Project: Innovative Open Source Courses for Computer Science

Open source tools for text processing
Teaching Material

Jiří Rybička
Mendelova univerzita v Brně

29. 5. 2021

Project: Innovative Open Source Courses for Computer Science

This teaching material was written as one of the outputs of the project “Innovative Open Source Courses for Computer Science”, funded by the Erasmus+ grant no. 2019-1-PL01-KA203-065564. The project is coordinated by West Pomeranian University of Technology in Szczecin (Poland) and is implemented in partnership with Mendel University in Brno (Czech Republic) and University of Žilina (Slovak Republic). The project implementation timeline is September 2019 to December 2022.

Project information

Project was implemented under the Erasmus+.
Project name: **“Innovative Open Source courses for Computer Science curriculum”**
Project nr: **2019-1-PL01-KA203-065564**
Key Action: **KA2 – Cooperation for innovation and the exchange of good practices**
Action Type: **KA203 – Strategic Partnerships for higher education**

Consortium
ZACHODNIOPOMORSKI UNIWERSYTET TECHNOLOGICZNY W SZCZECINIE
MENDELOVA UNIVERZITA V BRNĚ
ŽILINSKÁ UNIVERZITA V ŽILINE

Erasmus+ Disclaimer
This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Copyright Notice

Co-funded by the
Erasmus+ Programme
of the European Union

Method of document processing

Open source tools for text processing

Jiří Rybička
Department of Informatics
FBE MENDELU in Brno
rybicka@mendelu.cz

Project: Innovative Open Source Courses for Computer Science

Funded by
the European Union

Contents

- A new approach to document processing

Contents

- A new approach to document processing
- Typography as a second step

Contents

- A new approach to document processing
- Typography as a second step
- Structural markup as a common tool

Contents

- A new approach to document processing
- Typography as a second step
- Structural markup as a common tool
- Open source implementation of documents

Document

- Document is composition of contents and format

Document

- Document is composition of contents and format
- Author – Designer – Typesetter

Document

- Document is composition of contents and format
- Author – Designer – Typesetter
- Elements detection in document

Document

- Document is composition of contents and format
- Author – Designer – Typesetter
- Elements detection in document
- Visual representation of document elements – typography

Implementation

- Structural markup

Implementation

- Structural markup
- Break of markup definitions from document

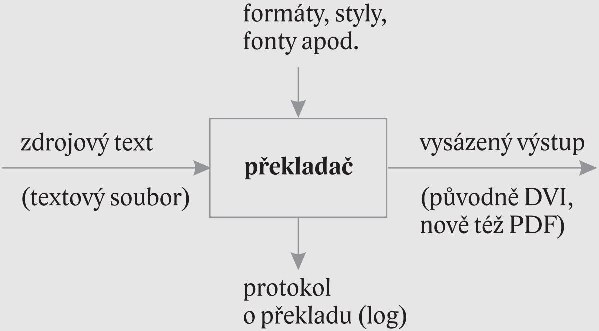
Implementation

- Structural markup
- Break of markup definitions from document
- Possibilities of structural markup in various systems

Implementation

- Structural markup
- Break of markup definitions from document
- Possibilities of structural markup in various systems
- Open source systems for text processing

- Basic principle of TEX-like systems



- Brief TEX history

TeX

- Brief TeX history
- Extensions (LaTeX, XeTeX, XeLaTeX), distributions

TEX

- Brief TEX history
- Extensions (LATEX, XƎTEX, XƎLATEX), distributions
- TEXonWeb, Overleaf

- Brief TEX history
- Extensions (LATEX, XƎTEX, XƎLATEX), distributions
- TEXonWeb, Overleaf
- TEX macro language: active characters, commands, macros

TEX

- Brief TEX history
- Extensions (LATEX, XƎTEX, XƎLATEX), distributions
- TEXonWeb, Overleaf
- TEX macro language: active characters, commands, macros
- XƎLATEX: commands, parameters, scope (groups, environments)

TeX

- Brief TeX history
- Extensions (LaTeX, XeTeX, XeLaTeX), distributions
- TeXonWeb, Overleaf
- TeX macro language: active characters, commands, macros
- XeLaTeX: commands, parameters, scope (groups, environments)
- Possibilities of new command definitions

TeX

- Brief TeX history
- Extensions (LaTeX, XeTeX, XeLaTeX), distributions
- TeXonWeb, Overleaf
- TeX macro language: active characters, commands, macros
- XeLaTeX: commands, parameters, scope (groups, environments)
- Possibilities of new command definitions
- Document implementation, styles and definition of structural markup

- My first document (overview), work with TEXonWeb

  `tex.mendelu.cz/new`; `tex.mendelu.cz/new/auth`

- My first document (overview), work with TeXonWeb

  `tex.mendelu.cz/new`; `tex.mendelu.cz/new/auth`
- Styles (predefined, user defined)

- My first document (overview), work with TEXonWeb

  `tex.mendelu.cz/new`; `tex.mendelu.cz/new/auth`
- Styles (predefined, user defined)
- Definition of new commands (macros) – basic

- My first document (overview), work with TeXonWeb

  `tex.mendelu.cz/new`; `tex.mendelu.cz/new/auth`
- Styles (predefined, user defined)
- Definition of new commands (macros) – basic
- Compilation, log file, errors

Basic font – types

- Font types: monospace/proportional; 3 categories

Basic font – types

- Font types: monospace/proportional; 3 categories
- Serif fonts – basic text in printed documents

Basic font – types

- Font types: monospace/proportional; 3 categories
- Serif fonts – basic text in printed documents
- Sans-serif fonts – second font in printed documents, primary font in electronic documents

Basic font – types

- Font types: monospace/proportional; 3 categories
- Serif fonts – basic text in printed documents
- Sans-serif fonts – second font in printed documents, primary font in electronic documents
- Other fonts: occasional printed or electronic matter, such as invitations, announcements, advertisements

Basic font – types

- Font types: monospace/proportional; 3 categories
- Serif fonts – basic text in printed documents
- Sans-serif fonts – second font in printed documents, primary font in electronic documents
- Other fonts: occasional printed or electronic matter, such as invitations, announcements, advertisements
- Optimal solution: one document – one font type

Basic font – types

- Font types: monospace/proportional; 3 categories
- Serif fonts – basic text in printed documents
- Sans-serif fonts – second font in printed documents, primary font in electronic documents
- Other fonts: occasional printed or electronic matter, such as invitations, announcements, advertisements
- Optimal solution: one document – one font type
- Mixing font types: basic text is serif, headings, titles etc. are sans-serif

Basic font – point sizes

- Font size: font parameter derived from metal typesetting systems

Basic font – point sizes

- Font size: font parameter derived from metal typesetting systems
- Measuring systems: basic unit of measure is **typographic point – pt**

Basic font – point sizes

- Font size: font parameter derived from metal typesetting systems
- Measuring systems: basic unit of measure is **typographic point – pt**
- English system: 1 pt = 0.353 mm

Basic font – point sizes

- Font size: font parameter derived from metal typesetting systems
- Measuring systems: basic unit of measure is **typographic point – pt**
- English system: 1 pt = 0.353 mm
- Bigger unit: 1 pica = 12 pt

Basic font – point sizes

- Font size: font parameter derived from metal typesetting systems
- Measuring systems: basic unit of measure is **typographic point – pt**
- English system: 1 pt = 0.353 mm
- Bigger unit: 1 pica = 12 pt
- Font point size: basic text in books: 10–12 pt

Basic font – point sizes

- Font size: font parameter derived from metal typesetting systems
- Measuring systems: basic unit of measure is **typographic point – pt**
- English system: 1 pt = 0.353 mm
- Bigger unit: 1 pica = 12 pt
- Font point size: basic text in books: 10–12 pt
- Other sizes: footnotes 8 pt, headings 12–24 pt

Plain typesetting

- Plain typesetting is part of a document with one font type, font face and font size.

Plain typesetting

- Plain typesetting is part of a document with one font type, font face and font size.
- Special characters – spaces, hyphens, dashes, quotes...

