LaTeX: An Introduction (Part 1)

University Graduate College Training Course

Dr Martin Chorley

School of Computer Science & Informatics, Cardiff University

February 10th, 2013

Introduction

- What, Where and How of LaTeX
- Writing LaTeX files
- Document Classes & Structure
- Packages
- Sections & Chapters
- Text Formatting
- Tables
- Lists
- Graphics, Images & Figures in LaTeX
- Maths
    - Typesetting Maths
    - Equations

Not Covered

- Floating Environments
- Referencing & Bibliographies using BibTeX
- Complex commands
- Customising environments & commands
- Presentations in LaTeX

All covered in 'LaTeX: An Introduction (Part 2)' – 21st February, 2014.

Schedule

09:00 - 09:15	Welcome & Introduction
09:15 - 10:30	Basic LaTeX Introduction, Exercise 1 & 2
10:30 - 11:00	**Coffee Break**
11:00 - 12:30	More LaTeX, Exercise 2 & 3
12:30 - 13:30	**Lunch**
13:30 - 15:00	Graphics, Images & Figures, Exercise 4
15:00 - 15:30	**Coffee Break**
15:30 - 17:00	Typesetting Maths & Equations, Referencing, Exercise 5
17:00	Close

What is LaTeX?

TeX is software developed by Donald Knuth for typesetting documents.

- low level markup language and compiler
- very powerful, but difficult to use

LaTeX is a collection of software built around TeX to make life easier

- macros and scripts to convert LaTeX commands to TeX
- many packages for doing complex formatting/layouts

Why LaTeX?

- Allows you to concentrate on content and structure, rather than layout and presentation
- Keeps formatting consistent throughout the document
- No design or typography knowledge required
- Excellent for writing mathematical expressions or equations
- Long and complex documents can be created easily
- Free!

Why not LaTeX?

- Can't easily see how document looks as we write it
- Need to learn LaTeX to be able to create anything
- Lose some control over formatting and layout
- LaTeX is not the prettiest or simplest language out there

Where do we get LaTeX?

LaTeX is freely available from a number of different sources. There are many different implementations and collections of packages online.

Windows	**proTeXt** is a good solution for Windows, providing **MiKteX** (a set of LaTeX packages and package manager for Windows) along with a decent editor: **TeXnicCenter**
Mac OSX	**MacTeX** provides **TeXLive** (a set of LaTeX packages and package manager) along with a number of different editors and utilities.
Linux	Most distributions will come with LaTeX ready installed but if not you can install **TeXLive**.

How do we use LaTeX?

Creating documents with LaTeX is simple:

1. Write our document as plain text in a '`.tex`' file, using LaTeX commands to structure and format it
2. Compile our '`.tex`' file to produce the output

First (basic) LaTeX Example

```
\documentclass{article}
\begin{document}
   Hello World!
\end{document}
```

Hello World!

Writing LaTeX – Whitespace

Whitespace (spaces or tabs) are all seen as a 'space' by LaTeX. Several concurrent spaces are all seen as one space only. Any whitespace at the beginning of a line will be ignored, and a blank line is needed between two lines of text for them to be considered separate paragraphs.

```
If I add    multiple    spaces    between    words    \LaTeX\
will treat them as one   space.
This will not start a new paragraph.

This is a new paragraph.
```

If I add multiple spaces between words LaTeX will treat them as one space. This will not start a new paragraph.

This is a new paragraph.

Writing LaTeX – Reserved Characters

LaTeX uses a number of characters that have a special meaning as part of the language, so care must be taken when using these in `.tex` files. They will either not be displayed, or can cause LaTeX to do something differently.

```
\# \$ \^{} \& \_ \{ \} \~{} \textbackslash{} \%
```

```
# $ ˆ & _ { } ˜ \ %
```

To use these in normal text, add a \ before them. The exception to this rule is when using a \ itself, as LaTeX uses \\ to start a new line. If you want to use a backslash in normal text, use the `\textbackslash` command.

