LaTeX: from dummy to TeXnician

core Typography. How TeX works-1

Anton Lioznov

Skoltech,
Project Center of Omics Technologies and Advanced Mass Spectrometry

ISP 2025,
lesson 4

Skoltech

What we will know?

Technical agreements

Introduction

How TEX “sees” the document

TEX primitives

What we will know?

Technical agreements

Introduction

How TEX “sees” the document

TEX primitives

Agreements

inclass/outclass versions

- two slightly different versions for class and home
- class version is more interactive and contains less information
- $\longleftrightarrow$ this line will be shown only at home version

Frame for home

Agreements

II

Footnotes

- For second reading
- Contains advanced usage of the command
- Contains references to read more
  - to the exact chapter
  - (often) with the href to exact page
- Contains some comments
- Mostly for outclass version

Like this

Agreements
III

Addition information – “magic”

- To have the full picture
- Not to analyze or to puzzle out in class

★ Agreements ★

V

Exercises

- To work in class

Special thanks to

Our TAs:

- Peter Borisovets
- Pavel Kuzmin
- Anna Litvin

What we will know?

Technical agreements

Introduction

How TEX “sees” the document

TEX primitives

Separation “how” and “what”

$\longleftrightarrow$ This is *the* point of non-WYSiWYG

- Focus on what you want to do, not how to do it. Leave the creation of “how” to someone else
- But *who* create these “how” rules?
- Who needed to :). In these last lectures it will be you

Ugly “backend”, beautiful “frontend”

→

- TEX is a very old language
- it allows you so many things, redefine almost anything...
- but its syntax may be unclear

←

- Why figures and footnotes don’t rendered where they appears, but sent where they must be?
- Why can you write `\pictofraction{xx}{blue}{3}{black!30}{3}{\tiny}` and the \tiny size will appears in the command result, and not just change the following text?
- Why can I use something like `\magicPage` command and don’t change the template itself? (And why now you see the command, not its result?)

Examples

I need to write as simple as

- `\magicPage` to create a "magic hat"
- `\twocolImg{<anything>}{image}` to create the pattern "code to the left, picture to the right"
- `\overC{https://www.overleaf.com/learn/latex/Page_size_and_margins}` to shorten this (with hyperref!) to /Page_size_and_margins
- no addition code for the progressbar at the top, except `\usepackage{progressbar}` and define `progressbarcolor`

but the tricks I use...

“Backend” of the commands I

“Magic hat”:

```latex
\def\magicPage{\ifinclassmode%  indicate page as "magic" -- some addition information that is not stricly required
\begin{tikzpicture}[remember picture,overlay,shift={(current page.north east)}]
\node[anchor=north east,xshift=-0cm,yshift=-0cm]{%
{\includegraphics[width=1cm]{images/magic}}%
};
\end{tikzpicture}%
\fi}
```

Code and image pattern:

```latex
\newbox\Abox
\newlength{\myhh}
\newcommand{\twocolImg}[2]{  % add two columns: (1) is some text (2) is an image.
    \begin{columns}
        \begin{column}{0.45\textwidth}
            #1
        \end{column}
        \begin{column}{0.45\textwidth}
        %
            % https://tex.stackexchange.com/questions/3166/how-can-the-dimensions-of-a-box-be-retrieved-with-latex
            \savebox{\Abox}{\includegraphics[width=\textwidth, keepaspectratio]{#2}}
            \settoheight{\myhh}{\usebox{\Abox}}
            \raisebox{-0.5\myhh+1ex}[0pt][0pt]{\includegraphics[width=\textwidth, keepaspectratio]{#2}}
        \end{column}
    \end{columns}
}
```

“Backend” of the commands II

Overleaf short view:

```
\def\overRepl https://www.overleaf.com/learn/latex#1 {#1}
\newcommand{\overC}[1]{
{\normalfont \href{https://www.overleaf.com/learn/latex\overRepl #1
↪  }{\raisebox{-0.5ex}{\includegraphics[width=3ex,height=3ex,keepaspectratio]{images/overleaflogo}}\!\!\url{\overRepl #1
↪  }}}}
```

Progressbar:

```
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesPackage{progressbar}[2024/12/17 The section-based progressbar]

\def\ifnot#1{#1\else\expandafter\expandafter\fi\iffalse\iftrue\fi} % <--                    if.
↪  https://tex.stackexchange.com/questions/108911/negating-ifeof
\def\ifnottitle#1{\ifnot{\ifnum\insertframenumber=1}\relax#1\fi} %           (       )

\newif\ifsupressprogressbar % <--              -
\newdimen\rule@length
\rule@length=\paperwidth%
\setbeamertemplate{headline}{%
\immediate\update@section@counters% <--          ,              (     )
\immediate\set@sectionprogress@length% <--
    \ifnottitle{%
    \smash{%
        \ifnot{\ifsupressprogressbar}%
            \lower0.1cm\hbox{%
                {%
```


Skoltech

“Backend” of the commands III

```
                    \color{progressbarcolor}%
                    \rule{\rule@length}{0.1cm}%
                }%
            }%
        \fi%
    }}%
}%


\let\old@section=\beamer@section        % <---     "   "             ,

%% 2.
\newwrite\sectionfile@writer
\openout\sectionfile@writer=\jobname.secw\relax

\long\def\beamer@section[#1]#2{% <--                \expandmacros
    \old@section[#1]{#2}\relax% <--
    \immediate\write\sectionfile@writer{\insertframenumber}% <--
}
\def\progressend{% <-                   -
    \immediate\write\sectionfile@writer{\the\numexpr\insertframenumber+1\relax}% <--                +1
    \supressprogressbartrue
}

%% 3.
\newtoks\sections@list% <----     ,
\newread\sectionfile@reader
```


Skoltech

“Backend” of the commands IV

```
\openin\sectionfile@reader=\jobname.secw\relax
\def\readallsections{%
    \def\stopread{\par }% <--                    ,
    \loop\relax% <--
    \ifnot{\ifeof\sectionfile@reader}\relax%  <--                    ,
    \read\sectionfile@reader to\newline% <--
        \ifx\newline\stopread\relax%  <--          '\par ',
            \def\newline{-1\space}%  <--          '\par ' '-1 '.
        \fi\relax%
        \edef\tempa{\the\sections@list\newline}% <-- edef    ,                                    ,                    .
        ↪                          https://tex.stackexchange.com/questions/38067/how-does-one-append-material-to-a-token-list
        \sections@list=\expandafter{\tempa}%  <--                    .    \expandafter              \tempa
        ↪       '\tempa'
    \repeat\relax%
}
\readallsections% <--                  .

%% 4.
\newcount\section@start@slide% <--
\newcount\section@end@slide% <--
\section@start@slide=1
\section@end@slide=-1

\def\spit@head@section#1 #2\relax{% <--                          '3 8 11 -1 '.                        '3'                ,
↪   '8 11 -1 '
    \def\a{#1}%
    \ifnot{\ifx\a\@empty}%
```