Plain typesetting

- Plain typesetting is part of a document with one font type, font face and font size.
- Special characters – spaces, hyphens, dashes, quotes...
- Any special character has its own shape, space and placement in the text

Plain typesetting

- Plain typesetting is part of a document with one font type, font face and font size.
- Special characters – spaces, hyphens, dashes, quotes...
- Any special character has its own shape, space and placement in the text
- It depends on used language, some characters differ in different languages (e.g. quotes)

Plain typesetting

- Plain typesetting is part of a document with one font type, font face and font size.
- Special characters – spaces, hyphens, dashes, quotes...
- Any special character has its own shape, space and placement in the text
- It depends on used language, some characters differ in different languages (e.g. quotes)
- Language and typographic rules define proper shape and placement

Mixed and paragraph typesetting

Open source tools for text processing

Jiří Rybička
Department of Informatics
FBE MENDELU in Brno
rybicka@mendelu.cz

Project: Innovative Open Source Courses for Computer Science

Funded by
the European Union

Font types

- Font types in mixed typesetting – optimal is less than 3 types

Font types

- Font types in mixed typesetting – optimal is less than 3 types
- Additional font – for headings, captions, table data etc.

Font types

- Font types in mixed typesetting – optimal is less than 3 types
- Additional font – for headings, captions, table data etc.
- Usual implementation for printed documents: Basic font is serif, additional font is sans-serif from the same family or from *visual compatible* family.

Font types

- Font types in mixed typesetting – optimal is less than 3 types
- Additional font – for headings, captions, table data etc.
- Usual implementation for printed documents: Basic font is serif, additional font is sans-serif from the same family or from *visual compatible* family.
- Visual compatibility: ideal solution is special couple straight by professional foundry (e.g. Baskerville + John Sans by F. Štorm)

Font types

- Font types in mixed typesetting – optimal is less than 3 types
- Additional font – for headings, captions, table data etc.
- Usual implementation for printed documents: Basic font is serif, additional font is sans-serif from the same family or from *visual compatible* family.
- Visual compatibility: ideal solution is special couple straight by professional foundry (e.g. Baskerville + John Sans by F. Štorm)
- Advantages of sans-serif additional font: emphasized and good readable in a short scope (heading on the one line, short captions, page headings etc.)

Font types – implementation

- In XƎLaTeX: `\fontspec{type}`

Font types – implementation

- In XƎLaTeX: `\fontspec{type}`
- Any installed font is available, font formats: TTF, OTF, Adobe Type 1

Font types – implementation

- In XƎLaTeX: `\fontspec{type}`
- Any installed font is available, font formats: TTF, OTF, Adobe Type 1
- Optional parameters: `\fontspec[options]{type}`

Font types – implementation

- In XƎLATEX: `\fontspec{type}`
- Any installed font is available, font formats: TTF, OTF, Adobe Type 1
- Optional parameters: `\fontspec[options]{type}`
- Widely used parameter: `[Mapping=tex-text]` – use automatic ligatures for dashes etc. like TEX standard font (Computer/Latin Modern)

Typefaces

- Typefaces – modification of basic shape of font

Typefaces

- Typefaces – modification of basic shape of font
- Modification of slope (italic, slanted)

Typefaces

- Typefaces – modification of basic shape of font
- Modification of slope (italic, slanted)
- Modification of weight (thin, light, semibold, bold, heavy, black)

Typefaces

- Typefaces – modification of basic shape of font
- Modification of slope (italic, slanted)
- Modification of weight (thin, light, semibold, bold, heavy, black)
- Modification of character width (compressed, extended)

Typefaces

- Typefaces – modification of basic shape of font
- Modification of slope (italic, slanted)
- Modification of weight (thin, light, semibold, bold, heavy, black)
- Modification of character width (compressed, extended)
- Modification of stroke (decorative, outlined)

Typefaces

- Typefaces – modification of basic shape of font
- Modification of slope (italic, slanted)
- Modification of weight (thin, light, semibold, bold, heavy, black)
- Modification of character width (compressed, extended)
- Modification of stroke (decorative, outlined)
- Combination: bold italic, bold extended, light compressed etc.

Typefaces

- Typefaces – modification of basic shape of font
- Modification of slope (italic, slanted)
- Modification of weight (thin, light, semibold, bold, heavy, black)
- Modification of character width (compressed, extended)
- Modification of stroke (decorative, outlined)
- Combination: bold italic, bold extended, light compressed etc.
- Special case: small caps

How to emphasize

- According to typographic rules we emphasize by italic typeface

How to emphasize

- According to typographic rules we emphasize by italic typeface
- The LaTeX command is `\emph{text}` This command is usable into other `\emph{}` command and use plain typeface (according to typographic rules)

How to emphasize

- According to typographic rules we emphasize by italic typeface
- The LaTeX command is `\emph{text}` This command is usable into other `\emph{}` command and use plain typeface (according to typographic rules)
- Change of typeface – two ways: command with parameter like `\textbf{text}` and command as a switch – working inside group, like `{\bfseries text}`

How to emphasize

- According to typographic rules we emphasize by italic typeface
- The LaTeX command is `\emph{text}` This command is usable into other `\emph{}` command and use plain typeface (according to typographic rules)
- Change of typeface – two ways: command with parameter like `\textbf{text}` and command as a switch – working inside group, like `{\bfseries text}`
- Other commands for typeface changes see textbook

Point sizes

- Basic point size is 10 pt, other sizes are determined by a `\documentclass` command and its optional parameter `[11pt]` or `[12pt]`

Point sizes

- Basic point size is 10 pt, other sizes are determined by a `\documentclass` command and its optional parameter `[11pt]` or `[12pt]`
- Document elements with various type sizes can be used as a predefined commands like `\section` or `\footnote`

Point sizes

- Basic point size is 10 pt, other sizes are determined by a `\documentclass` command and its optional parameter `[11pt]` or `[12pt]`
- Document elements with various type sizes can be used as a predefined commands like `\section` or `\footnote`
- Point size setting is available by set of switch commands with relative point sizes: `\tiny`; `\scriptsize`; `\footnotesize`; `\small`; `\normalsize`; `\large`; `\Large`; `\LARGE`; `\huge` and `\Huge`

Point sizes

- Basic point size is 10 pt, other sizes are determined by a `\documentclass` command and its optional parameter `[11pt]` or `[12pt]`
- Document elements with various type sizes can be used as a predefined commands like `\section` or `\footnote`
- Point size setting is available by set of switch commands with relative point sizes: `\tiny`; `\scriptsize`; `\footnotesize`; `\small`; `\normalsize`; `\large`; `\Large`; `\LARGE`; `\huge` and `\Huge`
- Any absolute point size can be set by command `\fontsize{size}{line spacing}`

Point sizes

- Basic point size is 10 pt, other sizes are determined by a `\documentclass` command and its optional parameter `[11pt]` or `[12pt]`
- Document elements with various type sizes can be used as a predefined commands like `\section` or `\footnote`
- Point size setting is available by set of switch commands with relative point sizes: `\tiny`; `\scriptsize`; `\footnotesize`; `\small`; `\normalsize`; `\large`; `\Large`; `\LARGE`; `\huge` and `\Huge`
- Any absolute point size can be set by command `\fontsize{size}{line spacing}`
- This command have to be followed by `\selectfont` command

Paragraph parameters

- Geometric parameters (see figure in the textbook): paragraph skip, special indent, left/right margin, line spacing, alignment

Paragraph parameters

- Geometric parameters (see figure in the textbook): paragraph skip, special indent, left/right margin, line spacing, alignment
- Standard behavior: Indent 0 pt, special indent 15 pt, left and right margin 0 pt (full width of typesetting), line spacing 12 pt for 10-point size text; alignment justify

Paragraph parameters

- Geometric parameters (see figure in the textbook): paragraph skip, special indent, left/right margin, line spacing, alignment
- Standard behavior: Indent 0 pt, special indent 15 pt, left and right margin 0 pt (full width of typesetting), line spacing 12 pt for 10-point size text; alignment justify
- Parameters are *lengths*. Lengths are stored in length registers

Paragraph parameters

- Geometric parameters (see figure in the textbook): paragraph skip, special indent, left/right margin, line spacing, alignment
- Standard behavior: Indent 0 pt, special indent 15 pt, left and right margin 0 pt (full width of typesetting), line spacing 12 pt for 10-point size text; alignment justify
- Parameters are *lengths*. Lengths are stored in length registers
- Lengths are solid and flexible

Length units

- The TEX system has unique length units system. It includes the Didôt European system, English system, inches, metric system and special unit “scaled point”, relative units em and ex.

