How is LaTeX Useful

Will Anderson

May 18, 2018

Contents

List of Figures

Preface

This document will not be talking about the benefits of using open-source software, despite being a strong point of using LaTeX And your own tools, this is because these are not inherent advantages to the actual document writing process, except in the case where you're using another programming language to make something in LaTeX

1 How LaTeX Documents are laid out.

LaTeX Is a combination of written language and code, this can make reading some things more difficult, but when you understand LaTeX They will make much more sense, this guide is going to teach you about how LaTeX Documents are formed and what makes them great.

1.1 Document formatting

There are lots of ways to format your documents, such as title pages and putting content into your headers or footers, when it comes to putting content in your header or footer it is managed by a package, packages are little "Add-Ons" That are used in LaTeX and are how it functions with a lot of the things that I will describe in this document, however these packages will generally be pre-installed depending on the editor that you use to create LaTeX Documents.

1.1.1 What are sections?

Sections are the main part that get added to the table of contents, they are the top level and are used just like Headings in Word, you can call them with this command in your LaTeXDocument

```
1 \section{Insert Section Title Here}
```

Sections are used to seperate large differing subjects, if you want to differ between various points inside of certain subject you will be calling these commands, depending on which you want to use.

```
1 \subsection{Insert Subsection Title Here}
2 \subsubsection{Insert Subsubsection Title Here}
```

1.1.2 The Table Of Contents

The table of contents is the first thing I will talk about when it comes to document formatting since it is the first thing that people will see after they open the title page, also because it is one of the easiest things to talk about. The table of contents is called with this command,

```
1 \tableofcontents
```

Figure 1: Table of Contents Code

this is simple because it does not need to be modified, however it does have customisation options anyway, some of these customization options are as basic as changing the name of the table of contents, or drastic style changes, you can read more about theming the table of contents here, on https://www.sharelatex.com/learn/Table_of_contents

1.1.3 Headers & Footers

Headers and footers are managed by the **fancyhdr** package, it is easy to understand when you see the syntax and have it explained to you, if you refer to the code below which is used in this very document.

```
1 \pagestyle{fancy}
2 \fancyhf{}
3 \rhead{\LaTeX{} - How is it useful?}
4 \lhead{Will Anderson - Barrel}
5 \rfoot{Page \textbf{\thepage\ }of \textbf{\pageref{LastPage}}}
```

Figure 2: Header & Footer syntax for this document

Now this may look difficult but it's easy to explain, really.

The first line is talking about the pagestyle, this simply sets the style to "Fancy" easy stuff, right?

The second line is resets the layout, so that you can add your own stuff, this might be confusing, but it makes sense since the configuration comes after this.

The third line prints the text into the braces on the right side of the header, the fourth line does this as well, but on the left.

The fifth line is used to modify the right footer on your page, in this case it is used to show the page numbering in bold.

1.1.4 List Of Figures & List of Tables

There are two very simple commands to display a list of all of your figures and tables, just like with the table of contents they can be modified the same

way.

```
1 \listoffigures
2 \listoftables
```

These two commands will print every single figure and table, wherever you put them in the document.

2 Documents inside of LaTeX

You can insert different arbitary files inside of LaTeX, the ones that I use the most are

- PDF's
- Images
- CSV's
- Python outputs

For the first two of these you can use the command below, which will allow you to easily insert images and other documents.

```
1  \includegraphics
2
```

3 Running python inside of LaTeX

This is a graph that was generated using python, this is the code used that you use to do so.

```
1 \begin{pycode}
2 from pylab import *
3 from matplotlib2tikz import save as tikz_save
4 x = linspace(0, 10, 101)
5 plot(x, sin(x))
6 xlabel('$x$-axis')
7 ylabel('$y$-axis')
8 tikz_save('fig.tikz',
9         figureheight = '\\figureheight',
10        figurewidth = '\\figurewidth')
11 \end{pycode}
12
```

