted, a particularly long line of text can extend out of formatted text columns and off the page.

6 Mathematical Forumlas

A defining aspect of LaTeXis the mathematics environment, which contains a simple shorthand for expressing mathematical symbols and equations. Though it can be invoked similarly to the

environments described in the last section, there is a convenient shorter notation:

- `\( ... \)` - inline expression
- `\[ ... \]` - block expression

The associated environment `equation` provides the same commands with an automatic sequential numbering of each equation. This is invoked like the standard environments with `\begin{equation}` and `\end{equation}`. An equation can be excluded from the numbering by including an asterisk after `equation`.

The following is a brief list of some typesetting control commands within the math environment.

- `{ab}` $\rightarrow ab$ - group characters within the math environment
- `a_b` $\rightarrow a_b$ subscript, for indicies
- `a^b` $\rightarrow a^b$ superscript, for exponents
- `\frac{a}{b}` $\rightarrow \frac{a}{b}$ fraction notation
- `\sqrt[b]{a}` $\rightarrow \sqrt[b]{a}$ root notation, the [ ] argument optionally denotes a nonsquare root.

Common mathematical functions or symbols are each invoked using a `\` command. The list of supported symbols is extensive, with a list in the end references of this text[1]. Most follow an intuitive construction, such as `\sum` and `\int` for $\sum$ and $\int$.

This is only an introduction, the flexibility of the math environment far exceeds the scope of this guide. Options exist for manual control of sizing and spacing, font styles, and every commonly used symbol accent.

7 User-Defined Macros

Through the last few sections, this tutorial has demonstrated the ease of implementing formatting structures through LaTeXcommands, and the flexibility with which one can do so. Macros, or user-defined commands, allow the most complex parts of LaTeXto be managed simply.

Suppose within an examination of CPU architecture, an author wishes to reference the state of various registers. Putting this information into a 2-cell table could separate it from the text and improve readability. One such implementation is:

eax	0x00000000

```
\begin{tabular}{| c | c |} \\
  \cline{1-2}
  \texttt{eax} & \texttt{0x00000000} \\
  \cline{1-2}
\end{tabular}
```

It isn't too complicated, but many similar tables will have to be created and the only unique elements of this particular one are `eax` and `0x00000000`. Further, if the author decided to change any part of this address formatting later, they would need to go back and adjust each example. A better way to implement this would be within a macro:

```
\newcommand{\register}[2]{
  \begin{tabular}{| c | c |} \\
    \cline{1-2}
    \texttt{#1} & texttt{#2} \\
    \cline{1-2}
  \end{tabular}
}
```

`\newcommand` declares a new command with two necessary arguments. First, a name through

which to invoke it (in this case `\register`.) Second, the text and commands which are to replace it. An optional digit enclosed in brackets designates a number of parameters for the command to take in as arguments. The arguments are then referenced by a hash sign and the corresponding digit. In this example, the command takes two arguments and places each in the cell of a table. The above code is placed before the document body. A register can then be invoked through the `\register{adr}{REG_STATE}` command.

7.1 Using Counters

Though one does not need to write macros in order to use counters, counters are a necessary part of many powerful macros. As shown previously in section 4, LaTeXhas a framework for automatic sequential numbering which is already implemented in section titles. This framework can be further used within macros for additional sequential labeling. Suppose an author wanted to number sentences within a paragraph, perhaps for later reference. This is a possible implementation:

```
\newcounter{sentencenumber}[paragraph]
\newcommand{\sentence}{
  \stepcounter{sentencenumber}
  (\thesentencenumber):
}
```

The command `\newcounter` declares a new counter with the name `sentencenumber`. The optional parameter in brackets specifies another counter, in this case `paragraph`, which will cause `sentencenumber` to reset whenever it is incremented[3].

(1): As a demonstration, each sentence in this paragraph is numbered by this command. (2): The name, `\sentence`, is merely added to the beginning of each sentence. (3): Each use of the command increments the `sentencenumber` counter before inserting its value into the text.