Length units

- The TEX system has unique length units system. It includes the Didôt European system, English system, inches, metric system and special unit “scaled point”, relative units em and ex.
- Names of all units is given in following table:

Length units

- The TEX system has unique length units system. It includes the Didôt European system, English system, inches, metric system and special unit “scaled point”, relative units em and ex.
- Names of all units is given in following table:
-

name	*abbrev.*	*note*
English old point	pt	0,351 mm
Monotype point (big point)	bp	0,353 mm
pica	pc	1 pc = 12 pt
European Didôt point	dd	0,376 mm
cicero	cc	1 cc = 12 dd
inch	in	1 in = 25,4 mm
centimeter	cm	
milimeter	mm	
scaled point	sp	65 536 sp = 1 pt

Length registers

- Registers may be predefined or user defined. The use of register value is simple – only write the name of register

Length registers

- Registers may be predefined or user defined. The use of register value is simple – only write the name of register
- Length register definition: `\newlength\newname` (solid) or `\newdimen\newname` (flexible)

Length registers

- Registers may be predefined or user defined. The use of register value is simple – only write the name of register
- Length register definition: `\newlength\newname` (solid) or `\newdimen\newname` (flexible)
- Default value after definition of register is 0 pt

Length registers

- Registers may be predefined or user defined. The use of register value is simple – only write the name of register
- Length register definition: `\newlength\newname` (solid) or `\newdimen\newname` (flexible)
- Default value after definition of register is 0 pt
- Length setting (solid): `\register=length`; the equal sign is optional

Length registers

- Registers may be predefined or user defined. The use of register value is simple – only write the name of register
- Length register definition: `\newlength\newname` (solid) or `\newdimen\newname` (flexible)
- Default value after definition of register is 0 pt
- Length setting (solid): `\register=length`; the equal sign is optional
- Length setting (flexible): `\register=length plus X minus Y` where X and Y are lengths with any unit

Length registers

- Registers may be predefined or user defined. The use of register value is simple – only write the name of register
- Length register definition: `\newlength\newname` (solid) or `\newdimen\newname` (flexible)
- Default value after definition of register is 0 pt
- Length setting (solid): `\register=length`; the equal sign is optional
- Length setting (flexible): `\register=length plus X minus Y` where X and Y are lengths with any unit
- Value of any register may be multiplied by a constant, e.g. `3\register` is three times of register value, or `-0.5\register` is a half of register value

Length registers

- Registers may be predefined or user defined. The use of register value is simple – only write the name of register
- Length register definition: `\newlength\newname` (solid) or `\newdimen\newname` (flexible)
- Default value after definition of register is 0 pt
- Length setting (solid): `\register=length`; the equal sign is optional
- Length setting (flexible): `\register=length plus X minus Y` where X and Y are lengths with any unit
- Value of any register may be multiplied by a constant, e.g. `3\register` is three times of register value, or `-0.5\register` is a half of register value
- Add to length: `\addtolength\register by length`

Paragraph settings

- Predefined registers are available: `\parskip` (flexible), `\parindent`, `\baselineskip`, `\leftskip`, `\rightskip` (all solid)

Paragraph settings

- Predefined registers are available: `\parskip` (flexible), `\parindent`, `\baselineskip`, `\leftskip`, `\rightskip` (all solid)
- Change of geometric parameters: e.g.
  `\parskip=0.5\baselineskip plus 2pt minus 1pt`
  or
  `\parindent=2em` (relative; `2×` of actual point size)

Paragraph settings

- Predefined registers are available: `\parskip` (flexible), `\parindent`, `\baselineskip`, `\leftskip`, `\rightskip` (all solid)
- Change of geometric parameters: e.g. `\parskip=0.5\baselineskip plus 2pt minus 1pt` or `\parindent=2em` (relative; 2× of actual point size)
- The `\baselineskip` is not available for given changing – it is changed by redefining of coefficient `\baselinestretch` from value 1 to any other value, e.g. `\def\baselinestretch{1.3}`

Paragraph settings

- Predefined registers are available: `\parskip` (flexible), `\parindent`, `\baselineskip`, `\leftskip`, `\rightskip` (all solid)
- Change of geometric parameters: e.g.
  `\parskip=0.5\baselineskip plus 2pt minus 1pt`
  or
  `\parindent=2em` (relative; 2× of actual point size)
- The `\baselineskip` is not available for given changing – it is changed by redefining of coefficient `\baselinestretch` from value 1 to any other value, e.g. `\def\baselinestretch{1.3}`
- Paragraph aligning is set by three environments: `flushleft`, `flushright` and `center`

Page design

Open source tools for text processing

Jiří Rybička
Department of Informatics
FBE MENDELU in Brno
rybicka@mendelu.cz

Project: Innovative Open Source Courses for Computer Science

Funded by
the European Union

Page break

- Any document is divided into pages

Page break

- Any document is divided into pages
- Page break is allowed only in some places

Page break

- Any document is divided into pages
- Page break is allowed only in some places
- Widow and orphan setting: `\widowpenalty=n` is penalty on page break after the first line of paragraph

Page break

- Any document is divided into pages
- Page break is allowed only in some places
- Widow and orphan setting: `\widowpenalty=n` is penalty on page break after the first line of paragraph
- `\clubpenalty=n` is a penalty on page break before the last line of paragraph

Page break

- Any document is divided into pages
- Page break is allowed only in some places
- Widow and orphan setting: `\widowpenalty=n` is penalty on page break after the first line of paragraph
- `\clubpenalty=n` is a penalty on page break before the last line of paragraph
- *n* is integer between 0 (always) and 10000 (never)

Page break

- Any document is divided into pages
- Page break is allowed only in some places
- Widow and orphan setting: `\widowpenalty=n` is penalty on page break after the first line of paragraph
- `\clubpenalty=n` is a penalty on page break before the last line of paragraph
- *n* is integer between 0 (always) and 10000 (never)
- Unconditional page break: `\newpage` or `\clearpage` or `\cleardoublepage`

Material on the page

- All content of a page is divided into three parts: page heading, main part and page foot

Material on the page

- All content of a page is divided into three parts: page heading, main part and page foot
- Common page design is set by `\pagestyle{X}` command, where X is: `plain`, `headings`, `myheadings` or `empty`

Material on the page

- All content of a page is divided into three parts: page heading, main part and page foot
- Common page design is set by `\pagestyle{X}` command, where X is: `plain`, `headings`, `myheadings` or `empty`
- The `\pagestyle` should be placed into document preamble and it affects all followed pages

Material on the page

- All content of a page is divided into three parts: page heading, main part and page foot
- Common page design is set by `\pagestyle{X}` command, where X is: `plain`, `headings`, `myheadings` or `empty`
- The `\pagestyle` should be placed into document preamble and it affects all followed pages
- For page design setting of individual page can be used `\thispagestyle{X}` with the same options

Material on the page

- All content of a page is divided into three parts: page heading, main part and page foot
- Common page design is set by `\pagestyle{X}` command, where X is: `plain`, `headings`, `myheadings` or `empty`
- The `\pagestyle` should be placed into document preamble and it affects all followed pages
- For page design setting of individual page can be used `\thispagestyle{X}` with the same options
- Material into headings is set by section commands or explicitly by `\markright{text}` or `\markboth{left text}{right text}`

Material on the page

- All content of a page is divided into three parts: page heading, main part and page foot
- Common page design is set by `\pagestyle{X}` command, where X is: `plain`, `headings`, `myheadings` or `empty`
- The `\pagestyle` should be placed into document preamble and it affects all followed pages
- For page design setting of individual page can be used `\thispagestyle{X}` with the same options
- Material into headings is set by section commands or explicitly by `\markright{text}` or `\markboth{left text}{right text}`
- The footnote is automatically set by command `\footnote{text}`