Figure 3: Python Code

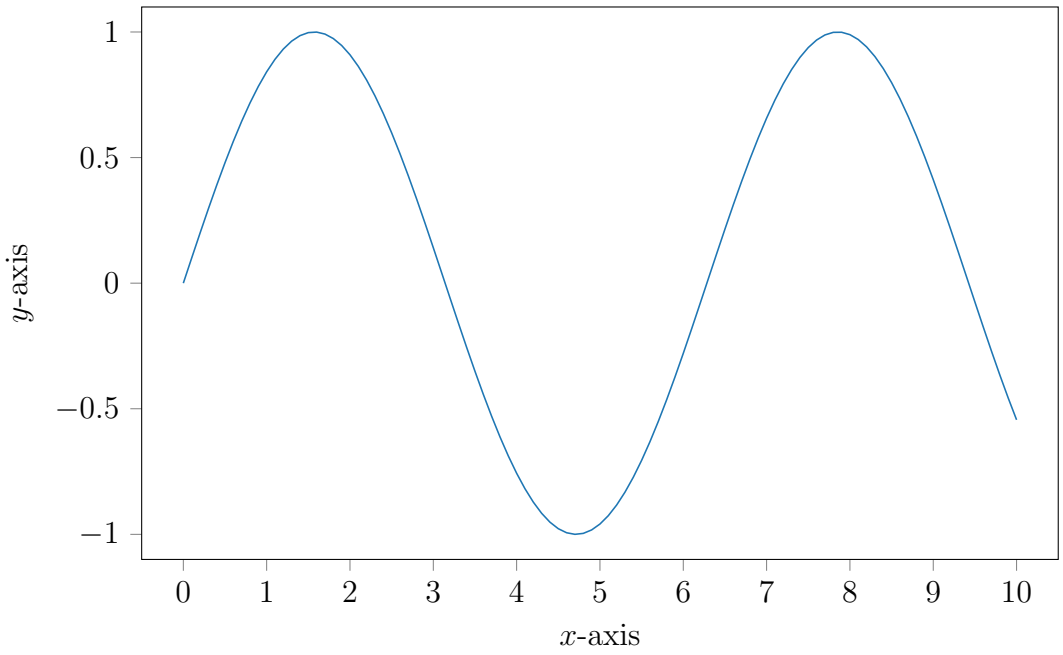

Figure 4: Graph Generated Using Python

3.1 The different packages for python inside of LaTeX

There are two main packages that you will use to run python inside of LaTeX, the first one is the ***python*** package and the second is the ***pythontex*** package.

3.2 Pythontex

Pythontex is used a lot for inline python coding, since the python package does not support it, the package also provides a depythontex utility, that creates a copy of the document in which all Python code has been replaced by its output. This is useful for journal submissions, sharing documents, and conversion to other formats.
This information is taken from CTAN
Another advantage of pythontex over the python package is that Python code is only executed when it has been modified, or when it meets userspecified criteria. Code may be divided into userdefined sessions, which automatically

run in parallel.

3.3 Python

This package isn't really all that well documented and because of this very few people know about the full functionality of it and even fewer will actually use it over pythontex, becuase it doesn't really bring any benefits to the table but can be used in a two second job, for reference even the readme is incredibly simple for it.

```
1 This package enables you to embed Python code (www.python.org)
    in LaTeX and insert the output into your LaTeX document.
2
3 In you LaTeX document insert:
4
5 \usepackage{python}
6 ...
7 \begin{python}[optional.py]
8   print("42")
9 \end{python}
10
11 The file optional.py is included together with the code between
    \begin{python} and \end{python}. Hence, you can extract
    common code to a shared file.
12
13 Compile the document with the shell escape option and of course
    Python installed:
14
15 % latex -shell-escape document.tex
16
17 or insert/change "shell_escape = t" in your texmf.cnf
18
19 If you have any comments, suggestions, or patches please email
    them to me.
20
21 Please note that this material is subject to the GNU General
    Public License as published by the Free Software Foundation;
    either version 2 of the License, or (at your option) any
    later version.
22 See http://www.gnu.org/copyleft/gpl.html for the details of that
     license.
23
24 Martin R. Ehmsen
25 martin@ehmsen.org
```

Yeah, there's so little about this apckage that it's readme is almost entirely comments and licensing.

