

# Introduction to $\LaTeX$

## Lecture 5: Advanced usages of $\LaTeX$

Liu Yihao

SJTU-UMJI Technology Department

March 5, 2019

# Newcommand

Sometimes you are building a huge project (like this lecture), and you may use certain type of syntax for many many times. Now it's time to define your own command with `\newcommand` in the beginning of the document (where the `\usepackage` commands appear).

## Command

```
\newcommand{\yourcommand}[arg_num]{code}
```

- `arg_num` - number of arguments in your command
- `code` - the code of your command, use `#1`, `#2`, ..., `#n` to represent the arguments

## Example

```
\newcommand{\samplecommand}[1]{\textbackslash #1}
```

It is defined to simply display the commands in red in this lecture.

# Renewcommand

Another times you need to redefine the commands, then `\renewcommand` can be used. It's very similar to `\newcommand`, the only difference is that you must use `\newcommand` when the command doesn't exist, while using `\renewcommand` when the command has been defined (by you or  $\text{\LaTeX}$  packages) before.

## Command

```
\renewcommand\definedcommand[arg_num]{code}
```

## Example

```
\renewcommand\thesection{\Roman{section}}
\renewcommand\thesubsection{\Alph{subsection}}
```

By default, the number before the section titles of `\section` is 1, 2, 3, etc, this command will change them to a capital form of roman numbers, I, II, III, etc. And subsection numbers become A, B, C, etc.

# New/Renewenvironment

Environments can also be defined.

## Command

```
\newenvironment{name}[arg_num]{begdef}{enddef}
\renewenvironment{name}[arg_num]{begdef}{enddef}
```

- `name` - the name of your environment
- `arg_num` - number of arguments in your environment
- `begdef` - the code to substitute the begin clause of your environment
- `enddef` - the code to substitute the end clause of your environment

## Example

```
\newenvironment{command}{\begin{block}{Command}}{\end{block}}
```

# Include and Input

When you are building a huge project, if you write all of the code in a single file, the compiling of the whole project will be very slow, and the length of the file will also confuse you. Then you can use `\include` and `\input` to avoid this.

## Command

`\include{file}` - Include the file on a new page, the files are compiled separately.

`\input{file}` - Directly replace the command with the whole file, doesn't start a new page, but the compiling won't speed up.

If you are including a .tex file, then the extension name can be omitted. Another command `\includeonly{list}` can be added to the beginning of the document, so that only the include files in `list` are compiled and others are ignored, this is very useful in debugging huge projects.

# Insert pdf

If you want to insert pdf files into your tex file, you can

## Command

```
1 \usepackage{pdfpages}
2 \includepdf{file}
```

# Chemical formulas

For typing formulas in  $\text{\LaTeX}$ , besides equation environment, you can also use `mhchem` package.

The documentation for `mhchem` can be referred at <http://ctan.mirrors.hoobly.com/macros/latex/contrib/mhchem/mhchem.pdf>

# Hyperlink

Hyperlinks are supported in  $\text{\LaTeX}$ , use the `hyperref` package.

## Command

```
\usepackage{hyperref}
\hypersetup{options}
\url{url}
\href{url}{text}
```

Some common `options` are listed below:

- `colorlinks` - boolean (default false)
- `urlcolor` - color for linked URLs (default magenta)
- `linkcolor` - color for normal internal links (default red)



# Listings

Sometimes you are asked to attach your code about your report or homework. Using `listings` package will avoid dealing with various special symbols and rearranging all of your code. (`texdoc listings` for more information)

## Example

```

1 \usepackage{listings}
2 \lstset{language=[LaTeX]TeX, numbers=left, tabsize=4,
   ↪ keywordstyle=\color{blue}\bfseries, identifierstyle=\bf,
   ↪ breaklines=true, basicstyle=\tiny, rulecolor=\color{brown},
   ↪ numberstyle=\color[RGB]{20,20,20}}
3 \begin{lstlisting}
4 %code here
5 \end{lstlisting}

```

# minted

**Minted** is another way to include code into  $\text{\LaTeX}$ . Unlike listings, it is easier to use because there is no need to pre-set the syntax highlighting. However, you need to download **pygments**, and add `--shell-escape` in the configuration of XeLaTeX.

