Introduction to LaTeX

Lecture II: Text

Liu Yihao

SJTU-UMJI Technology Department

April 15, 2020

Use UTF-8 encoding in LaTeX

UTF-8 encoding is widely used in modern computer applications, so it's useful to include the inputenc package and use UTF-8 encoding.

Command

```
\usepackage[utf-8]{inputenc}
```

Example

café

However, different operating systems and compiling engines have different support on UTF-8 encoding, some UTF-8 codes that work on your computer may not work on others (though rarely), so it is recommended to use commands (will be introduced later) instead of directly copy and paste the UTF-8 codes from the Internet.

Special symbols

Some special symbols can't be directly used since they are reserved by LaTeX

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

Many LaTeX starters are confused with how to correctly print quotes, hyphens and dots.
` prints a left single quote, ' prints a right single quote.
`` prints a left double quote, '' prints a right double quote.
one hyphen (-) print like -
two hyphens (--) print like –
three hyphens (---) print like —
\dots prints the dots with a correct format (. . .) instead of directly use three dots (...)

Accent on letters

Sometimes you may need an accent form of a letter, here is an example of letter o

\`{o}	ò	\'{o}	ó	\^{o}	ô	\"{o}	ö	\~{o}	õ
\={o}	ō	\.{o}	ȯ	\u{o}	ŏ	\v{o}	ǒ	\H{o}	ő
\t{oo}	o͡o	\r{o}	o̊	\c{o}	o̧	\d{o}	ọ	\b{o}	o̲

Something interesting

You may be curious about how to print words like LaTeX, actually it's defined as a command.

- \TeX - TeX
- \LaTeX - LaTeX
- \LaTeXe - LaTeX 2$_\varepsilon$

Deal with unfamiliar symbols

Sometimes you may want to deal with symbols you have never seen. In this case, you may refer to http://detexify.kirelabs.org/classify.html to find out how to output the character.

Basic commands about fonts

First, lets start with some commands that transform font types

- \bf - **Sample Text**
- \it - *Sample Text*
- \rm - Sample Text
- \sc - Sample Text
- \sf - Sample Text
- \sl - *Sample Text*
- \tt - `Sample Text`

Note that the commands that transform font types influence the text in the whole scope ({...}) until another font type is specified. For example, how to use the first command \bf is shown below

{\bf Sample Text}

Sometimes we don't want to transform all the font types, instead, we can only change the font type of some specified text.

Example

```
\textbf{Sample text}
```

There are more options for fonts.

- \textit - *Sample Text*
- \textsc - Sample Text
- \textsf - Sample Text
- \texttt - `Sample Text`

However, in a math environment (will be introduced later), some other commands should be used

- \mathbf - $\mathbf{Sample\ Text}$
- \mathit - $\mathit{Sample\ Text}$
- \mathsf - $\mathsf{Sample\ Text}$

Note that the math environment doesn't include all of the font types on the previous page. More information about font types can be found here.

Font size can also be easily modified

- \tiny - Sample Text
- \scriptsize - Sample Text
- \footnotesize - Sample Text
- \small - Sample Text
- \normalsize - Sample Text
- \large - Sample Text
- \Large - Sample Text
- \LARGE - Sample Text
- \huge - Sample Text
- \Huge - Sample Text

Build a colorful document

Changing the color is similar to changing font types.

If you want to transform to a color (like transforming to bold with `\bf`), you can use `\color{name}`.

Similarly, you can use `\textcolor{name}` like `\textbf`.

The background color of the whole page can be set using `\pagecolor{name}`.

There are some defined color name in the xcolor package.

black	gray	olive	teal	
green	orange	violet	brown	
pink	white	cyan	lime	
yellow	darkgray	magenta	red	

You can find more information in the documentation of xcolor (texdoc xcolor)

Ulem package

If you want to add some lines on the text, use the ulem package.