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`
- This command works only between paragraphs

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`
- This command works only between paragraphs
- The star-variant works on the beginning and the end of page

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`
- This command works only between paragraphs
- The star-variant works on the beginning and the end of page
- Horizontal space: `\hspace{length}` or `\hspace*{length}`

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`
- This command works only between paragraphs
- The star-variant works on the beginning and the end of page
- Horizontal space: `\hspace{length}` or `\hspace*{length}`
- The star-variant works on the beginning and the end of line

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`
- This command works only between paragraphs
- The star-variant works on the beginning and the end of page
- Horizontal space: `\hspace{length}` or `\hspace*{length}`
- The star-variant works on the beginning and the end of line
- Special length: `\fill` has zero natural length and is infinitely expandable

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`
- This command works only between paragraphs
- The star-variant works on the beginning and the end of page
- Horizontal space: `\hspace{length}` or `\hspace*{length}`
- The star-variant works on the beginning and the end of line
- Special length: `\fill` has zero natural length and is infinitely expandable
- `\hspace{\fill}` can be abbreviated to `\hfill`

Vertical and horizontal spaces

- Vertical space: `\vspace{any length}` or `\vspace*{any length}`
- This command works only between paragraphs
- The star-variant works on the beginning and the end of page
- Horizontal space: `\hspace{length}` or `\hspace*{length}`
- The star-variant works on the beginning and the end of line
- Special length: `\fill` has zero natural length and is infinitely expandable
- `\hspace{\fill}` can be abbreviated to `\hfill`
- `\vspace{\fill}` can be abbreviated to `\vfill`

Sections

- The whole document can be divided into smaller parts: sections

Sections

- The whole document can be divided into smaller parts: sections
- Section headings are supported by a couple of similar commands

Sections

- The whole document can be divided into smaller parts: sections
- Section headings are supported by a couple of similar commands
- `\section{text}` is top level in `article` document class

Sections

- The whole document can be divided into smaller parts: sections
- Section headings are supported by a couple of similar commands
- `\section{text}` is top level in `article` document class
- `\chapter{text}` is top level in `book` and `report` document classes

Sections

- The whole document can be divided into smaller parts: sections
- Section headings are supported by a couple of similar commands
- `\section{text}` is top level in `article` document class
- `\chapter{text}` is top level in `book` and `report` document classes
- Next levels: `\subsection{}`; `\subsubsection{}`; `\paragraph{}` and `\subparagraph{}`

Sections

- The whole document can be divided into smaller parts: sections
- Section headings are supported by a couple of similar commands
- `\section{text}` is top level in `article` document class
- `\chapter{text}` is top level in `book` and `report` document classes
- Next levels: `\subsection{}`; `\subsubsection{}`; `\paragraph{}` and `\subparagraph{}`
- Each of section heading commands solves 4 tasks: a) visual shape of heading; b) numbering of section; c) material into page headings; d) material into table of contents

Starred sections

- Each of section heading commands have star-version – this variant solves only visual shape of heading

Starred sections

- Each of section heading commands have star-version – this variant solves only visual shape of heading
- Numbering of sections can be solved by manipulation with appropriate counter

Starred sections

- Each of section heading commands have star-version – this variant solves only visual shape of heading
- Numbering of sections can be solved by manipulation with appropriate counter
- Material into page headings can be set by `\markright` or `\markboth` command

Starred sections

- Each of section heading commands have star-version – this variant solves only visual shape of heading
- Numbering of sections can be solved by manipulation with appropriate counter
- Material into page headings can be set by `\markright` or `\markboth` command
- Material into table of contents can be set by `\addcontentsline{file}{level}{text}` command

Starred sections

- Each of section heading commands have star-version – this variant solves only visual shape of heading
- Numbering of sections can be solved by manipulation with appropriate counter
- Material into page headings can be set by `\markright` or `\markboth` command
- Material into table of contents can be set by `\addcontentsline{file}{level}{text}` command
- File (extension of file) can be `toc` for standard table of contents information, or `lof` for standard list of figures, or `lot` for standard list of tables

Starred sections

- Each of section heading commands have star-version – this variant solves only visual shape of heading
- Numbering of sections can be solved by manipulation with appropriate counter
- Material into page headings can be set by `\markright` or `\markboth` command
- Material into table of contents can be set by `\addcontentsline{file}{level}{text}` command
- File (extension of file) can be `toc` for standard table of contents information, or `lof` for standard list of figures, or `lot` for standard list of tables
- Level can be `section`, `subsection` etc.

Starred sections

- Each of section heading commands have star-version – this variant solves only visual shape of heading
- Numbering of sections can be solved by manipulation with appropriate counter
- Material into page headings can be set by `\markright` or `\markboth` command
- Material into table of contents can be set by `\addcontentsline{file}{level}{text}` command
- File (extension of file) can be `toc` for standard table of contents information, or `lof` for standard list of figures, or `lot` for standard list of tables
- Level can be `section`, `subsection` etc.
- Any material into table of contents can be inserted by `\addtocontents{file}{text}`

Numbering

- Each numbering is connected with **counter**

Numbering

- Each numbering is connected with **counter**
- Counter is variable for integer value

Numbering

- Each numbering is connected with **counter**
- Counter is variable for integer value
- Counters are predefined or user defined

Numbering

- Each numbering is connected with **counter**
- Counter is variable for integer value
- Counters are predefined or user defined
- Manipulation with counters: set value; display value; add value to counter; step value by 1; step value by 1 and set the label; use value in expressions

Numbering

- Each numbering is connected with **counter**
- Counter is variable for integer value
- Counters are predefined or user defined
- Manipulation with counters: set value; display value; add value to counter; step value by 1; step value by 1 and set the label; use value in expressions
- Predefined counters are connected with some commands, e.g. `page` for page numbering, `footnote` for numbering of footnotes, `section` for section numbering

Numbering

- Each numbering is connected with **counter**
- Counter is variable for integer value
- Counters are predefined or user defined
- Manipulation with counters: set value; display value; add value to counter; step value by 1; step value by 1 and set the label; use value in expressions
- Predefined counters are connected with some commands, e.g. `page` for page numbering, `footnote` for numbering of footnotes, `section` for section numbering
- Display value of counter is available via `\the`counter, e.g. `\thepage` or `\thesection`

Manipulation with counters

- User defined counter: `\newcounter{name}`

Manipulation with counters

- User defined counter: `\newcounter{name}`
- Automatically is created corresponding command `\thename`

Manipulation with counters

- User defined counter: `\newcounter{name}`
- Automatically is created corresponding command `\thename`
- Default value of new counter is zero

Manipulation with counters

- User defined counter: `\newcounter{name}`
- Automatically is created corresponding command `\thename`
- Default value of new counter is zero
- Set any value: `\setcounter{name}{value}`

Manipulation with counters

- User defined counter: `\newcounter{name}`
- Automatically is created corresponding command `\thename`
- Default value of new counter is zero
- Set any value: `\setcounter{name}{value}`
- Add value to counter: `\addtocounter{name}{value}`

Manipulation with counters

- User defined counter: `\newcounter{name}`
- Automatically is created corresponding command `\thename`
- Default value of new counter is zero
- Set any value: `\setcounter{name}{value}`
- Add value to counter: `\addtocounter{name}{value}`
- Step value by 1: `\stepcounter{name}`

Manipulation with counters

- User defined counter: `\newcounter{name}`
- Automatically is created corresponding command `\thename`
- Default value of new counter is zero
- Set any value: `\setcounter{name}{value}`
- Add value to counter: `\addtocounter{name}{value}`
- Step value by 1: `\stepcounter{name}`
- `\refstepcounter{name}` adds the one to counter and sets label to the new value of counter (usable for cross references)

Display variants

- Each counter can be used (displayed) into any text of document

Display variants

- Each counter can be used (displayed) into any text of document
- Command `\thename` (without parameters) places output shape of counter value

Display variants

- Each counter can be used (displayed) into any text of document
- Command `\thename` (without parameters) places output shape of counter value
- Output shape can be changed by redefining of `\thename` command

Display variants

- Each counter can be used (displayed) into any text of document
- Command `\thename` (without parameters) places output shape of counter value
- Output shape can be changed by redefining of `\thename` command
- Available output shapes are: `\arabic{counter}` (default); `\alph{}` (small letters); `\Alph{}` (capital letters); `\roman{}` (roman number with small letters); `\Roman{}` (roman number with capital letters); `\fnsymbol{}` (symbols for footnotes)

Display variants

- Each counter can be used (displayed) into any text of document
- Command `\thename` (without parameters) places output shape of counter value
- Output shape can be changed by redefining of `\thename` command
- Available output shapes are: `\arabic{counter}` (default); `\alph{}` (small letters); `\Alph{}` (capital letters); `\roman{}` (roman number with small letters); `\Roman{}` (roman number with capital letters); `\fnsymbol{}` (symbols for footnotes)
- Example: `\def\thesection{\Roman{section}}` redefines arabic numbers of sections to roman numbers with capital letters

Dependency of counters

- One counter can be set as dependent to other counter. If superior counter is stepped, dependent counter is set to zero.