## Example

```
\usepackage{minted}
\begin{minted}{c++}
#include <iostream>
int main{
    std::cout<<"Hello world";
}
\end{minted}
```

```
1  #include <iostream>
2  int main{
3      std::cout<<"Hello world";
4  }
```

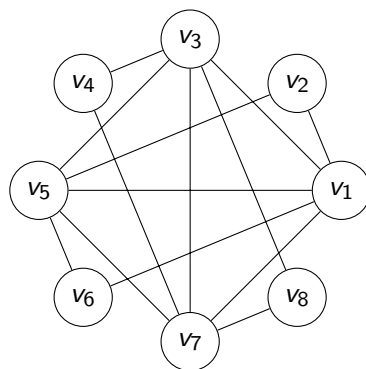
# Draw graphs with TikZ and PGF

In your VE203 or some projects, you may need this package to draw graphs. There is a document of more than one thousand pages about it ([texdoc tikz](#) or [texdoc pgf](#))

```

1  \begin{tikzpicture}[scale=2,
   ↪ bend angle=22.5]
2  \tikzstyle{every
   ↪ node}=[draw,shape=circle];
3  \foreach \i in {1,...,8}
4  {
5  \path (45*\i-45:1cm) node (v\i)
   ↪ {$v\_i$};
6  }
7  \draw
8  (v1) -- (v2) (v3) -- (v4) (v5)
   ↪ -- (v6) (v7) -- (v8)
9  (v1) -- (v3) (v3) -- (v5) (v5)
   ↪ -- (v7) (v7) -- (v1)
10 (v2) -- (v5) (v4) -- (v7) (v6)
   ↪ -- (v1) (v8) -- (v3)
11 (v1) -- (v5) (v3) -- (v7);
12 \end{tikzpicture}

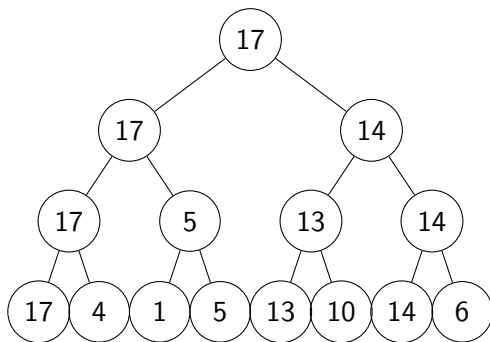
```



```

1  \begin{tikzpicture}[scale=0.8]      14      }
2  \tikzstyle{every                    15      }
   ⇨ node}=[draw,shape=circle,minimum child { node {14}[sibling
   ⇨ size=0.8cm];                      ⇨ distance=2cm]
3  \node {17}[sibling                  17      child {
   ⇨ distance=4cm]                     18          node {13}[sibling
4  child { node {17}[sibling           ⇨ distance=1cm]
   ⇨ distance=2cm]                     19          child { node {13} }
5      child {                         20          child { node {10} }
6          node {17}[sibling           21      }
   ⇨ distance=1cm]                     22      child {
7          child { node {17} }         23          node {14}[sibling
8          child { node {4} }         ⇨ distance=1cm]
9      }                               24          child { node {14} }
10     child {                         25          child { node {6} }
11         node {5}[sibling            26      }
   ⇨ distance=1cm]                     27      };
12         child { node {1} }         28  \end{tikzpicture}
13         child { node {5} }

```



The process of memorizing the code in TikZ is quite hard, so while you need to plot some graph with TikZ, it is highly recommended that you refer to the <http://www.texample.net/tikz/> for the codes of examples shown in it.