Writing LaTeX – Commands

LaTeX commands have an effect on the text in the document. Some commands have additional arguments or optional parameters. The general syntax for a LaTeX command is:

```
\commandname[opt1, opt2, ...]{arg1}{arg2}...
```

Writing LaTeX – Commands & Whitespace

Whitespace after LaTeX commands will generally be ignored. If you need a space after a command, you can either add an empty parameter to the command, or use a (breaking or non-breaking) space command.

```
\LaTeX commands will ignore whitespace after them.\newline
We can force a space after a \LaTeX{} command using an empty
parameter. \\
Or we can use a space command (texttt{\ } or \texttt{~} )after
our \LaTeX\ command.
This way our \LaTeX~commands and text do not flow together!
```

LaTeXcommands will ignore whitespace after them.
We can force a space after a LaTeX command using an empty parameter.
Or we can use a space command (texttt or   )after our LaTeX command.
This way our LaTeX commands and text do not flow together!

Writing LaTeX – Switches

Most LaTeX commands have a switch equivalent, which have no arguments but apply to all of the current scope.

```
\texttt{some text in typewriter font}, some text in normal font

{\tt some text in typewriter font}, some text in normal font
```

`some text in typewriter font`, some text in normal font
`some text in typewriter font`, some text in normal font

Be careful with switches. If you use them incorrectly, you can end up applying them to the whole document!

Writing LaTeX – Groups

You can restrict the effect of LaTeX commands or switches by using groups. A group is defined using a pair of braces.

```
Here is some text before a group
{
\it This text within the group is in italic
}
This text is not in italic as it is outside of the group
```

Here is some text before a group *This text within the group is in italic* This text is not in italic as it is outside of the group

Writing LaTeX – Comments

The ‘%’ character is used to create comments in LaTeX. When LaTeX is processing your `.tex` file and it comes across a ‘%’, it ignores the rest of the line.

```
%This is a comment and will not be shown.
Here is some text in our file that will be shown. %but the rest
of the line will not be.
We can even do things like br%
eak words up with comm%
ents if we want to.
```

Here is some text in our file that will be shown. We can even do things like break words up with comments if we want to.

Compiling

That's more than you need to create a basic `.tex` file and create your first document.

To compile your `.tex` file and create your document, you use a LaTeX compiler:

- `latex` calls the `tex` compiler and outputs `.dvi` files
- `pdflatex` calls the `pdftex` compiler and outputs `.pdf` files

Compiling

Compiling creates a lot of extra files, including the output of your document. All of these files are recoverable and can be remade by re-compiling, so can be deleted safely.

The only files you always need to keep and should not delete are `.tex`, `.cls`, `.sty`, `.bib` and `.bst`.

Exercise 1

Have a go at creating your first LaTeX document

Document Structure

Every LaTeX document must have a certain structure:

```
\documentclass{...}
\usepackage{...}
\begin{document}
...
\end{document}
```

The area before `\begin{document}` is called the *preamble*. It contains commands concerning the setup of the document.

The text of your document is enclosed between the `\begin{document}` and `\end{document}`, within the 'document' *environment*.

Environments

Environments enclose text and cause it to be treated a certain way, similar to commands. They usually have a larger scope than a command though. They begin with `\begin{...}` and end with `\end{...}`

```
\begin{document}
   Here is some text
   \begin{center}
       Here is some centred text
   \end{center}
\end{document}
```

Here is some text

Here is some centred text

Document Class

The \documentclass{...} command tells LaTeX which type of document we are creating, and how it should be set up and formatted. This command usually comes at the very beginning of the file.

As with many commands it has optional parameters, which will change aspects of the structure, formatting or layout.

```
\documentclass[opt1,opt2,...]{class}
```

Document Class

LaTeX comes with many types of document class built in. Some of the most commonly used are:

`article`	for scientific articles, short reports, papers etc.
`IEEEtran`	for articles in the IEEE Transactions format.
`report`	for longer reports containing chapters, small books, theses.
`books`	for real books
`beamer`	for writing presentations

Document Class Parameters (1/2)

Common optional parameters for classes include:

`10pt`, `11pt`, `12pt`	set the base font size for the document
`a4paper, letterpaper, ...`	set the paper size for the document
`titlepage`, `notitlepage`	`report` and `book` usually start a new page after the document title, `article` does not.