4 Tables in LaTeX

Tables are very nice in LaTeX as they can be easily manipulated once you grasp the syntax and can be transplanted into any document and will still function, this is incredibly useful when trasferring tables from someone elses work.

- Table
- Longtable
- Tabular

And these are all functionally different, the **table** command doesn't allow for seperators so you're left with a table like this

cell1	cell2	cell3
cell4	cell5	cell6
cell7	cell8	cell9

Figure 5: A **table** table.

The code for this table is listed below.

```
1 \begin{center}
2 \begin{tabular}{ c c c }
3  cell1 & cell2 & cell3 \\
4  cell4 & cell5 & cell6 \\
5  cell7 & cell8 & cell9
6 \end{tabular}
7 \end{center}
```

Figure 6: Code for a **table** table.

Now this isn't all that fancy, after all we are using a supposedly incredibly advanced typesetting language, so what do **longtable** and **tabular** do? Let's find out right now by inserting a table using the tabular environment.

cell1	cell2	cell3
cell4	cell5	cell6
cell7	cell8	cell9

```
\begin{center}
\begin{tabular}{ |c|c|c| }
 \hline
 cell1 & cell2 & cell3 \\
 cell4 & cell5 & cell6 \\
 cell7 & cell8 & cell9 \\
 \hline
\end{tabular}
\end{center}

```

Figure 7: Code for a vertically seperated table

This is much nicer, I can have seperators in my cells so that it is clear which cell belongs which to which, however I don't have it so that they're in a proper grid, however that can easily be solved with a few horizontal lines which can be called like this.

```
\hline
```

cell1	cell2	cell3
cell4	cell5	cell6
cell7	cell8	cell9

```
\begin{center}
\begin{tabular}{ |c|c|c| }
 \hline
 cell1 & cell2 & cell3 \\
 \hline
 cell4 & cell5 & cell6 \\
 \hline
 cell7 & cell8 & cell9 \\
 \hline
\end{tabular}
\end{center}
\caption{Code for the grid layout table}
```

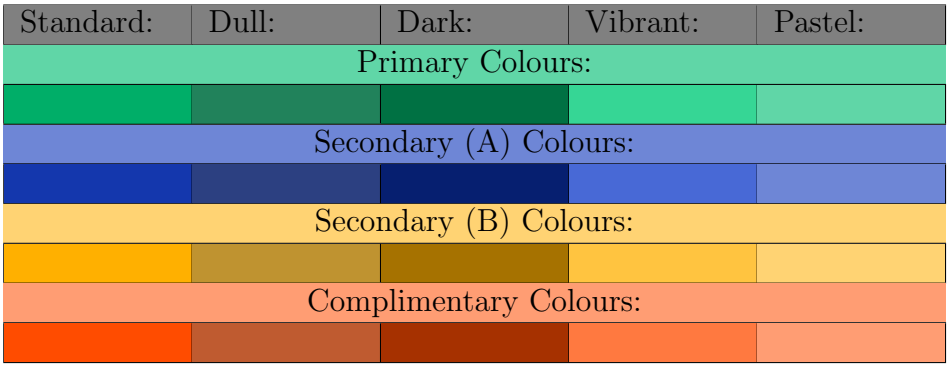

Figure 8: My color table that I used in another project.

Below is the syntax for it, at first it's kind of scary but the moment you read through it you understand it a lot better.

```
1
2  \begin{tabular}{ | m{.15\linewidth} | m{.15\linewidth} | m{.15\linewidth} | m{.15\
     linewidth} | m{.15\linewidth} | }
3          \hline
4   \rowcolor{gray}Standard: & Dull: & Dark: & Vibrant: & Pastel: \\ \hline
5          \rowcolor{P5} \multicolumn{5}{c}{Primary Colours:} \\ \hline
6          \cellcolor{P1} & \cellcolor{P2} & \cellcolor{P3} & \cellcolor{P4} & \cellcolor{P5}
     \\ \hline
7          \rowcolor{A5} \multicolumn{5}{c}{Secondary (A) Colours:} \\ \hline
8          \cellcolor{A1} & \cellcolor{A2} & \cellcolor{A3} & \cellcolor{A4} & \cellcolor{A5}
     \\ \hline
9          \rowcolor{B5} \multicolumn{5}{c}{Secondary (B) Colours:} \\ \hline
10         \cellcolor{B1} & \cellcolor{B2} & \cellcolor{B3} & \cellcolor{B4} & \cellcolor{B5}
     \\ \hline
11         \rowcolor{C5} \multicolumn{5}{c}{Complimentary Colours:} \\ \hline
12         \cellcolor{C1} & \cellcolor{C2} & \cellcolor{C3} & \cellcolor{C4} & \cellcolor{C5}
     \\ \hline
13         \end{tabular}
14
15
```