Dependency of counters

- One counter can be set as dependent to other counter. If superior counter is stepped, dependent counter is set to zero.
- Any other changing method of superior counter don’t affected dependent counter (`\addtocounter`, `\setcounter`)

Dependency of counters

- One counter can be set as dependent to other counter. If superior counter is stepped, dependent counter is set to zero.
- Any other changing method of superior counter don’t affected dependent counter (`\addtocounter`, `\setcounter`)
- Definition of dependency:
  `\newcounter{name}[superior]` – new counter `name` will be dependent on counter `superior`

Dependency of counters

- One counter can be set as dependent to other counter. If superior counter is stepped, dependent counter is set to zero.
- Any other changing method of superior counter don’t affected dependent counter (`\addtocounter`, `\setcounter`)
- Definition of dependency:

  `\newcounter{name}[superior]` – new counter `name` will be dependent on counter `superior`
- Expression of dependency in output value: for example

  `\def\thename{\thesuperior:\arabic{name}}` sets display of value with current value of superior counter separated by colon

Typesetting of mathematics

Open source tools for text processing

Jiří Rybička
Department of Informatics
FBE MENDELU in Brno
rybicka@mendelu.cz

Project: Innovative Open Source Courses for Computer Science

Funded by
the European Union

Math

- Rules for math typesetting are more strict than rules for plain text

Math

- Rules for math typesetting are more strict than rules for plain text
- Math has huge amount of various symbols and each of them has its own shape, spacing and method of place into expression

Math

- Rules for math typesetting are more strict than rules for plain text
- Math has huge amount of various symbols and each of them has its own shape, spacing and method of place into expression
- Each symbol have to by the same shape in display equation, text equation or in paragraph text, sizes may differ only

Math

- Rules for math typesetting are more strict than rules for plain text
- Math has huge amount of various symbols and each of them has its own shape, spacing and method of place into expression
- Each symbol have to by the same shape in display equation, text equation or in paragraph text, sizes may differ only
- All math symbols have to by placed into math environment

Math

- Rules for math typesetting are more strict than rules for plain text
- Math has huge amount of various symbols and each of them has its own shape, spacing and method of place into expression
- Each symbol have to by the same shape in display equation, text equation or in paragraph text, sizes may differ only
- All math symbols have to by placed into math environment
- TeX and its extensions have wide support for math typesetting; it is difficult to find a system that would make this better

Math

- Rules for math typesetting are more strict than rules for plain text
- Math has huge amount of various symbols and each of them has its own shape, spacing and method of place into expression
- Each symbol have to by the same shape in display equation, text equation or in paragraph text, sizes may differ only
- All math symbols have to by placed into math environment
- TeX and its extensions have wide support for math typesetting; it is difficult to find a system that would make this better
- Math typesetting was said to be the main reason to develop the TeX (Knuth)

Math environments

- They are two ways to present math expressions: **text math** (inside a paragraph) or **display math** (between paragraphs)

Math environments

- They are two ways to present math expressions: **text math** (inside a paragraph) or **display math** (between paragraphs)
- So TeX has two basic math environments: `$...$` (text math) and `$$...$$` (display math)

Math environments

- They are two ways to present math expressions: **text math** (inside a paragraph) or **display math** (between paragraphs)
- So TEX has two basic math environments: `$...$` (text math) and `$$...$$` (display math)
- The LATEX adds two more environments and adds variant commands for the two basic environments

Math environments

- They are two ways to present math expressions: **text math** (inside a paragraph) or **display math** (between paragraphs)
- So TeX has two basic math environments: `$...$` (text math) and `$$...$$` (display math)
- The LaTeX adds two more environments and adds variant commands for the two basic environments
- The text math can be bounded with `\(` and `\)` or with `\begin{math}...\end{math}`

Math elements

Math environments

Math symbols and elements

Math environments

- They are two ways to present math expressions: **text math** (inside a paragraph) or **display math** (between paragraphs)
- So TeX has two basic math environments: `$...$` (text math) and `$$...$$` (display math)
- The LaTeX adds two more environments and adds variant commands for the two basic environments
- The text math can be bounded with `\(` and `\)` or with `\begin{math}...\end{math}`
- The display math can be bounded with `\[` and `\]` or with `\begin{displaymath}...\end{displaymath}`

LaTeX math environments

- Advanced LaTeX math environmens are `equation` and `eqnarray`

LaTeX math environments

- Advanced LaTeX math environmens are `equation` and `eqnarray`
- The `\begin{equation}...\end{equation}` environment numbers this display math equation

LaTeX math environments

- Advanced LaTeX math environmens are `equation` and `eqnarray`
- The `\begin{equation}...\end{equation}` environment numbers this display math equation
- The `equation` counter is connected with this environment

LaTeX math environments

- Advanced LaTeX math environmens are `equation` and `eqnarray`
- The `\begin{equation}...\end{equation}` environment numbers this display math equation
- The `equation` counter is connected with this environment
- The counter is automatically stepped with each placed environment and can be referenced

LaTeX math environments

- Advanced LaTeX math environmens are `equation` and `eqnarray`
- The `\begin{equation}...\end{equation}` environment numbers this display math equation
- The `equation` counter is connected with this environment
- The counter is automatically stepped with each placed environment and can be referenced
- Example:

  `\begin{equation} c^2= a^2+b^2\end{equation}`

  yields

$$c^2 = a^2 + b^2 \tag{1}$$

LaTeX math environments

- The `eqnarray` math environment is intended for systems of equations and allows vertical align of three parts

LaTeX math environments

- The `eqnarray` math environment is intended for systems of equations and allows vertical align of three parts
- One part is on the left, the other in the middle and the third on the right; parts are divided by `&`

LaTeX math environments

- The `eqnarray` math environment is intended for systems of equations and allows vertical align of three parts
- One part is on the left, the other in the middle and the third on the right; parts are divided by `&`
- The left part is aligned to the right, the middle part is centered and the right part is aligned to the left

LaTeX math environments

- The `eqnarray` math environment is intended for systems of equations and allows vertical align of three parts
- One part is on the left, the other in the middle and the third on the right; parts are divided by `&`
- The left part is aligned to the right, the middle part is centered and the right part is aligned to the left
- Simple example:

```
\begin{eqnarray}
    c^2 & = & a^2+b^2\\
    c   & = &\sqrt{a^2+b^2}
\end{eqnarray}
```

yields

$$\begin{aligned} c^2 &= a^2+b^2 \qquad (2)\\ c &= \sqrt{a^2+b^2} \qquad (3)\end{aligned}$$

LaTeX math environments

- Each equation in `eqnarray` environment is numbered. To suppress of numbering can be used `\nonumber` command after the end of appropriate line

LaTeX math environments

- Each equation in `eqnarray` environment is numbered. To suppress of numbering can be used `\nonumber` command after the end of appropriate line
- The `\\` command is used to divide system into separate lines

LaTeX math environments

- Each equation in `eqnarray` environment is numbered. To suppress of numbering can be used `\nonumber` command after the end of appropriate line
- The `\\` command is used to divide system into separate lines
- The `\\` command has optional parameter as usually: `\\[distance]` to add vertical space between lines