Document Class Parameters (2/2)

`onecolumn`, `twocolumn`	typeset the document in one or two columns
`twoside`, `oneside`	sets double sided or single sided output. `article` and `report` are single sided and `book` is double sided by default
`landscape`	changes the document to landscape rather than portrait

Third party classes will often come with documentation specifying which optional parameters they accept.

Document Class Example

So, to make a two-sided `article` in 12pt font on A4 paper, you can use the command:

```
\documentclass[12pt,a4paper,twoside]{article}
```

Packages

Often,the default set of commands available to LaTeX cannot solve all of our problems alone. To include graphics, use coloured text or other complicated functionality you will need to include extra packages.

These packages will often have extra optional parameters:

```
\usepackage[opt1, opt2, ...]{packagename}
```

So, for example, to use the package allowing us to use coloured text:

```
\usepackage{color}
```

Packages

We can include multiple packages in the `\usepackage` command:

```
\usepackage{color,graphicx,geometry}
```

Any packages where we want to set optional parameters need to use their own `\usepackage` command:

```
\usepackage{color,graphicx}
\usepackage[margin=2cm]{geometry}
```

Top Matter

After we've specified the document class and included any packages we want to use, we can define information about the document in the top matter.

```latex
\documentclass{article}

\title{Document Title}
\author{Me}
\date{February 2013}

\begin{document}
    \maketitle
\end{document}
```

Abstract

Usually, scientific papers and reports will have an abstract, so LaTeX includes an environment for specifying which part of your document is the abstract. `article` and `report` document classes can use the `abstract` environment.

```
\documentclass{article}

\begin{document}
    \begin{abstract}
        ...
        Abstract goes here
        ...
    \end{abstract}
    \ldots
\end{document}
```

Sections & Chapters

We often want to break documents into different parts, chapters or sections.

Command	Level
`\part{part_title}`	-1
`\chapter{chapter_title}`	0
`\section{section_title}`	1
`\subsection{subsection_title}`	2
`\subsubsection{subsubsection_title}`	3
`\paragraph{paragraph_title}`	4
`\subparagraph{subparagraph_title}`	5

Which section commands you can use depends on which document class you are using.

Text Formatting

LaTeX has many text formatting options included by default that allow you to change text size, style and spacing.

Text Size

There are several commands for changing the font size, used as below:

```
    {\Large This text is large} \\
    {\tiny This text is tiny}
```

This text is large

This text is tiny

Text Size

There are ten different options for specifying size:

1. `\tiny`
2. `\scriptsize`
3. `\footnotesize`
4. `\small`
5. `\normalsize`
6. `\large`
7. `\Large`
8. `\LARGE`
9. `\huge`
10. `\Huge`

Text Font

The font can be adjusted simply in a few ways:

```
\textsf{Sans Serif font} \\
\texttt{Teletype font} \\
\textit{Italic} \\
\textsc{Small Capitals} \\
\textbf{Bold} \\
\textrm{Roman font}
```

```
Sans Serif font
Teletype font
Italic
Small Capitals
Bold
Roman font
```

Text Font

Each font command has a corresponding switch:

Command	Switch
`\textsf{...}`	`{\sffamily ...}`
`\texttt{...}`	`{\ttfamily ...}`
`\textit{...}`	`{\itshape ...}`
`\textsc{...}`	`{\scshape ...}`
`\textbf{...}`	`{\bfseries ...}`
`\textrm{...}`	`{\rmfamily ...}`

Line Spacing

LaTeX has a built in command for inter-line spacing:

```
\linespread{factor}
```

Use a factor of 1.3 for one and a half line spacing, and 1.6 for double line spacing (the default line spacing factor is 1).