Skoltech

“Backend” of the commands V

```
        \global\section@end@slide=#1% <--
        \ifx\relax#2\relax% <--    #2=\relax,              .
            \global\sections@list=\expandafter{-1 \space}% <--                          '-1 '
        \else\relax% <--
            \global\sections@list=\expandafter{#2}% <--
        \fi\relax%
        \ifnum\section@end@slide=-1\relax% <--            .               -1,
            \global\section@end@slide=\numexpr \inserttotalframenumber + 1\relax%
        \fi\relax%
    \fi%
}

\def\getnext@section@slidenumber{% <--           ,
    \edef\tempa{\the\sections@list}% <--
    \ifnot{\ifx\tempa\empty}\relax% <--      -
        \expandafter\spit@head@section\tempa\relax%<--
    \fi\relax%
}


\def\update@section@counters{% <--
    \ifnum\section@end@slide=-1\relax% <--
        \ifnum\inserttotalframenumber=1\relax% <--                          ,
            \global\section@end@slide=1\relax%
        \else%
            \global\section@end@slide=\numexpr \inserttotalframenumber + 1\relax%
        \fi%
```

“Backend” of the commands VI

```
    \fi\relax%
    \ifnot{\ifnum\insertframenumber<\section@end@slide}\relax% <--                                  ,                    ,
    ↪        !
        \getnext@section@slidenumber% <--
        \global\section@start@slide=\insertframenumber% <--
    \fi\relax
}
\update@section@counters% <--          ,


%% 5.
\def\set@sectionprogress@length{% <--          .                  a,                    b,                d...
%                <basic>/(b-a) * (d-1-a)
    \ifnum\section@start@slide>1\relax%
        \rule@length=\paperwidth%
        \newcount\thisposprogressbar%
        \thisposprogressbar=\numexpr\insertframenumber +1 - \the\section@start@slide\relax%
        \newcount\thatposprogressbar%
        \thatposprogressbar=\numexpr\the\section@end@slide - \the\section@start@slide\relax%
        \ifnot{\ifnum\thatposprogressbar>0}% <--                                  "            "
            \global\thatposprogressbar=1%
        \fi
        \divide\rule@length by \thatposprogressbar%
        \multiply\rule@length by \thisposprogressbar%
    \else%
        \rule@length=0pt
```

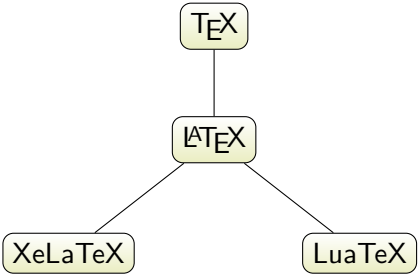

Skoltech

“Backend” of the commands VII

```
    \fi%
}
```

We will learn how to do things like the above

LaTeX vs TeX



- LaTeX — is the most popular set of macro-extensions (or macro package) of the computer typesetting system TeX, which facilitates a typesetting of complex documents.
- TeX — is a typesetting system designed and mostly written by Donald Knuth — the "father of modern Computer Science". TeX was designed with two main goals in mind: to allow anybody to produce high-quality books using minimal effort and to provide a system that would give exactly the same results on all computers, at any point in time

$\longleftrightarrow$ Now it is become important to separate the LaTeX ideas from TeX's ones

What we will know?

Technical agreements

Introduction

How TEX “sees” the document

TEX primitives

What we will know?

How TEX “sees” the document

- Modes
- Boxes and glue
- Paragraphs and pages creation

Modes

TEX has 3(6) modes:

1. **Vertical mode.** [Building the main vertical list, from which the pages of output are derived.]
2. **Internal vertical mode.** [Building a vertical list for a vbox.]
3. **Horizontal mode.** [Building a horizontal list for a paragraph.]
4. **Restricted horizontal mode.** [Building a horizontal list for an hbox.]
5. **Math mode.** [Building a mathematical formula to be placed in a horizontal list.]
6. **Display math mode.** [Building a mathematical formula to be placed on a line by itself, temporarily interrupting the current paragraph.]

Difference between modes

The modes have lots of differences. For example:

- in horizontal mode only first space is taking into account
- in math mode generic font is italic, all spaces are ignored
- in Display math mode operators are drawing bigger, than in the regular one
- in vertical mode all spaces and <return>s are ignored

use `\leavevmode` to leave vertical mode

More about math mode

Math actually has 4 different styles. When you see that superscript x^y is smaller then the text — it is a different style. The styles are:

Display style	`\displaystyle`	A	main style for displayed formula
Text style	`\textstyle`	A	main style for in-text formula
Script style	`\scriptstyle`	A	main style for scripts
Script-script style	`\scriptscriptstyle`	A	main style for scripts in scripts

What we will know?

How TEX “sees” the document

- Modes
- Boxes and glue
- Paragraphs and pages creation

Boxes and Glue

$\longleftrightarrow$ TEX sees the document as boxes and glue

Boxes and Glue

$\rightarrow$

- Box is a rectangle with width, heigh and depth
- Glue is a tensile space

$\leftarrow$

Main idea

A symbol is a box
it is part of a word, that is a box
the words are connected with glue
into sentances and paragraphs.
A paragraph is a box
it connected with another one with glue
to the page. Which is a box

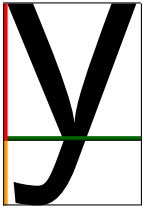
by the way: table, picture, ... is a box

Box params

y

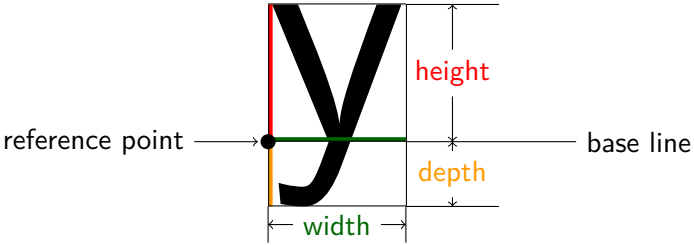
/40977/confused-with-tex-terminology-height-depth-width

Box params

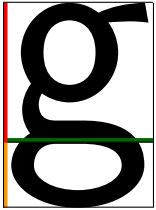


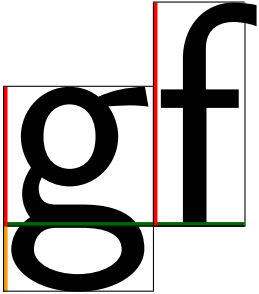
/40977/confused-with-tex-terminology-height-depth-width

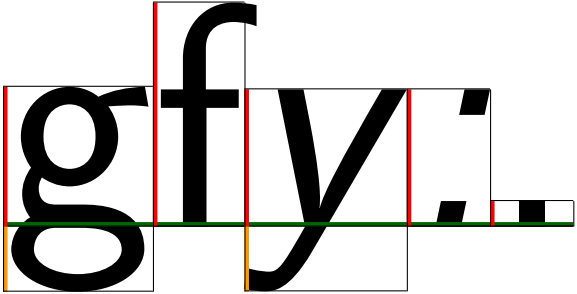
Box params

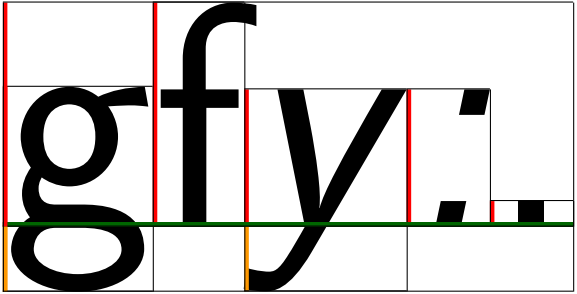


/40977/confused-with-tex-terminology-height-depth-width




gf


gfy:.


gfy:.