LaTeX math environments

- Each equation in `eqnarray` environment is numbered. To suppress of numbering can be used `\nonumber` command after the end of appropriate line
- The `\\` command is used to divide system into separate lines
- The `\\` command has optional parameter as usually: `\\[distance]` to add vertical space between lines
- To suppress any numbering of the whole equation system can be used a `eqnarray*` environment

LaTeX math environments

- Each equation in `eqnarray` environment is numbered. To suppress of numbering can be used `\nonumber` command after the end of appropriate line
- The `\\` command is used to divide system into separate lines
- The `\\` command has optional parameter as usually: `\\[distance]` to add vertical space between lines
- To suppress any numbering of the whole equation system can be used a `eqnarray*` environment
- More information about vertical aligning see `array` environment

Math symbols and elements

- Math rules: math variables are typeset by math italic typeface (default typeface in any math environment)

Math symbols and elements

- Math rules: math variables are typeset by math italic typeface (default typeface in any math environment)
- Constants, functions, total differencial and some other cases are typeset by upshape typeface

Math elements

Math environments

Math symbols and elements

Math symbols and elements

- Math rules: math variables are typeset by math italic typeface (default typeface in any math environment)
- Constants, functions, total differencial and some other cases are typeset by upshape typeface
- Matrices, vectors and similar structures are typeset by bold typeface

Math symbols and elements

- Math rules: math variables are typeset by math italic typeface (default typeface in any math environment)
- Constants, functions, total differencial and some other cases are typeset by upshape typeface
- Matrices, vectors and similar structures are typeset by bold typeface
- So we need to switch typeface in some cases: `\mathrm{}` for upshape typeface, `\mathbf{}` for bold and `\mathit{}` for math italic

Math symbols and elements

- Math rules: math variables are typeset by math italic typeface (default typeface in any math environment)
- Constants, functions, total differencial and some other cases are typeset by upshape typeface
- Matrices, vectors and similar structures are typeset by bold typeface
- So we need to switch typeface in some cases: `\mathrm{}` for upshape typeface, `\mathbf{}` for bold and `\mathit{}` for math italic
- Many of symbols are defined as a command – its shape is properly displayed

Math symbols and elements

- Math rules: math variables are typeset by math italic typeface (default typeface in any math environment)
- Constants, functions, total differencial and some other cases are typeset by upshape typeface
- Matrices, vectors and similar structures are typeset by bold typeface
- So we need to switch typeface in some cases: `\mathrm{}` for upshape typeface, `\mathbf{}` for bold and `\mathit{}` for math italic
- Many of symbols are defined as a command – its shape is properly displayed
- Example: `$$C=A(\cos\alpha+\mathrm{i}\sin\alpha)=A\mathrm{e}^{\mathrm{i}\alpha}$$`

$$C=A(\cos\alpha+\mathrm{i}\sin\alpha)=A\mathrm{e}^{\mathrm{i}\alpha}$$

Math elements

- Fractions: `\frac{X}{Y}` yields

$$\frac{X}{Y}$$

Math elements

- Fractions: `\frac{X}{Y}` yields

$$\frac{X}{Y}$$

- Roots: `\sqrt[n]{xyz}` yields

$$\sqrt[n]{xyz}$$

Math elements

Math environments

Math symbols and elements

Math elements

- Fractions: `\frac{X}{Y}` yields

$$\frac{X}{Y}$$

- Roots: `\sqrt[n]{xyz}` yields

$$\sqrt[n]{xyz}$$

- Indices and exponents: `a_1^3-a_{2x}^{3b}` yields

$$a_1^3-a_{2x}^{3b}$$

Math elements

- Sums, limits, integrals...:

```
\sum_{a=1}^N x_a\cdot w_a
\lim_{x\rightarrow\infty}\frac{x+3}{x-1}
\int_0^\infty f(x)\mathrm{d}x
```

$$\textstyle\sum_{a=1}^N x_a\cdot w_a$$
$$\textstyle\lim_{x\rightarrow\infty}\frac{x+3}{x-1}$$
$$\textstyle\int_0^\infty f(x)\mathrm{d}x$$

Matrices, large delimiters

- Matrix is implemented as an `array` environment (see more in `tabular` environment)

Matrices, large delimiters

- Matrix is implemented as an `array` environment (see more in `tabular` environment)
- Various parts of expressions may be bounded by large delimiters (braces etc.)

Matrices, large delimiters

- Matrix is implemented as an `array` environment (see more in `tabular` environment)
- Various parts of expressions may be bounded by large delimiters (braces etc.)
- Commands `\left(` and `\right)` typesets braces around expression

Matrices, large delimiters

- Matrix is implemented as an `array` environment (see more in `tabular` environment)
- Various parts of expressions may be bounded by large delimiters (braces etc.)
- Commands `\left(` and `\right)` typesets braces around expression
- Simple example:

```
\mathbf{A}=\left( % left large delimiter
\begin{array}{cc} % matrix, two centered columns
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{array}\right)
```

$$\mathbf{A}=\left(\begin{array}{cc} a_{11} & a_{12} \\ a_{21} & a_{22} \end{array}\right)$$

Small overview of other symbols

- Greek alphabet: `\alpha` α `\beta` β `\gamma` γ `\delta` δ `\omega` ω `\phi` ϕ `\varphi` φ `\Delta` Δ `\Omega` Ω ...

Small overview of other symbols

- Greek alphabet: `\alpha` α `\beta` β `\gamma` γ `\delta` δ `\omega` ω `\phi` ϕ `\varphi` φ `\Delta` Δ `\Omega` Ω ...
- Operators: `\cdot` $\cdot$ `\bullet` $\bullet$ `\circ` $\circ$ `\pm` $\pm$ `\times` $\times$ `\diamond` $\diamond$ `\cap` $\cap$ `\cup` $\cup$ `\oplus` $\oplus$ `\dagger` $\dagger$ ...

Small overview of other symbols

- Greek alphabet: `\alpha` α `\beta` β `\gamma` γ `\delta` δ `\omega` ω `\phi` ϕ `\varphi` φ `\Delta` Δ `\Omega` Ω ...
- Operators: `\cdot` $\cdot$ `\bullet` $\bullet$ `\circ` $\circ$ `\pm` $\pm$ `\times` $\times$ `\diamond` $\diamond$ `\cap` $\cap$ `\cup` $\cup$ `\oplus` $\oplus$ `\dagger` $\dagger$ ...
- Relations: `\leq` $\leq$ `\geq` $\geq$ `\in` $\in$ `\sim` $\sim$ `\approx` $\approx$ `\equiv` $\equiv$ `\subset` $\subset$ `\supset` $\supset$ `\ll` $\ll$ `\gg` $\gg$ ...

Small overview of other symbols

- Greek alphabet: `\alpha` α `\beta` β `\gamma` γ `\delta` δ `\omega` ω `\phi` ϕ `\varphi` φ `\Delta` Δ `\Omega` Ω ...
- Operators: `\cdot` $\cdot$ `\bullet` $\bullet$ `\circ` $\circ$ `\pm` $\pm$ `\times` $\times$ `\diamond` $\diamond$ `\cap` $\cap$ `\cup` $\cup$ `\oplus` $\oplus$ `\dagger` $\dagger$ ...
- Relations: `\leq` $\leq$ `\geq` $\geq$ `\in` $\in$ `\sim` $\sim$ `\approx` $\approx$ `\equiv` $\equiv$ `\subset` $\subset$ `\supset` $\supset$ `\ll` $\ll$ `\gg` $\gg$ ...
- Arrows: `\leftarrow` $\leftarrow$ `\rightarrow` $\rightarrow$ `\Leftarrow` $\Leftarrow$ `\longleftarrow` $\longleftarrow$ `\longleftrightarrow` $\longleftrightarrow$ `\uparrow` $\uparrow$ `\mapsto` $\mapsto$ `\nearrow` $\nearrow$ `\swarrow` $\swarrow$ ...