Line Spacing - The setspace Package

The setspace package allows further control over line spacing, providing commands and environments for changing the line spacing for the whole document or parts of the document.

For example, to change the line spacing for the whole document to one and half lines:

```latex
\usepackage{setspace}

%\singlespacing
\onehalfspacing
%\doublespacing
%\setstretch{1.1}
```

Text Emphasis

In order to emphasise text, use the `\emph{text}` command.

```
We can \emph{emphasise} a word in the sentence.
```

We can *emphasise* a word in the sentence.

The `\emph{...}` command is nice because it can be used within itself.

Quotation Marks

The " character is not used for quotation marks in LaTeX. Instead, use two ‘ (grave accent) characters for the opening quotation mark, and two ’ (vertical quote) characters for the closing quotation mark. For single quotes, use one of each:

```
The "double-quote" characters do not render as well as a
combination of ``grave accents'' and ``vertical quotes'' when
writing quotation marks.
```

The "double-quote" characters do not render as well as a combination of “grave accents” and “vertical quotes” when writing quotation marks.

Hyphenation

In order to lay text out properly, LaTeX will hyphenate words when necessary, to split them over multiple lines. If you find LaTeX is hyphenating words in the wrong places, you can suggest the correct places to hyphenate a word using the `\hyphenation{wo-rd-li-st ...}` command:

```
\hyphenation{DISCUSSION Hy-phen-a-tion}
```

Hyphenation

If you don't want to list all words at the beginning of your document in a `\hyphenation{}` command, you can add *suggested* hyphenation points to your text using `\-`.

The village 'Llanfairpwllgwyngyll-
gogerychwyrndrobwllllantysiliogogo-
goch' has a very long name indeed,
so we may want to suggest to LaTeX
how it might be hyphenated.

```
The village ‘Llan\-fair\-pwll
\-gwyn\-gyll\-go\-gerych\-
wyrnd\-rob\-wll\-llanty\-
siliogo\-go\-goch’ has a very
long name indeed, so we may
want to suggest to \LaTeX\ how
 it might be hyphenated.
```

This is useful for words with accented characters, as LaTeX will not hyphenate these words by default.

Hyphenation

To stop words being split over one line you can use the `\mbox` command:

My phone number will be changing
soon, the new number is 020 345
6721.

```
My phone number will be
changing soon, the new number
is 020 345 6721.
```

My phone number will be chang-
ing soon, the new number is
020 345 6721.

```
My phone number will be
changing soon, the new number
is \mbox{020 345 6721}.
```

Exercise 2

Use the more advanced structure and formatting commands you've learnt to create a paper with sections, subsections and formatted text.

Lists

LaTeX provides three different environments for defining lists:

`itemize`	For basic bulleted lists
`enumerate`	For numbered lists
`description`	For lists with a label for each item

Lists

All lists have the same basic structure:

```
    \begin{list_type}
        \item first list item
        \item second list item
        \item third list item
    \end{list_type}
```

itemize

itemize gives us a simple bulleted list:

```
\begin{itemize}
    \item the first item
    \item the second item
    \item the third item
\end{itemize}
```

- the first item
- the second item
- the third item

enumerate

enumerate gives us a numbered list:

```
\begin{enumerate}
   \item the first item
   \item the second item
   \item the third item
\end{enumerate}
```

1. the first item
2. the second item
3. the third item

description

`description` gives us a list where items have labels:

```
\begin{description}
    \item[First] the first item
    \item[Second] the second item
    \item[Third] the third item
\end{description}
```

First	the first item
Second	the second item
Third	the third item

Nested Lists

We can nest lists within lists, up to a depth of four

```
\begin{enumerate}
    \item the first item
         \begin{enumerate}
        \item the first subitem
        \item the second subitem
    \end{enumerate}
    \item the second item
    \item the third item
\end{enumerate}
```

1. the first item
    1.1 the first subitem
    1.2 the second subitem
2. the second item
3. the third item

Tables

The tabular environment allows us to create tables within our LaTeX documents.

```
    \begin{tabular}{ table_spec }
    \end{tabular}
```

The table_spec argument specifies the alignment of text to use in each column, and which columns should have vertical lines between them.