7.2 Generating Graphics

To demonstrate the power of macros, the following example uses counters and a parameter-defined environment along with two packages to draw an analog clock face, set to an arbitrary time, on the page.

```
\usepackage{tikz}%for drawing commands
\usepackage{calc}%for infix expressions
% macro \aclock{clock time} generates an
% analog clock face displaying the input
% clock time. input form: HMM or HHMM
% where each H or M is a digit of hours
% or minutes.
\newcounter{hour}%init counters for
\newcounter{minute}%each value ref'd
\newcounter{degree}%by tikz
\newcommand{\aclock}[1]{
  \setcounter{hour}{#1 / 100}
  \setcounter{minute}
    {#1 - {\thehour * 100}}
  \setcounter{degree}%set hour hand degr
    {90-{30* \thehour}-{\theminute /2}}
  \begin{tikzpicture}[scale = 20pt]
    \draw (0.5pt,0.5pt)
      circle [radius=.5pt];%draw face
    \draw (0.5pt,0.5pt) %draw hour hand
      -- ++(\thedegree:0.3pt);
    \setcounter{degree}{90 %set minute
      - {6 * \theminute}} %hand degree
    \draw (0.5pt,0.5pt)%draw min hand
      -- ++(\thedegree:0.4pt);
    \draw (0.5pt,0.5pt)%draw center
      circle [radius=0.03pt];
  \end{tikzpicture}
}
```

Suppose an author is writing a novel in which passages are entries in a journal. An imporant aspect of the story could be the time each entry is taken. This could be neatly represented by embedding a small clock in the header of each entry. As specified, one only needs write `\aclock{530}` to draw such a clock face with the time 5:30.

The exact process of this macro is too long to explain within this document, but one can see its utility. A macro can simplify a list of vector graphic instructions into a single command and value.

8 Bibliography and Citation

Though often an afterthought, the most important part of a document is often the material referenced in its creation. A good author will always catalog their sources, and ideally place markers for appropriate citation as they write. LaTeXoffers many capabilities for managing this, some of which will be explained here.

In LaTeX, the author assigns a `cite_key` or name to each source, and invokes a citation with the `\cite{cite_key}` command. By default, this will insert a numerical citation which will correspond to the number given to that source in the bibliography. Additonal formats can be specified through the bibliography style.

The simpler bibliographies can be embedded directly into the document, right before the `\end{document}` command, by invoking the `thebibliography` environment.

```
\begin{thebibliography}{9}

\bibitem{cite_key}
  FirstName LastName,
  \textit{Source Title},
  Publisher, Date.

\end{thebibliography}
```

LaTeXwill take each `\bibitem` entry and order them according to author name with a reference label. These labels are numerical by default, but different styles will use different labels. The environment parameter specifies the number of digits to reserve for entry numbering. Any single digit (here a `{9}`) will allow for 9 entries, and any two digits would allow for 99 entries.

A larger bibliography however, is best separated into a separate file for independent management. The standard way to do this in LaTeXis with an auxillary program called BibTeX. BibTeX comes with most LaTeXdistributions and compiles a separate `.bib` file containing the information for each source. This resulting bibliography is then appended to the LaTeXdocument by including the following commands in place of the previous example:

```
\bibliographystyle{stylename}
\bibliography{bibfile}
```

`bibfile` refers to a `.bib` file included in the same directory as the document, and `stylename` specifies a style for the bibliography included in either the LaTeXdistribution or a package. Some common styles are `plain`, which uses numerical labels, and `alpha` which uses the `cite_key` as the printed label. Both sort entries alphabetically by author name.

BibTeX file structure is well documented. A reference for creating and compiling a `.bib` file is included on the following page[5]; an example implementaion of a `plain` bibliography and the resources used in this tutorial.

References

[1] Scott Pakin. The comprehensive latex symbol list, 2017. https://ctan.org/tex-archive/info/symbols/comprehensive.

[2] The LaTeX Project. An introduction to latex, 2017. https://www.latex-project.org/about/.

[3] ShareLaTeX. Counters, 2018. https://www.sharelatex.com/learn/Counters.

[4] Wikibooks. Latex, January 2018. https://en.wikibooks.org/wiki/LaTeX/.

[5] Wikibooks and Andy Roberts. Bibliography management: Bibtex, 2018. https://en.wikibooks.org/wiki/LaTeX/Bibliography_Management#BibTeX.