What we will know?

How TEX "sees" the document

- Modes
- Boxes and glue
- Paragraphs and pages creation

“this, in fact, is probably the most interesting aspect of the whole TEX system”

D. Knuth, the TEXBook

Slide for perfectionists

how many word-breaks are

Microsoft Word 2008

Call me Ishmael. Some years ago – never mind how long precisely – having little or no money in my purse, and nothing particular to interest me on shore, I thought I would sail about a little and see the watery part of the world. It is a way I have of driving off the spleen, and regulating the circulation. Whenever I find myself growing grim about the mouth; whenever it is a damp, drizzly November in my soul; whenever I find myself involuntarily pausing before coffin

Adobe InDesign CS4

Call me Ishmael. Some years ago – never mind how long precisely – having little or no money in my purse, and nothing particular to interest me on shore, I thought I would sail about a little and see the watery part of the world. It is a way I have of driving off the spleen, and regulating the circulation. Whenever I find myself growing grim about the mouth; whenever it is a damp, drizzly November in my soul; whenever I find myself involuntarily pausing before coffin warehouse

pdf-LaTeX 3.1415926

Call me Ishmael. Some years ago – never mind how long precisely – having little or no money in my purse, and nothing particular to interest me on shore, I thought I would sail about a little and see the watery part of the world. It is a way I have of driving off the spleen, and regulating the circulation. Whenever I find myself growing grim about the mouth; whenever it is a damp, drizzly November in my soul; whenever I find myself involuntarily pausing before coffin

Hyphenation and inter-word spacing statistics

	Word	InDesign	pdf-LaTeX
Number of hyphenations	9	10	4
SD of IWS (pt)	2.26	1.94	1.42
Maximum IWS (pt)	14.4	13.2	9.0
Number of lines with IWS > 9 pt	5	2	0

SD: standard deviation; IWS: inter-word spacing

/110133/visual-comparison-between-latex-and-word-output-hyphenation-typesetting-ligat
http://www.rtznet.nl/zink/latex.php?lang=en

Skoltech

Paragraph creation overview

- All paragraph is considered as one: the words in the last line can change the typesetting in the first line.
- TEX will never put words narrow than the glue allow.
- TEX tries out all possible varients for line breaks. For each varient and each line TEX calculates the *badness*. If it is lower than `\tolerance`, TEX will try to create paragraph with the minimum of hyphenation.
- if TEX fails, it provides **Overfull** or **Underfull** warnings.

How to suggest a hyphentation

Locally: use `\-` as in this ve\-ry long se\-nta\-nce
Globally: `\hyphenation{some-thing poss-ible}`

`\-` is just a short version for `\discretionary{hpre-break texti}{hpost-break texti}{hno-break texti}`
also: TEX will never hyphenate the word with "/". Use `\slash` if you want to allow it. And `\uchyph=0` will prohibit hyphenation in words on uppercase letter.

Manual line break manipulation

Never break: non-breaking space ~, `\nobreak`, `\nolinebreak`
Always: `\\`, `\break`, `\linebreak`
You can use `\obeylines` to follow the line breaks in the source code.

by the way: `\\` has an optional parameter: the vertical space after the command. Also `\smallskipamount`, `\medskipamount` and `\bigskipamount` responsible for skipping after paragraph. `\linebreak` has an optional parameter: [0-4] how much you want to have the break here.

Algorithm: part 1

1. TEX produce varients without word breaks. It compare the *badness* with `\pretolerance` param.
2. *badness* is $\simeq$ 100·<proportion-between-the-normal-glue-and-its-stretching/compression>^{3}
3. if `\pretolerance`-try fall, TEXwill try to use all posible line breaks to make each badness less than `\tolerance`

defaults: `\pretolerance`=100, `\tolerance`=200. `\pretolerance`>=10000, the first try will always succeed

Algorithm: part 2

1. line breaks are allowed only in certain places:
   1.1 glue
   1.2 kern with glue after
   1.3 and of math ($) and glue after
   1.4 the manual or auto-passed penalty
   1.5 discretionary break
2. The penalty to the first three is 0. For the last one, it is defined by `\hyphenpenalty=` or `\exhyphenpenalty=`. The penalty can be manually added as `\penalty`
3. Penalty can both positive and negative. If it is $> 10^4$ there will be no break ever, if it is $< -10^4$ there always will be a break

defaults: `\hyphenpenalty`=50, `\exhyphenpenalty`=50
you can put addition spaces between math and text with `\mathsurround`

Algorithm: part 3

1. in reality, TEX tries to minimize the *demerits*. It is proportional to the badnesses, `\linepenalty`(determines how much you want tex to produce a minimum amount of lines) and penalty
2. TEX also takes into account and add penalty if two lines one after another has a hyphenation (`\doublehyphendemerits`), if lines are *visually incompatible* (ex: if a tight line is next to a loose one) (`\adjdemerits`) and if the second-last line of the entire paragraph ends with a discretionary (`\finalhyphendemerits`)

defaults: `\linepenalty`=10, `\adjdemerits`=10000, `\doublehyphendemerits`=10000, `\finalhyphendemerits`=5000.
`\hfuzz`=... will add the maximum length of string's aligment

What else?

- Use `\narrow` to make lines narrow
- Use `\looseness`=-1 to ask TEX to try make one line less in paragraph
- `\prevgraf` shows the curent line in the paragraph.
- `\vadjust` adds something at the vertical list after current line. With it we add the star to the left

*

- `\everypar`adds something in each paragraph
- `\parfillskip`— the glue after last line
- `\parskip`— the vertical glue between paragraphs

Non-standart paragraph form

```
\hangindent=1.5cm
\hangafter=-2 \noindent
With such paragraphs we can add something
↪  to the begin of the paragraph! It is
↪  really interesting.

\vspace*{\fill}

\hangindent=-1.5cm
\hangafter=1 \noindent
With such paragraphs we can add something
↪  to the end of the paragraph! It is
↪  really interesting.
```

With such paragraphs we can add something to the begin of the paragraph! It is really interesting.

With such paragraphs we can add something to the end of the paragraph! It is really interesting.

`\hangindent`=[>0] — the addition indent to the left. ...=[<0] — indent to the right.
`\hangafter`=[>0] — the indent to all lines after. ...=[<0] — before.