Command

```
\usepackage{ulem}
\uline{Sample Text}
```

There are different kinds of lines supported:

- \uline - Sample Text
- \uuline - Sample Text
- \uwave - Sample Text
- \sout - ~~Sample Text~~
- \xout - Sample Text
- \dashuline - Sample Text
- \dotuline - Sample Text

Enumerate

When you need to enumerate some items as a list, you may use the enumerate package.

Command

```
\usepackage{enumerate}
\begin{enumerate}[style]
\item % ...
\item % ...
\item % ...
\end{enumerate}
```

This will generate a normal list with the serial numbers in the specified style, which could be the following (as example)

- 1 - 1, 2, 3, 4, ...
- (i) - (i), (ii), (iii), (iv), ...
- [1.] - [1.], [2.], [3.], [4.], ...

Itemize

If you want to generate an unordered list, use itemize instead of enumerate.

Command

```
\usepackage{enumerate}
\begin{itemize}
\item[style] % ...
\item[style] % ...
\item[style] % ...
\end{itemize}
```

In this case, style must be added after each item, which is different from that in enumerate, and the symbol displayed in the beginning of each item will be exactly same as the style. If style is not added, a default style will be used.

Alignment

If you want to align a paragraph of text, use these three environments for left/center/right align.

Command

```
\begin{flushleft/center/flushright}
% ...
\end{flushleft/center/flushright}
```

However, if only a single line needs to be aligned, use these three commands.

Command

```
\leftline{text}
\centerline{text}
\rightline{text}
```

Spaces may be confusing

There are defined command of spaces in different width and usages.

- ␣ - the basic space in LaTeX (printed in yellow since it's transparent). Note that any number of spaces or tabs is equal to one space, and the space after a command is ignored. If you want to add an extra space, use \␣ which makes a 1/3 em space (1 em is approximately the width of an M in the current font)
- ~ - If two words can't be separated on two lines, you can tell LaTeX about it using a tie (~), such as Prof.~Hamade (Prof. Hamade).
- \, - makes a 1/6 em space, commonly used before units (notice the space before em on this page)
- \; - makes a 2/7 em space
- \quad - makes a 1 em space
- \qquad - makes a 2 em space
- - makes actually the space of text, but text will be invisible.

Separate contents into lines and pages

Here are some basic commands about lines and pages in LaTeX, you will use them everywhere.

- \newline - begin a new line
- \\ - begin a new line (not recommended[1])
- \par - begin a new paragraph (a new line with indent)
- \\[offset] - begin a new line with an vertical offset, offset is the size of needed space (not recommended, using `\vspace` instead.)
- \newpage - begin a new page
- % - begin a line comment

[1] According to Manuel Charlemagne, \\ should only be used for a force break (where \newline doesn't work).

Spacing

When trying to separate two paragraphs by a certain space, many new learners of LaTeX may use multiple empty lines and linebreaks, which is a very dirty fix and is not so accurate. Actually, LaTeX provides a precise spacing mechanism.

Command

```
\vspace{space}
\vspace*{space}
```

When trying to show the next paragraph or sentence precisely at the bottom of the current page, we can use

Command

```
\vfill
```

between the contents of two paragraphs to separate them.

Predefined skipping

More often[1], we don't need to think about the skipping space, we can use the predefined skipping commands to achieve a small, medium or big skip. They are actually particular cases of `\vspace`

Command

`\smallskip`

`\medskip`

`\bigskip`

You may note that the effects are these skipping commands have been already shown above.

[1] According to Manuel Charlemagne, you should always use these skipping commands if possible instead of using `\\` (as in many online tutorials).

Spacing units

The space can be anything representing a size, such as 1cm, 2em and 10pt. In LaTeX, spacing units can be

- cm
- mm
- in - inch, 1 inch = 2.54 cm
- pt - 72 pt = 1 inch, the smallest unit in LaTeX
- em - 1em equals to the width of letter M
- ex - 1ex equals to the width of letter x
- `\linewidth` - the width of current line in the container
- `\pagewidth` - the width of the page
- `\pageheight` - the height of the page
- `\textwidth` - the normal width of text on the page
- `\textheight` - the normal height of text on the page

Minipage

minipage is a very useful environment for dividing pages into a grid.