Figure 9: The Syntax for my Colour Table

5 How I've been inserting code snippets

Now, you may be wondering how I've been inserting these random code snippets, because they're formatted so differently, the answer is simple, really, I'm using the ***lstlisting*** package, it's a package that is used to insert code snippets and can be manipulated extremely easily, it has a long list of languages that it supports, which can be found in the link below.

Code Listing

Sadly very few of the things that you see in the code blocks are not default to lstlisting, however they are managed through a global style which I have pasted below.

```
\lstdefinestyle{mystyle}{
    basicstyle=\footnotesize,
    breakatwhitespace=false,
    breaklines=true,
    captionpos=b,
    keepspaces=true,
    numbers=left,
    numbersep=5pt,
    showspaces=false,
    showstringspaces=false,
    showtabs=false,
    tabsize=2
}

\lstset{style=mystyle}

```

Figure 10: Code Block for theming of code blocks

6 Inputting CSV Files into LaTeX

CSV Files are incredibly easy to manage because of their simplicity, and because of this there happent to be an incredibly simple way of importing them into LaTeX Documents, this is using the package ***csvsimple***

Keyboard	Switch
Happy Hacking Keyboard Professional 2 Type-S JP	Topre
Ducky One	Cherry MX Blue
G80-1800	Cherry MX Black
DZ60	Gateron Black
Steelseries 6GV2	Cherry MX Black
GH60	Cherry MX Black
Corsair K95 RGB Platinum	Cherry MX Blue
Corsair K95 RGB Platinum	Cherry MX Brown
Corsair K95 RGB	Cherry MX Red
Pok3r	MX Clear
Coolermaster Masterkeys Pro M	MX Green
Dell AT102W	ALPS SKCM Black
IBM Model M	Membrane Buckling Spring
IBM Model F	Capacitive Buckling Spring
Razer Blackwidow Chroma	Razer Green
Matias Quiet Pro	Matias Quiet Click
Gherkin	Zealios
Leopold FC660C	Topre
Varmilio VA68M	Modded Cherry MX Black

Figure 11: Barrel's Keyboards (Incomplete)

Now, this may look like there's a lot to go here, seeing as there is a table and as we have shown previously tables usually require a lot of text, however the cvsimple package deals with it easily, here is the code for that entire table.

```
1 \csvautotabular{test.csv}
```

It seems almost too easy to simply get the content on the page, but that is *really* it, if you just want to insert it, you can manage it with the ***csvreader*** command.

7 Presentations

Presentations, the day I realised that I could be doing these in LaTeX Was the day I stopped using Microsoft's Office software, I had virtually zero need for it as the only thing that I Was really lacking was a competent slideshow program, you can use presentations using the **beamer** package, it's incredibly powerful and can easily be themed, it can also be used with markdown if you want to quickly make a presentation.