Non-standart paragraph form

```
\parshape=14
0cm 6cm .1cm 5.8cm .17cm 5.66cm .5cm 5cm
.9cm 4.2cm 1.05cm 3.9cm 1.1cm 3.8cm 1.1cm
↪  3.8cm
1.05cm 3.9cm .9cm 4.2cm .5cm 5cm .17cm
↪  5.66cm
.1cm 5.8cm 0cm 6cm
\noindent \small
Lorem ipsum dolor sit amet, consectetur
↪  adipiscing elit. Ut elit tellus,
↪  pharetra quis est ac, aliquam lobortis
↪  odio...
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut elit tellus, pharetra quis est ac, aliquam lobortis odio... Suspendisse at purus eu elit sagittis euismod sit amet id nulla. Quisque faucibus bibendum nisl ac commodo. Aliquam elit nisl, accumsan sit amet fermentum a, porttitor sit amet turpis. Sed a blandit leo, a suscipit nibh. Pellentesque non purus aliquam, rhoncus felis sed, accumsan nisi. Cras sed eros dapibus, blandit enim in, tempus massa. In tristique orci dui, eu porttitor mauris condimentum vitae.

Page creation

- TEX was created in time when it was not enough memory to optimize pages globally.
- TEX finds the best break to the current page and then erase it from memory.
- More or less the algorithms are the same.
- You can use `\penalty` or `\nobreak` in vertical mode
- You can use to remove bottom page alignment
- Also as in paragraph, you allow to use `\newpage`, `\pagebreak`, `\nopagebreak`

What we will know?

Technical agreements

Introduction

How TEX “sees” the document

TEX primitives

Entities

1. Primitive commands
2. Counters (=integer numbers)
3. Lengths
4. Boxes
5. Glues
6. Spaces
7. Toks (Strings)
8. Inserts

commands and macros will wait in details the next lecture. No we discuss it just in a few words

What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

Simple command creation

```latex
\usepackage{xcolor}

% without arguments
\newcommand{\sk}{Skolkovo institute of
↪  science and technology}
% with arguments
\newcommand{\pointIt}[1]{\fbox{\textit{#1}}}
% With two arguments
\newcommand{\highlight}[2]{\textcolor{#2}{#1}}

\sk\ is a \pointIt{great} university!
↪  \highlight{We love}{red} \sk!
```

Skolkovo institute of science and technology is a $\boxed{\textit{great}}$ university! We love Skolkovo institute of science and technology!

What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

What is “counter”

“Counter” is just an integer number.
It’s using in multiple places to count everything in LaTeX: sections, equations, references, citation, enumerate lists,...

Define and simple manipulation with counters

```latex
\newcounter{abcd}
\arabic{abcd}

\setcounter{abcd}{2017}
\arabic{abcd}

\addtocounter{abcd}{-27}
\arabic{abcd}
```

0
2017
1990

- `\newcounter` to define new counter
- `\setcounter` to set counter to new value
- `\addtocounter` to add a number to the counter

lv: VII.3.1 man: 12.5 man: 13.4-5 Counters#Counter_manipulation

Print counter

`\arabic`{countname}	1	2	3	4	5	6	7	8	9
`\alph`{countname}	a	b	c	d	e	f	g	h	i
`\Alph`{countname}	A	B	C	D	E	F	G	H	I
`\roman`{countname}	i	ii	iii	iv	v	vi	vii	viii	ix
`\Roman`{countname}	I	II	III	IV	V	VI	VII	VIII	IX
`\fnsymbol`{countname}	*	†	‡	§	¶	‖	**	††	‡‡

P.S. `\value` to get "raw" value of the counter

Counters#Counter_style man: 13.1
see lv: IX.2.3 for russian analog of `\alph`

pre-defined counters in standart classes

part chapter section subsection subsubsection paragraph subparagraph

page figure table footnote equation
enumi enumii enumiii enumiv

TEX's counters (will talk later) `\year` `\month` `\day` `\time`

Counters#LaTeX_default_counters man: 13

Counter Domination

problem

You may want to write something like

1 Action

Task #1.1. Prepare the template
Task #1.2. Write the code
Task #1.3. Look at it

2 Viewing

Task #2.1. Compile the code 1.2

But the straightforward solution will give you

1 Action

Task #1. Prepare the template
Task #2. Write the code
Task #3. Look at it

2 Viewing

Task #4. Compile the code 1

Counter Domination

straightforward solution

```
\begin{document}
\newcounter{task} % new counter here!
\newcommand{\tsk}{\par
↪  \addtocounter{task}{1}%
    \textbf{Task \#\arabic{task}.} }

\section{Action}
\tsk Prepare the template
\tsk Write the code \label{write}
\tsk Look at it
\section{Viewing}
\tsk Compile the code \ref{write}
\end{document}
```

1 Action

Task #1. Prepare the template
Task #2. Write the code
Task #3. Look at it

2 Viewing

Task #4. Compile the code 1

Counter Domination

The Way

\newcounter{task}	→	\newcounter{task}[section]

\newcounter{<slave>}[<master>] will resets the value of <slave> if the value of <master> is change

\addtocounter{task}{1}	→	\refstepcounter{task}

\refstepcounter{<counter>} use it to update \label–\ref mechanism

\textbf{Task \#\arabic{task}.	→	\textbf{Task \#\arabic{section}.\arabic{task}.

Inside \newcommand{\tsk} to redefine the labels

	→	\renewcommand{\thetask}{\arabic{section}.\arabic{task}}

\renewcommand{\the<counter>} to redefine the reference

Counter Domination

solution

```latex
\begin{document}
\newcounter{task} % new counter here!
\newcommand{\tsk}{\par \addtocounter{task}{1}%
    \textbf{Task \#\arabic{task}.} }

\section{Action}
\tsk Prepare the template
\tsk Write the code \label{write}
\tsk Look at it
\section{Viewing}
\tsk Compile the code \ref{write}
\end{document}
```

```latex
\begin{document}
\newcounter{task}[section] % new counter here!
\newcommand{\tsk}{ \par\refstepcounter{task}%
    \textbf{Task \#\arabic{section}.\arabic{task}.} }
\renewcommand{\thetask}{ \arabic{section}.\arabic{task}}
\section{Action}
\tsk Prepare the template
\tsk Write the code \label{write}
\tsk Look at it
\section{Viewing}
\tsk Compile the code \ref{write}
\end{document}
```

Counter Domination

solution

```
\begin{document}
\newcounter{task}[section] % new counter here!
\newcommand{\tsk}{\par\refstepcounter{task}%
    \textbf{Task \#\arabic{section}.\arabic{task}.}}
\renewcommand{\thetask}{\arabic{section}.\arabic{task}}
\section{Action}
\tsk Prepare the template
\tsk Write the code \label{write}
\tsk Look at it
\section{Viewing}
\tsk Compile the code \ref{write}
\end{document}
```

1 Action

Task #1.1. Prepare the template
Task #1.2. Write the code
Task #1.3. Look at it

2 Viewing

Task #2.1. Compile the code 1.2

Redefine existing counter domination

"equation" example

Package based solution:

```
\usepackage{chngcntr}
```

and `\counterwith{equation}{chapter}` to make the "equation" a slave or `\counterwithout{equation}{chapter}` to "free" the counter.

Core-based solution:

```
\makeatletter
\@removefromreset{equation}{section}
\@addtoreset{equation}{chapter}
\renewcommand{\theequation}{\thechapter.\@arabic\c@equation}
\makeatother
```

/61756/how-to-change-equation-numbering-style /54241/change-the-type-of-equation-numbering-in-document-class-article
/28333/continuous-v-per-chapter-section-numbering-of-figures-tables-and-other-docume https://texfaq.org/FAQ-running-nos lv: IX.2.1.
Also see `\p@` prefix lv: IX.2.2 and /61426/how-to-make-ref-display-only-subsection

Define and simple manipulation

Define new `\newcount\<countname>` as `\newcount\mycounter`
Set number `\<countname>=<number>` Or use `\countdef`. Like `\countdef\mynumber=43`
Add number `\advance\<countname> by <number>`. Also there are `\multiply` and `\divide`. As well as `\numexp`.
Show number `\the\<countname>` or `\number` or `\romannumeral`

kn: 15 /245635/formal-syntax-rules-of-dimexpr-numexpr-glueexpr
Actually you can use `\count<number>` like `\count212`. What `\newcount` do is just find a free number and fix it to your defined name.

Define and simple manipulation

Example

```
\newcount\counttest
\counttest=40
\advance\counttest by 2
\the\counttest

\number\year\ is \romannumeral\year\par
```

42
2019 is mmxix

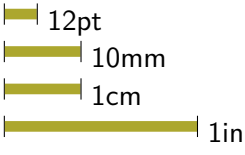
What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

Length

absolute values

		most common used:
pt	points	$\simeq$0.35mm
mm	millimeters	$\simeq$2.84pt
cm	centimeter	$\simeq$28.4pt, 10mm
in	inch	$\simeq$72.27pt, $\simeq$, 25.4mm



Length

absolute values

		not so common used:	
pt	points	$\simeq$0.35mm	12pt
mm	millimeters	$\simeq$2.84pt	10mm
bp	big point	1/72in, $\simeq$1.003pt	12bp
pc	pica	12pt, 4.2mm	1pc
dd	didot	$\simeq$1.07pt, $\simeq$0376mm	12dd
cc	cicero	12dd	1cc
sp	scaled point	$1/2^{16}$pt $= 1/65536$pt	2097152sp

(pt and mm here for comparison)

Every TEX's length is a integer number of sp

Length

Relative values

pt	points	12pt
mm	millimeters	10mm
em	roughly the width of an 'M' (uppercase)	1em
ex	roughly the height of an 'x'	1ex

example: \Huge

mm	10mm
em	1em
mm	10mm
em	1em

use em for horizontal and ex for vertical cases

Prebuild lengths

Most used

TEX's	
\parindent	The normal paragraph indentation
\parskip	The extra vertical space between paragraphs
LATEX's	
\textwidth	The width of the text on the page
\textheight	The height of the text on the page
\linewidth	The width of the text in this “box”
\lineheight	The height of the text in this “box”

By typography rules, don’t put both parskip and parindent as paragraph separation.

/Lengths#LaTeX_default_lengths man: 5.5 /16942/difference-between-textwidth-linewidth-and-hsize

\parskip is actually a glue. You can define your own lengths (See next lecture.)

Prebuild lengths

not so common used

LaTeX

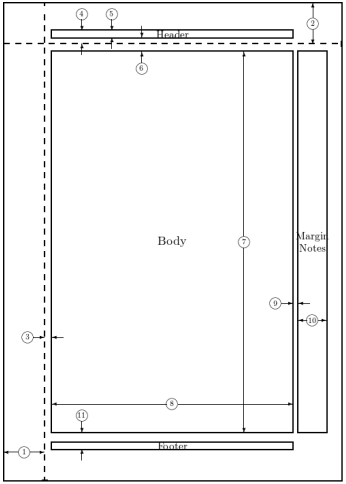
practicully every length — the margins; footnote, footer/header place; distance between columns,..

TeX

`\hsize`, `\vsize`	the normal size of text in page
`\hoffset`, `\voffset`	the offset of on page

lv: IV.4 man: 5.5

by the way, you can magnitude whole text by `\mag`=... parameter. Also there are `\hangindent` and `\hangafter` TeX params, but we'll speak in paragraph section.




```
1   one inch + \hoffset        2   one inch + \voffset
3   \oddsidemargin = 13pt      4   \topmargin = −23pt
5   \headheight = 12pt         6   \headsep = 25pt
7   \textheight = 674pt        8   \textwidth = 426pt
9   \marginparsep = 10pt       10  \marginparwidth = 50pt
11  \footskip = 30pt               \marginparpush = 5pt (not shown)
    \hoffset = 0pt                 \voffset = 0pt
    \paperwidth = 597pt            \paperheight = 845pt
```

How to use lengths

```
\usepackage{printlen}
\indent \printlength{\parindent}\par
\parindent=1pt\indent
↪  \printlength{\parindent}\par
\parindent=4em\indent
↪  \printlength{\parindent}\par
```

15.0pt

1.0pt

40.0pt

Just `\<length-command>=<length>`

How to use lengths

```
\usepackage{printlen}
\indent \printlength{\parindent}\par
\parindent=1pt\indent
↪  \printlength{\parindent}\par
\parindent=4em\indent
↪  \printlength{\parindent}\par
\parindent=0.5\parindent\indent
↪  \printlength{\parindent}\par
```

15.0pt

1.0pt

40.0pt

20.0pt

Arifmetics: <multiply-factor>\<length-command>

How to use lengths

```
\usepackage{printlen}
\indent \printlength{\parindent}\par
\parindent=1pt\indent
↪  \printlength{\parindent}\par
\parindent=4em\indent
↪  \printlength{\parindent}\par
\parindent=0.5\parindent\indent
↪  \printlength{\parindent}\par
\parindent=2em+1cm\indent
↪  \printlength{\parindent}\par
```

15.0pt

1.0pt

40.0pt

20.0pt

+1cm 20.0pt

Arifmetics: BUT You can't use simple notation +, -, /, *,...

How to use lengths

```
\usepackage{printlen}
\indent \printlength{\parindent}\par
\parindent=1pt\indent
↪  \printlength{\parindent}\par
\parindent=4em\indent
↪  \printlength{\parindent}\par
\parindent=0.5\parindent\indent
↪  \printlength{\parindent}\par
\parindent=2em+1cm\indent
↪  \printlength{\parindent}\par
\parindent=\dimexpr2em+1cm\indent
↪  \printlength{\parindent}\par
\parindent=\dimexpr(2em+1cm)/2\indent
↪  \printlength{\parindent}\par
```

15.0pt

1.0pt

40.0pt

20.0pt

+1cm 20.0pt

48.45274pt

24.22638pt

Arifmetics: `\dimexpr` allow to use “normal” notation.

Length manipulation

Define length `\newlength{\<lenname>}`
Set length `\setlength`
Add length `\addtolength`.
Show length `\the\<lenname>`. But also you can use `\usepackage{printlen}` and then `\uselengthunit`, `\printlength`

Length manipulation

Example