Small overview of other symbols

- Greek alphabet: `\alpha` α `\beta` β `\gamma` γ `\delta` δ `\omega` ω `\phi` ϕ `\varphi` φ `\Delta` Δ `\Omega` Ω ...
- Operators: `\cdot` $\cdot$ `\bullet` $\bullet$ `\circ` $\circ$ `\pm` $\pm$ `\times` $\times$ `\diamond` $\diamond$ `\cap` $\cap$ `\cup` $\cup$ `\oplus` $\oplus$ `\dagger` $\dagger$ ...
- Relations: `\leq` $\leq$ `\geq` $\geq$ `\in` $\in$ `\sim` $\sim$ `\approx` $\approx$ `\equiv` $\equiv$ `\subset` $\subset$ `\supset` $\supset$ `\ll` $\ll$ `\gg` $\gg$ ...
- Arrows: `\leftarrow` $\leftarrow$ `\rightarrow` $\rightarrow$ `\Leftarrow` $\Leftarrow$ `\longleftarrow` $\longleftarrow$ `\longleftrightarrow` $\longleftrightarrow$ `\uparrow` $\uparrow$ `\mapsto` $\mapsto$ `\nearrow` $\nearrow$ `\swarrow` $\swarrow$ ...
- Functions: `\sin` $\sin$ `\ln` $\ln$ `\inf` $\inf$ `\liminf` $\liminf$ `\max` $\max$ `\dim` $\dim$ `\arctan` $\arctan$ `\gcd` $\gcd$ `\lg` $\lg$ ...

Small overview of other symbols

- Greek alphabet: `\alpha` α `\beta` β `\gamma` γ `\delta` δ `\omega` ω `\phi` ϕ `\varphi` φ `\Delta` Δ `\Omega` Ω ...
- Operators: `\cdot` $\cdot$ `\bullet` $\bullet$ `\circ` $\circ$ `\pm` $\pm$ `\times` $\times$ `\diamond` $\diamond$ `\cap` $\cap$ `\cup` $\cup$ `\oplus` $\oplus$ `\dagger` $\dagger$ ...
- Relations: `\leq` $\leq$ `\geq` $\geq$ `\in` $\in$ `\sim` $\sim$ `\approx` $\approx$ `\equiv` $\equiv$ `\subset` $\subset$ `\supset` $\supset$ `\ll` $\ll$ `\gg` $\gg$ ...
- Arrows: `\leftarrow` $\leftarrow$ `\rightarrow` $\rightarrow$ `\Leftarrow` $\Leftarrow$ `\longleftarrow` $\longleftarrow$ `\longleftrightarrow` $\longleftrightarrow$ `\uparrow` $\uparrow$ `\mapsto` $\mapsto$ `\nearrow` $\nearrow$ `\swarrow` $\swarrow$ ...
- Functions: `\sin` $\sin$ `\ln` $\ln$ `\inf` $\inf$ `\liminf` $\liminf$ `\max` $\max$ `\dim` $\dim$ `\arctan` $\arctan$ `\gcd` $\gcd$ `\lg` $\lg$ ...
- Big delimiters: `\{` $\big\{$ `\lfloor` $\big\lfloor$ `\rfloor` $\big\rfloor$ `\lceil` $\big\lceil$ `\rceil` $\big\rceil$ `\langle` $\big\langle$ `\rangle` $\big\rangle$ `\|` $\big\|$ `\Uparrow` $\big\Uparrow$ ...

Small overview of other symbols

- Greek alphabet: `\alpha` α `\beta` β `\gamma` γ `\delta` δ `\omega` ω `\phi` ϕ `\varphi` φ `\Delta` Δ `\Omega` Ω ...
- Operators: `\cdot` $\cdot$ `\bullet` $\bullet$ `\circ` $\circ$ `\pm` $\pm$ `\times` $\times$ `\diamond` $\diamond$ `\cap` $\cap$ `\cup` $\cup$ `\oplus` $\oplus$ `\dagger` $\dagger$ ...
- Relations: `\leq` $\leq$ `\geq` $\geq$ `\in` $\in$ `\sim` $\sim$ `\approx` $\approx$ `\equiv` $\equiv$ `\subset` $\subset$ `\supset` $\supset$ `\ll` $\ll$ `\gg` $\gg$ ...
- Arrows: `\leftarrow` $\leftarrow$ `\rightarrow` $\rightarrow$ `\Leftarrow` $\Leftarrow$ `\longleftarrow` $\longleftarrow$ `\longleftrightarrow` $\longleftrightarrow$ `\uparrow` $\uparrow$ `\mapsto` $\mapsto$ `\nearrow` $\nearrow$ `\swarrow` $\swarrow$ ...
- Functions: `\sin` $\sin$ `\ln` $\ln$ `\inf` $\inf$ `\liminf` $\liminf$ `\max` $\max$ `\dim` $\dim$ `\arctan` $\arctan$ `\gcd` $\gcd$ `\lg` $\lg$ ...
- Big delimiters: `\{` $\{$ `\lfloor` $\lfloor$ `\rfloor` $\rfloor$ `\lceil` $\lceil$ `\rceil` $\rceil$ `\langle` $\langle$ `\rangle` $\rangle$ `\|` $\|$ `\Uparrow` $\Uparrow$ ...
- Other: `\aleph` $\aleph$ `\forall` $\forall$ `\infty` ∞ `\nabla` ∇ `\surd` $\surd$ `\flat` $\flat$ `\backslash` $\backslash$ `\partial` ∂ `\clubsuit` $\clubsuit$ ...

Tables, figures

Open source tools for text processing

Jiří Rybička
Department of Informatics
FBE MENDELU in Brno
rybicka@mendelu.cz

Project: Innovative Open Source Courses for Computer Science

Funded by
the European Union

Vertical align, tabbing

- To vertical align can be used the `tabbing` or `tabular` environments

Vertical align, tabbing

- To vertical align can be used the `tabbing` or `tabular` environments
- The `tabbing` environment is model of tab stops

Vertical align, tabbing

- To vertical align can be used the `tabbing` or `tabular` environments
- The `tabbing` environment is model of tab stops
- This environment is useful for simple open tables only

Vertical align, tabbing

- To vertical align can be used the `tabbing` or `tabular` environments
- The `tabbing` environment is model of tab stops
- This environment is useful for simple open tables only
- Basic principle: set tab stop `\=` and reference tab stop `\>`

Vertical align, tabbing

- To vertical align can be used the `tabbing` or `tabular` environments
- The `tabbing` environment is model of tab stops
- This environment is useful for simple open tables only
- Basic principle: set tab stop `\=` and reference tab stop `\>`
- Small example: `\begin{tabbing}`

  ```
  City\hspace{30mm}\= Temperature \\
  New York \> 25 $^\circ$C \\
  Sydney   \> $-3$ $^\circ$C
  \end{tabbing}
  ```

City	Temperature
New York	25 $^\circ$C
Sydney	-3 $^\circ$C

The tabular environment

- The `tabular` environment is intended for close tables with rules and various alignments

The tabular environment

- The `tabular` environment is intended for close tables with rules and various alignments
- Common shape of table:

```
\begin{tabular}{columns} \hline
 tab field & tab field & ... \\ \hline
 tab field & tab field & ... \\ \hline
\end{tabular}
```

The tabular environment

- The `tabular` environment is intended for close tables with rules and various alignments
- Common shape of table:

```
\begin{tabular}{columns} \hline
 tab field & tab field & ... \\ \hline
 tab field & tab field & ... \\ \hline
\end{tabular}
```

- Columns are defined as a list of alignment letters: l, c, r (left, center, right respectively) with vertical rules |

The tabular environment

- The `tabular` environment is intended for close tables with rules and various alignments
- Common shape of table:

```
\begin{tabular}{columns} \hline
 tab field & tab field & ... \\ \hline
 tab field & tab field & ... \\ \hline
\end{tabular}
```

- Columns are defined as a list of alignment letters: l, c, r (left, center, right respectively) with vertical rules |
- Special case: a paragraph column `p{width}` – material in tab field is justified to given width

The tabular environment

- The `tabular` environment is intended for close tables with rules and various alignments
- Common shape of table:

```
\begin{tabular}{columns} \hline
 tab field & tab field & ... \\ \hline
 tab field & tab field & ... \\ \hline
\end{tabular}
```