Table Specification

Each column or separator in a table is specified using one of the following specifiers:

Column Specifier	Result
`l`	left-justified column
`c`	centred column
`r`	right-justified column
`p{‘width’}`	paragraph column
`|`	vertical line
`||`	double vertical line

Table Layout

There are several commands and characters to use to enter text into and format a table:

Column Specifier	Result
`&`	column separator
`\\`	start a new row
`\hline`	add a horizontal line

Table Example 1

```
\begin{tabular}{ l | c | r }
   some text & some text & some text \\
   some text & some text & some text \\
   some text & some text & some text \\
\end{tabular}
```

some text	some text	some text
some text	some text	some text
some text	some text	some text

Table Example 2

```
\begin{tabular}{ | l || c || r | }
    \hline
    some text & some text & this text \\
    a text & other text & more text \\
    another text & some text & some other text \\
    \hline
\end{tabular}
```

some text	some text	this text
a text	other text	more text
another text	some text	some other text

Table Paragraphs

LaTeX will not automatically wrap text in table cells.

```
\begin{tabular}{ | l || c || r | }
    \hline
    some text & this text & some really long text that should be
    wrapped but it isn’t \\
    other text & more text & some more really long text that
    should be wrapped \\
    \hline
\end{tabular}
```

some text	this text	some really long text that should be wrapped but
other text	more text	some more really long text that should be w

Table Paragraphs

To make LaTeX do text wrapping in tables, the p specifier must be used for the table column and a width given.

```
\begin{tabular}{ | l || c || p{0.4\textwidth} | }
    \hline
    some text & this text & some really long text that should be
    wrapped and it is \\
    other text & more text & some more really long text that
    should be wrapped \\
    \hline
\end{tabular}
```

some text	this text	some really long text that should be wrapped and it is
other text	more text	some more really long text that should be wrapped

MultiColumn Rows

The `\multicolumn` command allows us to create table cells that span multiple columns. The syntax is:

`\multicolumn{numcols}{alignment}{contents}`

```
\begin{tabular}{ | r | l | }
    \hline
    \multicolumn{2}{|c|}{Table Heading} \\
    \hline
    some text & other text \\
    this text & more text \\
    \hline
\end{tabular}
```

Table Heading	
some text	other text
this text	more text

MultiRow Columns

Using the `multirow` package allows us to create tables with cells spanning multiple rows. The syntax is:

`\multirow{numrows}{width}{contents}`

```
\begin{tabular}{ | r | c | l | }
    \hline
    \multicolumn{3}{|c|}{Table Heading} \\
    \hline
    \multirow{2}{*}{this text} & some text & other text \\
    & more text & other more text \\
    \hline
\end{tabular}
```

Table Heading		
this text	some text	other text
	more text	other more text

Exercise 3

Pull everything together to create a paper with sections and subsections, text formatting, lists and tables.

Graphics

Including graphics in our LaTeX document is relatively simple.

The `graphicx` package allows us to use the `\includegraphics` command to read in an image from a file and insert it into our document.

Often the hardest part is actually getting the image in the first place!

Graphics - Raster vs Vector

Images can be stored in many ways within a file. Two common types of image are:

Raster	images stored as individual pixels within a file. Common raster image types include `.jpg`, `.png`, `.gif`
Vector	graphics stored as a series of points, lines, curves, shapes and polygons. A common vector image type is `.svg`.

`.pdf` and `.eps` files can contain both raster and vector image data within the same file.