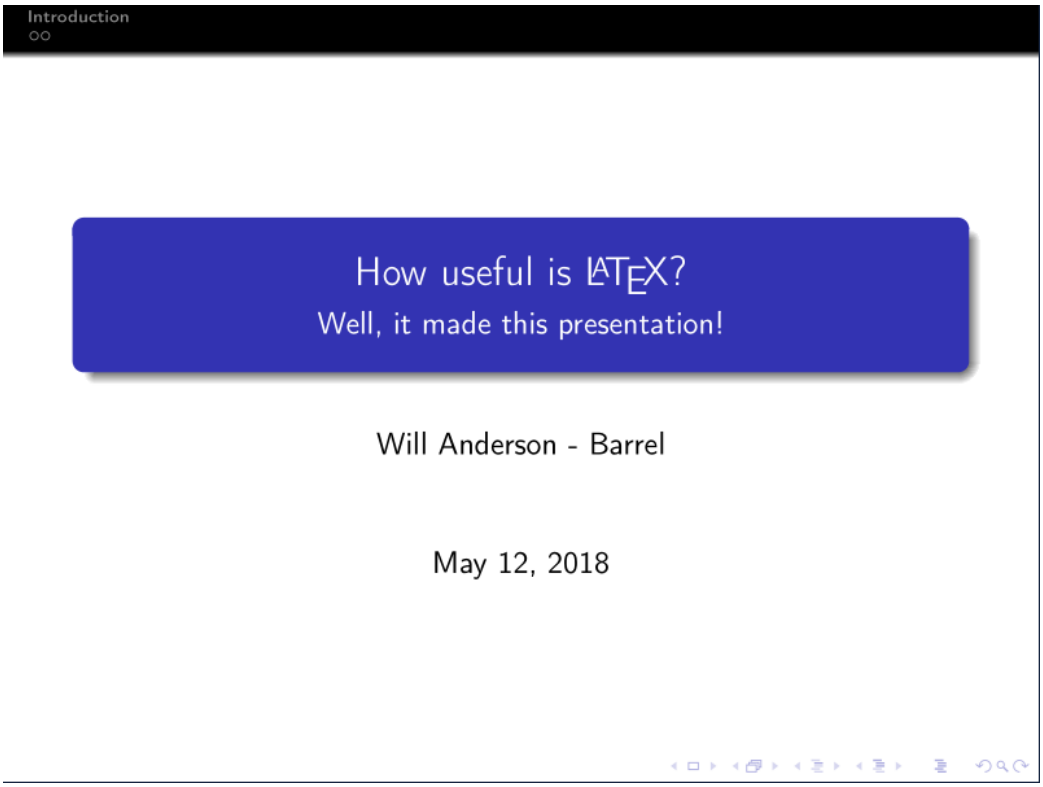

Figure 12: A Title page made with LaTeX and Beamer

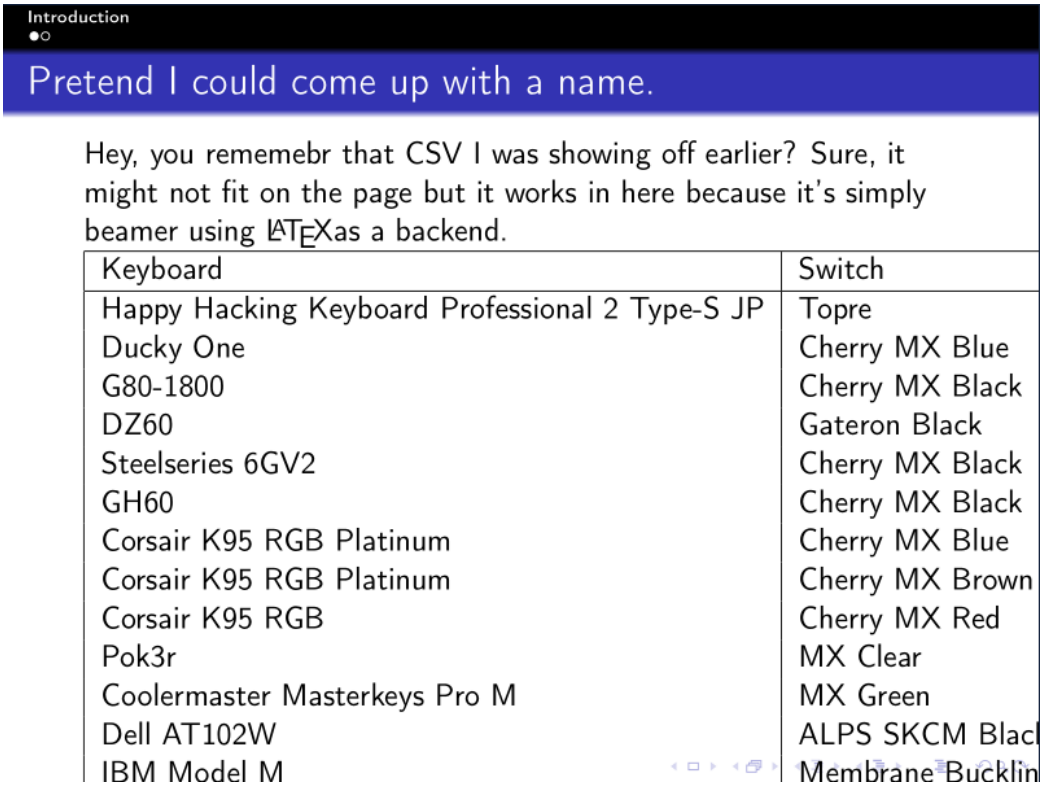

Figure 13: The second page made with LaTeX and Beamer

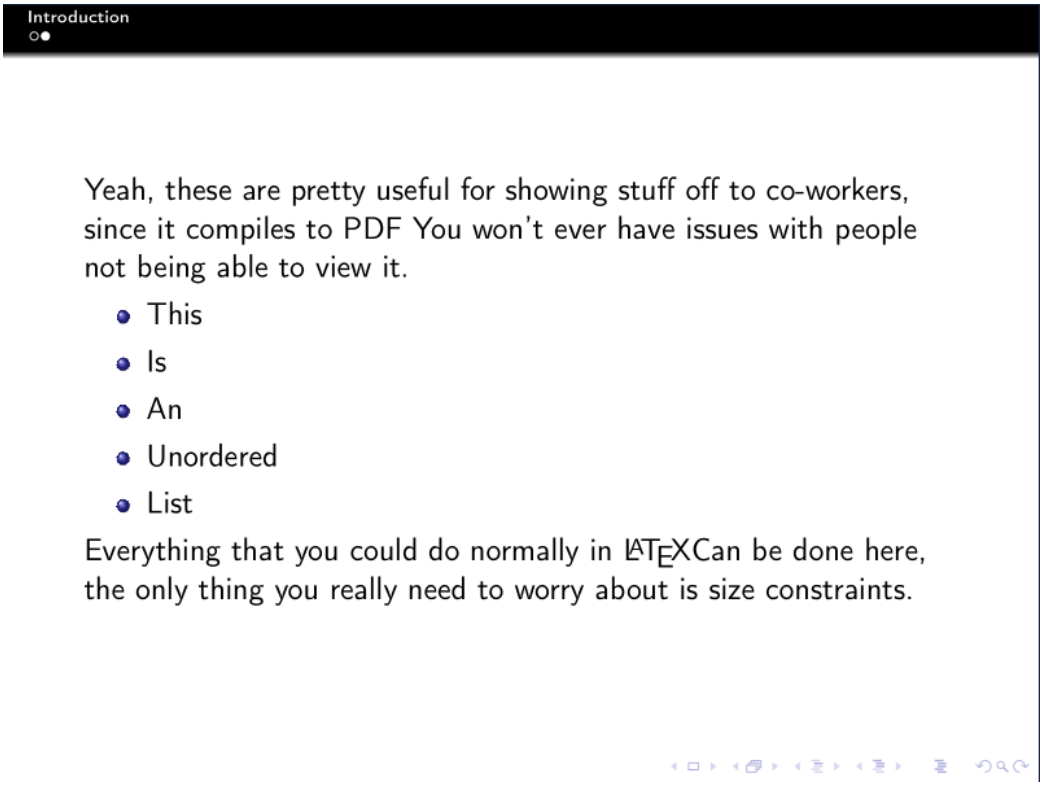

Figure 14: The third page made with LaTeX and Beamer

7.1 So, why should I use this instead of Powerpoint?

Well, this can literally take a normal document and quickly convert it into a slideshow, which is incredibly useful, it also will allow you to do weird stuff that I have shown off in this document, rather than using Visual Basic macros which don't really work, along with the fact that it's compiled to PDF, which makes it viewable on *any* device.