```latex
\usepackage{printlen}

\newlength{\mylen}
\setlength{\mylen}{40mm}
\addtolength{\mylen}{2cm}
\the\mylen

\uselengthunit{mm}\printlength{\mylen}
```

170.71652pt
59.99928 mm

Length manipulation

Define length `\newdimen\<lenname>`
Set length `\<lenname>=<len>`
Add length `\advance\<lenname> by <len>`. Also there are `\multiply` and `\divide`. As well as `\dimexp`.
Show length `\the`

Length manipulation

Example

```
\newdimen\mylen
\mylen=40mm
\advance\mylen by 2cm
\the\mylen
```

170.71652pt

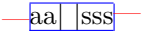
What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

TEX boxes

“Horizontal” boxes

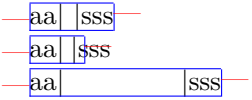
```
\boxing{\hbox        {aa| |sss}}\\
```



\hbox create box around the text. The box will never split in linebreak.

“Horizontal” boxes

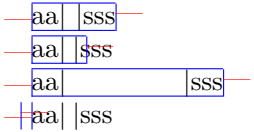
```
\boxing{\hbox        {aa| |sss}}\\
\boxing{\hbox to 20pt{aa| |sss}}\\
\boxing{\hbox to 70pt{aa| |sss}}\\
```



You can specify the length of the box with keyword `to`

“Horizontal” boxes

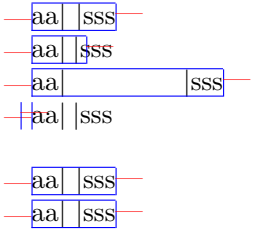
```
\boxing{\hbox        {aa| |sss}}\\
\boxing{\hbox to 20pt{aa| |sss}}\\
\boxing{\hbox to 70pt{aa| |sss}}\\
\boxing{\hbox to -4pt{aa| |sss}}\\
```



You even can set the box to negative size

“Horizontal” boxes

```
\boxing{\hbox        {aa| |sss}}\\
\boxing{\hbox to 20pt{aa| |sss}}\\
\boxing{\hbox to 70pt{aa| |sss}}\\
\boxing{\hbox to -4pt{aa| |sss}}\\
\boxing{\hbox             {aa| |sss}}\\
\boxing{\hbox spread 0pt  {aa| |sss}}\\
```

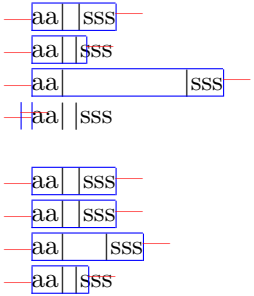


Another keyword, `spread` is the addition width

“Horizontal” boxes

```
\boxing{\hbox        {aa| |sss}}\\
\boxing{\hbox to 20pt{aa| |sss}}\\
\boxing{\hbox to 70pt{aa| |sss}}\\
\boxing{\hbox to -4pt{aa| |sss}}\\

\boxing{\hbox             {aa| |sss}}\\
\boxing{\hbox spread 0pt  {aa| |sss}}\\
\boxing{\hbox spread 10pt {aa| |sss}}\\
\boxing{\hbox spread -10pt {aa|
↪  |sss}}\\
```



also both positive and negative

“Horizontal” boxes

Usage

```
\hbox to -1pt{/}=
```

$\neq$

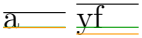
```
\begin{tabbing}
\hbox to 4em{}\=\hbox to 4em{}\kill
a \> b\\
hello\> world!
\end{tabbing}
```

a	b
hello	world!

“Vertical” boxes

TEX-way

```
\boxing{\vbox{a}}
\boxing{\vbox{yf}}
```



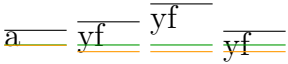
`\vbox` create box around the text

#vbox-rp, #vtop-rp

“Vertical” boxes

TEX-way

```
\boxing{\vbox{a}}
\boxing{\vbox{yf}}
\boxing{\vbox to 15pt{yf}}
\boxing{\vbox to 5pt{yf}}
```



You can specify the height of the box with keyword `to`

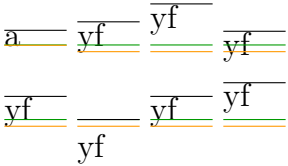
#vbox-rp, #vtop-rp

“Vertical” boxes

TEX-way

```
\boxing{\vbox{a}}
\boxing{\vbox{yf}}
\boxing{\vbox to 15pt{yf}}
\boxing{\vbox to 5pt{yf}}

\boxing{\vbox{yf}}
\boxing{\vbox to -5pt{yf}}
\boxing{\vbox spread 0pt{yf}}
\boxing{\vbox spread 5pt{yf}}
```



You even can set the box to negative size, use `spread`. But depth will remain the same

#vbox-rp, #vtop-rp

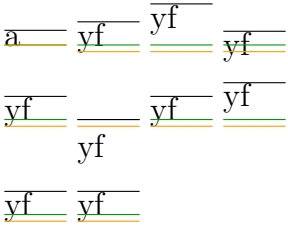
“Vertical” boxes

TEX-way

```
\boxing{\vbox{a}}
\boxing{\vbox{yf}}
\boxing{\vbox to 15pt{yf}}
\boxing{\vbox to 5pt{yf}}

\boxing{\vbox{yf}}
\boxing{\vbox to -5pt{yf}}
\boxing{\vbox spread 0pt{yf}}
\boxing{\vbox spread 5pt{yf}}

\boxing{\vbox{yf}}
\boxing{\vtop{yf}}
```



There is another box, `\vtop`

#vbox-rp, #vtop-rp

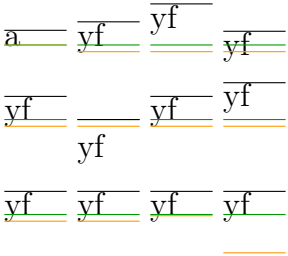
“Vertical” boxes

TEX-way

```
\boxing{\vbox{a}}
\boxing{\vbox{yf}}
\boxing{\vbox to 15pt{yf}}
\boxing{\vbox to 5pt{yf}}

\boxing{\vbox{yf}}
\boxing{\vbox to -5pt{yf}}
\boxing{\vbox spread 0pt{yf}}
\boxing{\vbox spread 5pt{yf}}

\boxing{\vbox{yf}}
\boxing{\vtop{yf}}
\boxing{\vtop spread -2pt{yf}}
\boxing{\vtop to 20pt{yf}}
```



it will change the depth for you

#vbox-rp, #vtop-rp

Move boxes

```
Write as
↪  L\raise0.5ex\hbox{A}T\lower0.5ex\hbox{E}X
```

Write as LATEX

\raise and \lower for horizontal boxes

#raise-rp #lower-rp #moveleft-rp #moveright-rp

Move boxes

```
Write as
↪  L\raise0.5ex\hbox{A}T\lower0.5ex\hbox{E}X
```

Write as L^{A}T$_{\mathrm{E}}$X