Graphics - Raster vs Vector

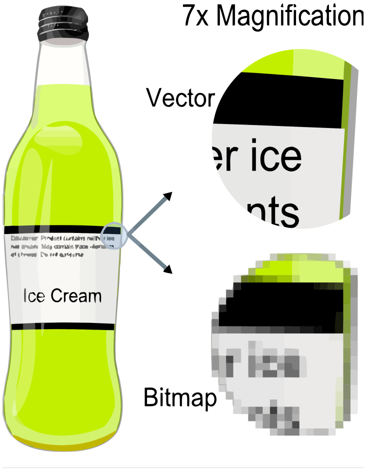


Image originally by Darth Stabro, taken from `http://en.wikipedia.org`

Graphics - Supported files

If you are using the `latex` compiler, the only supported graphic type is *Encapsulated PostScript*: `.eps` files.

If you are using the `pdflatex` compiler, `.jpg`, `.png`, `.pdf` and `.eps` files can be used.

Graphics - insertion

The `\includegraphics` command is used to insert an image file into your LaTeX document.

```
\includegraphics[opt1=val1, ..., ]{imagefile}
```

The argument to the command is the filename of the image, relative to the folder in which the LaTeX is compiled.

```
\includegraphics{img/rastervsvector}
```

Graphics - insertion

We usually supply the filename of the image *without* an extension, and allow LaTeX to choose which file to use. LaTeX will choose the best type of image depending on the compiler and output type.

You can specify which graphics types are to be used by including the `\DeclareGraphicsExtensions` command in your preamble.

```
\DeclareGraphicsExtensions{.pdf,.png,.jpg}
```

\includegraphics - options

`width=xx`	Specify the preferred width of the image
`height=xx`	Specify the preferred height of the image
`keepaspectratio`	True or False, affects behaviour when scaling
`scale=xx`	Scale the image by the given factor
`angle=xx`	Rotate the image by the given number of degrees
`trim=l b r t`	Crop the image
`clip`	Must be true for `trim` to work
`page=x`	Select a particular page from a `.pdf` file

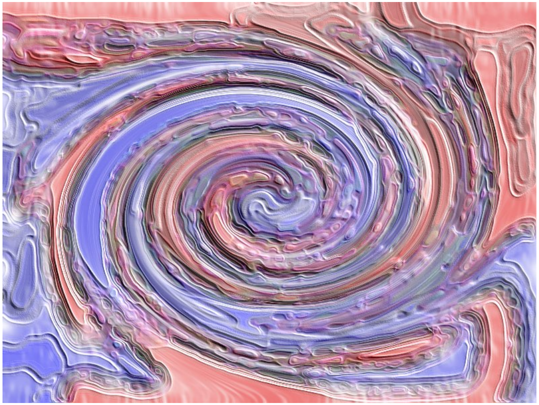
\includegraphics - examples

```
\includegraphics{img/background}
```

\includegraphics - scale

We can scale an image by a given factor

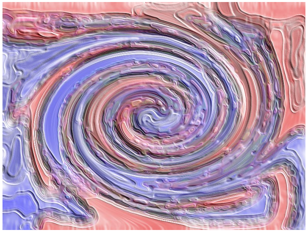
```
\begin{center}
   \includegraphics[scale=0.25]{img/background}
\end{center}
```



\includegraphics - width

We can set the image to a preferred width

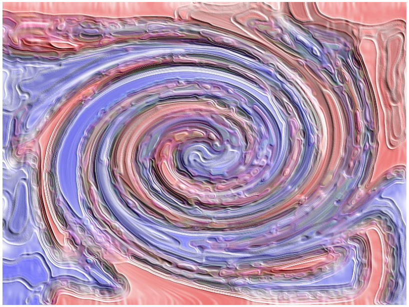
```
\begin{center}
    \includegraphics[width=4cm]{img/background}
\end{center}
```



\includegraphics - height

Or we can set the image to a preferred height

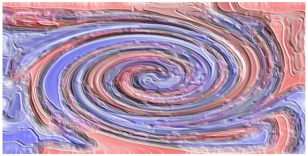
```
\begin{center}
    \includegraphics[height=4cm]{img/background}
\end{center}
```