Example

```
\begin{minipage}{0.32\linewidth}
  % ...
\end{minipage}
\hfill % Fill horizontal space
\begin{minipage}{0.32\linewidth}
  % ...
\end{minipage}
\hfill % Fill horizontal space
\begin{minipage}{0.32\linewidth}
  % ...
\end{minipage}
\vfill % Fill vertical space
```

```
\begin{minipage}{0.32\linewidth}
  % ...
\end{minipage}
\hfill % Fill horizontal space
\begin{minipage}{0.32\linewidth}
  % ...
\end{minipage}
\hfill % Fill horizontal space
\begin{minipage}{0.32\linewidth}
  % ...
\end{minipage}
```

The code above generate six minipages in a grid of 3 columns $\times$ 2 rows. Don't try to add up the width of minipages in a line for more than about `0.98\linewidth` (since a minipage have a small margin on each side), or the last minipage may be on a new line.

For each minipage, it can be seem as an independent LaTeX document, where text, formulas, graphics, tables and etc. can be inserted, and most importantly, they won't affect each other. What's more, you can even use minipages in a minipage to form a multi-level nesting.

The multicol package

When typesetting contents with small line width and many lines (for example, source code), the multicol package is recommended.

Command

```
\usepackage{multicol}
\begin{multicols}{cols}
    % contents on column one
    \breakcolumn % break the current column here
    % contents on column two
\end{multicols}
```

Here cols is the number of columns, it must be specified. If `\breakcolumn` is not used, the multicol package will automatically balance the length of each column.

Spelling languages

If you want to use a spelling language with characters similar to English, package babel can be used (exactly the same name as babel).

Command

```
\usepackage[languages]{babel}
```

- languages - a list of languages, the last one to be the default language

Example

```
\usepackage[greek,english]{babel}
\textgreek{abcdefgABCDEFG}
```

Then LaTeX will print αβςδεφγAB῍ΔΕΦΓ
Of course, you can use some simple commands to print these greek letters directly, such as `\alpha`, `\beta` and etc, which is more convenient only when few of them are needed.

Chinese

The Chinese TeX Community maintains a package called ctex for inputing Chinese in LaTeX. Note that it is only a package, which is shipped with most modern TeX Suites, not the CTEX Suite. I don't think it's a good choice to use the CTEX Suite directly.

Command

```
\usepackage{ctex}
```

The default LaTeX compiler pdflatex doesn't have support on Chinese input with ctex package, xelatex is a recommended modern LaTeX compiler as a replacement.

However, the ctex package is too heavy and it can slow down the total compilation speed seriously.

Usage of scope in LaTeX

First, you should realize the meaning of "scope" in programming. Let's start with a simple example in C/C++ (assuming you know that):

```
1  int main()
2  { // The scope "main" of function main
3    int a = 1; // int a is defined in scope "main"
4    for (int i = 0; i < 10; i++)
5    { // The scope "for" of the for loop
6      int b = i; // int b and i are both defined in scope
       ↪ "for"
7      a += b; // int a can be visited here!
8    }
9    { // The scope "other", we can directly define a scope
     ↪ like this
10     int c; // int c is defined in scope "other"
11     c = a; // int a can be visited here!
12   }
13   a -= c // error: c is not in scope "main", can't be
     ↪ visited!
14 }
```

In the example of C/C++, we use brackets {} to define a scope, which is just the same in LaTeX. In addition, notice that an environment or a command also defines a scope.

Example

```
1  black (default) text \\
2  \color{blue}
3  blue text \\
4  { \color{brown} brown
   ↪  text }
5  \begin{center}
6    \color{red}
7    centered red text
8  \end{center}
9  \textbf{ \color{brown}
10 bold brown text } \\
11 blue text
```

black (default) text
blue text
brown text

centered red text

bold brown text
blue text

With the usage of scopes, you can flexibly change the color, font or anything else you wish in a self-defined range of the document.