```
\vspace{3ex}
\vbox{L}
\moveleft0.5em\vbox{A}
\vbox{T}
\moveright0.5em\vbox{E}
\vbox{X}
```

L
A
T
E
X

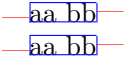
`\moveleft` and `\moveright` for vertical boxes

#raise-rp #lower-rp #moveleft-rp #moveright-rp

LaTeX boxes

Horizontal boxes

```
\boxing{\mbox{aa bb}}
\boxing{\makebox{aa bb}}
```

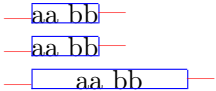


`\mbox` and `\makebox` are like `\hbox`

man: 20.1 Boxes#makebox_and_mbox

Horizontal boxes

```
\boxing{\mbox{aa bb}}
\boxing{\makebox{aa bb}}
\boxing{\makebox[20mm]{aa bb}}
```



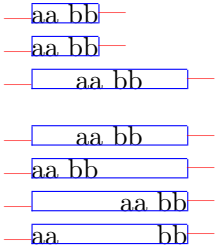
`\makebox` has a width as an optional param

man: 20.1 Boxes#makebox_and_mbox

Horizontal boxes

```
\boxing{\mbox{aa bb}}
\boxing{\makebox{aa bb}}
\boxing{\makebox[20mm]{aa bb}}

\boxing{\makebox[20mm][c]{aa bb}}
\boxing{\makebox[20mm][l]{aa bb}}
\boxing{\makebox[20mm][r]{aa bb}}
\boxing{\makebox[20mm][s]{aa bb}}
```



... and `\makebox` has text location as second optional param

man: 20.1 Boxes#makebox_and_mbox

Paragraph boxes

```
Space space \parbox{8em}{wanna go to
↪  space yes please space. Space
↪  space.} Go to space.
```

wanna go to space
Space space yes please space. Go to space.
Space space.

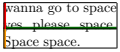
\parbox give you a box of text with some width. Also there are \pbox and minipage envirument

Paragraph boxes

```
Space space \parbox{8em}{wanna go to
↪  space yes please space. Space
↪  space.} Go to space.
```

```
\boxingDim{\parbox{8em}{wanna go to
↪  space yes please space. Space
↪  space.}}
```

wanna go to space
Space space yes please space. Go to space.
Space space.

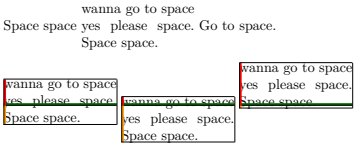


It is just a box with a big depth

man: 20.3 man: 8.1873 Boxes#parbox,_minipage,_and_pbox

Paragraph boxes

```
Space space \parbox{8em}{wanna go to
↪  space yes please space. Space
↪  space.} Go to space.
\boxingDim{\parbox{8em}{wanna go to
↪  space yes please space. Space
↪  space.}}
\boxingDim{\parbox[t]{8em}{wanna go to
↪  space yes please space. Space
↪  space.}}
\boxingDim{\parbox[b]{8em}{wanna go to
↪  space yes please space. Space
↪  space.}}
```



You can specify the position.

\parbox is useful when you want to put two lines to some command, that accepts only one line. Footnotes in the lectures use it.

man: 20.3 man: 8.1873 Boxes#parbox,_minipage,_and_pbox

Boxes-modifiers

```
\raisebox{0pt}[0pt][0pt]{\Large%
  \textbf{Aaaa\raisebox{-0.3ex}{a}%
    \raisebox{-0.7ex}{aa}%
    \raisebox{-1.2ex}{r}%
    \raisebox{-2.2ex}{g}%
    \raisebox{-4.5ex}{h}
  }
}
he shouted.
```

Aaaaaaaargh he shouted.

```
\rotatebox{45}{A}
```

A

```
\scalebox{2}{A}
```

`\raisebox{lift}[height][depth]{text}` change the text position. `\rotatebox` rotates the text, `\scalebox` scales it

man: 20.4 man: 22.3.2 man: 22.3.3 Boxes#box_modifiers

Box manipulation

Define box `\newsavebox{\<boxname>}`
Set box `\savebox`
Provide the content without deleting it: `\usebox`
Dimentions:

1. Create a length variable: `\newlength`
2. Set the variable to dimention of the box CONTENT:
   - width: `\settowidth{\<len-var>}{\usebox{\<box>}}`
   - height: `\settoheight{\<len-var>}{\usebox{\<box>}}`
   - depth: `\settodepth{\<len-var>}{\usebox{\<box>}}`

man: 14.4-6 man: 20.5, 20.7

Box manipulation

Example

```
\newsavebox{\mybox}
\savebox{\mybox}{\hbox{LaTeX content}}

\newlength{\boxwidth}
\settowidth{\boxwidth}{\usebox{\mybox}}
\newlength{\boxheight}
\settoheight{\boxheight}{\usebox{\mybox}}
\newlength{\boxdepth}
\settodepth{\boxdepth}{\usebox{\mybox}}

the box width \the\boxwidth
the box height \the\boxheight
the box depth \the\boxdepth

provide the content: \usebox{\mybox}
```

the box width 65.13pt
the box height 6.83pt
the box depth 0.10999pt
provide the content: LaTeX content

`\the` here is just to PRINT the length. Not the part of the command.

Box manipulation

Define box `\newbox\<boxname>`
Set box `\setbox\<boxname>=<box>`
Provide the content without deleting it: `\copy\<boxname>`
Provide the content with delete from memory: `\box\<boxname>`
Dimentions: width: `\wd`, height: `\ht`, depth: `\dp`

Box manipulation

Example

```
\newbox\mybox
\setbox\mybox=\hbox{TeX content}

the box width \the\wd\mybox
the box height \the\ht\mybox
the box depth \the\dp\mybox

provide the content: \copy\mybox
provide the content and free the box:
↪  \box\mybox
```

the box width 53.88pt
the box height 6.83pt
the box depth 0.10999pt
provide the content: TeX content
provide the content and free the box: TeX content

`\the` here is just to PRINT the length. Not the part of the command.

What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

Spaces

glue and kern provides spaces between boxes.