\includegraphics - width and height

Or we can set the image to a preferred height *and* width

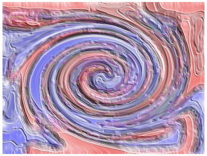
```
\begin{center}
   \includegraphics[width=4cm, height=2cm]{img/background}
\end{center}
```



\includegraphics - width and height

Or we can set the image to a preferred height *and* width, but maintain the aspect ratio.

```
\begin{center}
   \includegraphics[width=4cm, height=2cm, keepaspectratio=true
   ]{img/background}
\end{center}
```



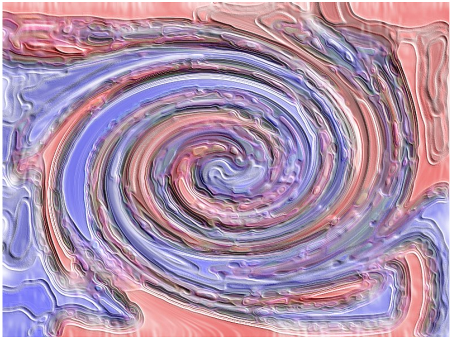
\includegraphics - relative width or height

We can set the image size relative to items in our environment such as the `\linewidth`, `\textwidth` or `\textheight`

```
\begin{center}
    \includegraphics[width=\textwidth]{img/background}
\end{center}
```

\includegraphics - relative width or height

```
\begin{center}
   \includegraphics[width=0.5\textwidth]{img/background}
\end{center}
```



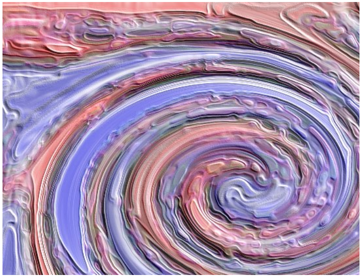
\includegraphics - rotating

```
\begin{center}
   \includegraphics[width=0.4\textwidth, angle=90]{img/
   background}
\end{center}
```

\includegraphics - trim

We can crop parts of the image off the `left`, `bottom`, `right` or `top`

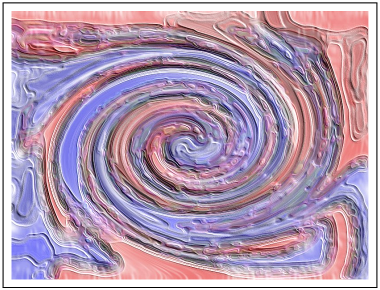
```
\begin{center}
    \includegraphics[trim= 10mm 60mm 80mm 5mm, clip, width=0.4\
    textwidth]{img/background}
\end{center}
```



Graphics - borders

The \fbox command can be used to put borders around things like pictures

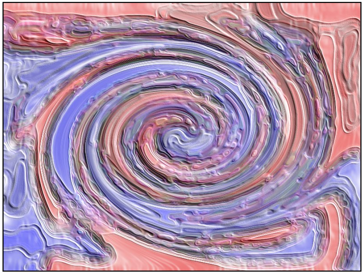
```
\begin{center}
    \fbox{\includegraphics[width=0.4\textwidth]{img/background}}
\end{center}
```



Graphics - borders

The border padding can be changed using the \setlength\fboxsep command

```latex
\begin{center}
   \setlength\fboxsep{0pt}
   \fbox{\includegraphics[width=0.4\textwidth]{img/background}}
\end{center}
```



Graphics - borders

The border thickness can be changed using the `\setlength\fboxrule` command

```
\begin{center}
    \setlength\fboxrule{4pt}
    \fbox{\includegraphics[width=0.4\textwidth]{img/background}}
\end{center}
```

Exercise 4

Experiment with adding images into your documents.

Help?

There are *many*, *many* places to get more help with LaTeX.

If you have a problem, use Google! Often that will lead you straight to the documentation for the package or command you have a problem with.

Otherwise, StackExchange has a thriving TeX community where you can ask for help and advice:

`http://tex.stackexchange.com`