```
G\hskip0em lu\hskip0.5em e and
↪ k\kern0em e\kern0.5em rn
↪ provides...
```

Glu e and ke rn provides...

What is glue

Glue is more than just “spaces” between the boxes.
Glue is a **tensile** spaces between boxes.
Glue syntax is: <normal-length> [plus <how-can-it-stretch>] [minus <how-can-it-shrink>].

\relax

Don't want unexpected adding to you glue? (Or to you commands?) Use \relax! \relax does nothing by itself but says TEX "This is the end of what you've been doing"

#relax-rp

Where glue adds implicitly?

Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.
Between words and sentences. Here are lots' of glue.

`\hbox spread 40pt {Between words and sentences. Here are lots' of glue.}` was used in range -40pt—40pt.

Between paragraphs, there is also a glue. Notice: between sentences the glue is bigger, than between words.

How to use glue in your own work

```
\TeX-way:
use \hskip2em plus 1em\relax to add
↪  horizontal glue

or in vertical mode
\vskip2em plus 1em\relax
\LaTeX-way:
use \hspace{2em plus 1em} to add
↪  horizontal glue

or in vertical mode
\vspace{2em plus 1em}

like this
```

For horizontal space use `\hskip` or `\hspace`

For vertical space use `\vskip` or `\vspace`.

TEX-way: use to add horizontal glue
or in vertical mode

LATEX-way: use to add horizontal glue
or in vertical mode

like this

#hskip-rp #vskip-rp man: 19.2 man: 19.14

use `\vspace*` if you want space at begin of the page!

Infinite glue

```
\hbox to 50mm{\hskip0em plus 1fil\relax
↪  1fil and 1fil \hskip0em plus
↪  1fil\relax}
\hbox to 50mm{\hskip0em plus 1fil\relax
↪  1fil and 2fil \hskip0em plus
↪  2fil\relax}
\hbox to 50mm{\hskip0em plus 1fill\relax
↪  fill and fill \hskip0em plus
↪  1fill\relax}
\hbox to 50mm{\hskip0em plus 1fill\relax
↪  fill vs fil \hskip0em plus
↪  999fil\relax}
\hbox to 50mm{\hskip0em plus 1filll\relax
↪  filll and filll \hskip0em plus
↪  1filll\relax}
\hbox to 50mm{\hskip0em plus 1filll\relax
↪  filll vs fill \hskip0em plus
↪  999fill\relax}
```

1fil and 1fil

1fil and 2fil

fill and fill

fill vs fil

filll and filll

filll vs fill

`fil`, `fill`, `filll` are infinity with different "power". Both "plus" and "minus" are allowed. Notice: alignment without tabbular!

Abbreviations

You can use:

`\hfil`	`\hfill`	`\hspace{\fil}`	`\hspace{\fill}`
`\vfil`	`\vfill`	`\vspace{\fil}`	`\vspace{\fill}`

Glue manipulation

TEX provides additional storage for glue and glue in math mode (that is sensible for math style). They have the same syntax as length, just with `\skip` or `\muskip` prefix/suffix

Glue manipulation

Define a glue `\newskip\<gluename>`
Set a glue `\<gluename>=<glue>`
Add a glue `\advance\<gluename> by <glue>`. Also there are `\multiply` and `\divide`. As well as `\glueexp`.
Show a glue `\the\<gluename>`

Glue manipulation

Example

```
\newskip\myskip % first glue
\myskip=15pt plus 5pt minus 3pt
\newskip\doubleskip % second glue
\doubleskip=\myskip
\multiply\doubleskip by 2 % it is double
↪  times bigger
\advance\doubleskip by 30pt plus 4pt
so, \\myskip is \the\myskip. While
↪  \\doubleskip is \the\doubleskip %
↪  print glue

|\hskip\myskip TEST \hskip\doubleskip | %
↪  show glue

\newmuskip\mathspace % for math mode
\mathspace=18mu plus 2mu minus 1mu

$$|\mskip\mathspace T |$$ % show glue in
↪  math mode
```

so,
myskip is 15.0pt plus 5.0pt minus 3.0pt. While
doubleskip is 60.0pt plus 14.0pt minus 6.0pt
| TEST |

$$|\quad T|$$

Glue manipulation

Define glue `\newlength{\<gluename>}`
Set glue `\setlength`
Add glue `\addtolength`.

Yes, it is the same as for length

Glue manipulation

Example

```
\newlength{\myskip} % first glue
\setlength{\myskip}{15pt plus 5pt minus
↪  3pt}
\newlength{\doubleskip} % second glue
\setlength{\doubleskip}{\myskip}
\multiply\doubleskip by 2 % it is double
↪  times bigger
\addtolength{\doubleskip}{30pt plus 4pt}

so, \\myskip is \the\myskip. While
↪  \\doubleskip is \the\doubleskip %
↪  print glue

|\hskip\myskip TEST \hskip\doubleskip | %
↪  show glue
```

so,
myskip is 15.0pt plus 5.0pt minus 3.0pt. While
doubleskip is 60.0pt plus 14.0pt minus 6.0pt
| TEST |

What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

What spaces we have?

`\qquad`

`\quad`

`\enspace`

`\ `

`\quad` is used below:

$x = y \quad$ if y=0

useful for math:

`\;`

`\>`

`\,`

`\!`

the last one is negative space

Phantoms

Phantoms have the same size as it's an argument without drawing.

```
\boxing{$\int\limits^a_b x d\!x$}
\boxing{\phantom{$\int\limits^a_b x
↪  d\!x$}}
\boxing{\hphantom{$\int\limits^a_b x
↪  d\!x$}}
\boxing{\vphantom{$\int\limits^a_b x
↪  d\!x$}}
\boxing{\strut}
\hfill
\boxing{\smash{$\int\limits^a_b x d\!x$}}
```

$\boxing{\int\limits^a_b x d\!x}$ $\boxing{}$ $\boxing{\hphantom{\int\limits^a_b x d\!x}}$ $\boxing{\vphantom{\int\limits^a_b x d\!x}}$ $\boxing{\strut}$ $\boxing{\smash{\int\limits^a_b x d\!x}}$

\phantom leaves both dimentions. \hphantom and \vphantom leaves only one dimention.

\strut is short for \vphantom{(}

\smash is using to leave only the horizontal coordinate of a formula

What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

Toks manipulation

TEX has addition registers for storing strings. They have `\toks` prefix/suffix. The difference between toks and storage inside macros are in extension (We mention the extension mechanism at the last lecture). In toks TEXstore tokens (unexpanded).

What we will know?

- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

Inserts

TEX has addition registers for storing floats. They have `\insert` prefix/suffix.

kn: 15 /527613/simple-example-of-using-newinsert-and-friends
But I failed to make a working example...

What we have learned today?

- Technical agreements
- Introduction
- How TEX "sees" the document
  - Modes
  - Boxes and glue
  - Paragraphs and pages creation
- TEX primitives
  - Commands (Macros)
  - Counters
  - Lengths
  - Boxes
  - Glue
  - Spaces
  - Toks
  - Inserts

references I

color from the footnotes corresponds to references' color.

- kn: Knuth "The TEXBook"
- lv: L'vovsky "Nabor i verstka v sisteme LATEX"
- lamport: Lamport. "LATEX. A Document Preparation System, User's Guide and Reference Manual"
- man: "LATEX2e: An unofficial reference manual" also at website https://latexref.xyz/
- : https://tex.stackexchange.com/questions
- : https://en.wikibooks.org/wiki/LaTeX
- : https://www.overleaf.com/learn/latex
- : https://www.tug.org/utilities/plain/cseq.html

Distribution

- the pdf-version of the presentation and all printed materials can be distributed under license Creative Commons Attribution-ShareAlike 4.0 https://creativecommons.org/licenses/by-sa/4.0/
- The source code of the presentation is available on https://github.com/Lavton/latexLectures and can be distributed under the MIT license https://en.wikipedia.org/wiki/MIT_License#License_terms