- Columns are defined as a list of alignment letters: l, c, r (left, center, right respectively) with vertical rules |
- Special case: a paragraph column `p{width}` – material in tab field is justified to given width
- The `\hline` command yields horizontal rule after given tab line

Table example

- Simple table with rules and various aligning in columns: `\begin{tabular}{|r|l|c|} \hline`

```
\bfseries No.&\bfseries Name &\bfseries University\\\hline
 1 & Paweł Obłąk & ZUT, Szczecin, Polska \\
 7 & Žaneta Čižmářová & MENDELU, Brno, Česko \\
 12 & Vladimír Bôčik & ŽU, Žilina, Slovensko \\ \hline
\end{tabular}
```

No.	Name	University
1	Paweł Obłąk	ZUT, Szczecin, Polska
7	Žaneta Čižmářová	MENDELU, Brno, Česko
12	Vladimír Bôčik	ŽU, Žilina, Slovensko

Figures, schemas

- To insert any picture we can use two common ways: external file or internal drawing tools

Figures, schemas

- To insert any picture we can use two common ways: external file or internal drawing tools
- External graphics files are prepared by appropriate graphic editors, so its possibilities are unlimited

Figures, schemas

- To insert any picture we can use two common ways: external file or internal drawing tools
- External graphics files are prepared by appropriate graphic editors, so its possibilities are unlimited
- External files may be in vector or raster form

Figures, schemas

- To insert any picture we can use two common ways: external file or internal drawing tools
- External graphics files are prepared by appropriate graphic editors, so its possibilities are unlimited
- External files may be in vector or raster form
- To insert graphic file into document can be use the `\includegraphics{file}` command

Figures, schemas

- To insert any picture we can use two common ways: external file or internal drawing tools
- External graphics files are prepared by appropriate graphic editors, so its possibilities are unlimited
- External files may be in vector or raster form
- To insert graphic file into document can be use the `\includegraphics{file}` command
- This command is defined in package `graphicx`, so we have to link this package via `\usepackage` command

Figures, schemas

- To insert any picture we can use two common ways: external file or internal drawing tools
- External graphics files are prepared by appropriate graphic editors, so its possibilities are unlimited
- External files may be in vector or raster form
- To insert graphic file into document can be use the `\includegraphics{file}` command
- This command is defined in package `graphicx`, so we have to link this package via `\usepackage` command
- To change of some parameters of inserted file can be used optional parameters of `\includegraphics[pars]{file}`

Inserted graphics – examples

- File formats of inserted graphics may be PDF (vector), JPG and PNG (raster)

Inserted graphics – examples

- File formats of inserted graphics may be PDF (vector), JPG and PNG (raster)
- `\includegraphics[width=.3\textwidth]{logo.pdf}`

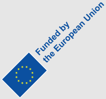
Inserted graphics – examples

- File formats of inserted graphics may be PDF (vector), JPG and PNG (raster)

-
  ```
  \includegraphics[width=.3\textwidth]{logo.pdf}
  ```

Funded by
the European Union

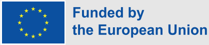
-
  ```
  \includegraphics[scale=0.05, angle=45]{logo.pdf}
  ```



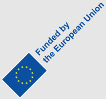
Inserted graphics – examples

- File formats of inserted graphics may be PDF (vector), JPG and PNG (raster)

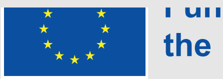
-
  ```
  \includegraphics[width=.3\textwidth]{logo.pdf}
  ```



-
  ```
  \includegraphics[scale=0.05, angle=45]{logo.pdf}
  ```



-
  ```
  \includegraphics[viewport=0 0 450 150,
      clip]{logo.pdf}
  ```



Drawing pictures

- The special LaTeX environment `picture` is intended for drawing of simple vector graphic; it is natural part of LaTeX system, no other package is needed

Drawing pictures

- The special LaTeX environment `picture` is intended for drawing of simple vector graphic; it is natural part of LaTeX system, no other package is needed
- A set of graphic commands are available in this environment

Drawing pictures

- The special LaTeX environment `picture` is intended for drawing of simple vector graphic; it is natural part of LaTeX system, no other package is needed
- A set of graphic commands are available in this environment
- Measure of graphic elements is set in `\unitlength` register; default value is 1 pt

Drawing pictures

- The special LaTeX environment `picture` is intended for drawing of simple vector graphic; it is natural part of LaTeX system, no other package is needed
- A set of graphic commands are available in this environment
- Measure of graphic elements is set in `\unitlength` register; default value is 1 pt
- A command `\put(X, Y){element}` puts given element to the workspace on coordinates X, Y

Drawing pictures

- The special LaTeX environment `picture` is intended for drawing of simple vector graphic; it is natural part of LaTeX system, no other package is needed
- A set of graphic commands are available in this environment
- Measure of graphic elements is set in `\unitlength` register; default value is 1 pt
- A command `\put(X, Y){element}` puts given element to the workspace on coordinates X, Y
- Size of workspace is given by parameters of `picture` environment

Drawing pictures

- The special LaTeX environment `picture` is intended for drawing of simple vector graphic; it is natural part of LaTeX system, no other package is needed
- A set of graphic commands are available in this environment
- Measure of graphic elements is set in `\unitlength` register; default value is 1 pt
- A command `\put(X, Y){element}` puts given element to the workspace on coordinates X, Y
- Size of workspace is given by parameters of `picture` environment
- Coordinates and size of workspace aren’t checked, so any element may be placed out of the workspace

Picture examples

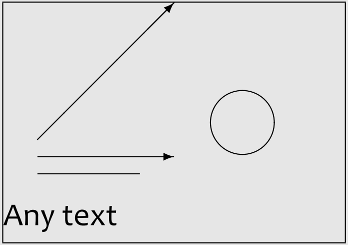
- Size of unit is set to 1 mm

```
\begin{picture}(100,70)
\put(0,5){Any text}
\put(10,20){\line(1,0){30}}
\put(10,25){\vector(1,0){40}}
\put(10,30){\vector(1,1){40}}
\put(0,0){\framebox(100,70){}}
\put(70,35){\circle{20}}
\end{picture}
```

Picture examples

- Size of unit is set to 1 mm

```
\begin{picture}(100,70)
\put(0,5){Any text}
\put(10,20){\line(1,0){30}}
\put(10,25){\vector(1,0){40}}
\put(10,30){\vector(1,1){40}}
\put(0,0){\framebox(100,70){}}
\put(70,35){\circle{20}}
\end{picture}
```

- 

Floating objects

- Sometimes it is not possible to place a picture or table in the appropriate place

Floating objects

- Sometimes it is not possible to place a picture or table in the appropriate place
- If the image or table does not fit in the rest of the page, a large hole would appear in the text

Floating objects

- Sometimes it is not possible to place a picture or table in the appropriate place
- If the image or table does not fit in the rest of the page, a large hole would appear in the text
- Floating environments are intended for this cases

Floating objects

- Sometimes it is not possible to place a picture or table in the appropriate place
- If the image or table does not fit in the rest of the page, a large hole would appear in the text
- Floating environments are intended for this cases
- Content of floating environment is placed to nearest appropriate place on next page(s)

Floating objects

- Sometimes it is not possible to place a picture or table in the appropriate place
- If the image or table does not fit in the rest of the page, a large hole would appear in the text
- Floating environments are intended for this cases
- Content of floating environment is placed to nearest appropriate place on next page(s)
- Algorithm for place of floating object is partially controlled by user specification

Floating objects

- Sometimes it is not possible to place a picture or table in the appropriate place
- If the image or table does not fit in the rest of the page, a large hole would appear in the text
- Floating environments are intended for this cases
- Content of floating environment is placed to nearest appropriate place on next page(s)
- Algorithm for place of floating object is partially controlled by user specification
- There are three floating environments – for tables, for figures and for marginal notes

Floating tables

- The `table` floating environment is available

Floating tables

- The `table` floating environment is available
- Simple example:

```
\begin{table}[htbp]
\caption{An example of floating table}
\begin{tabular}{|r|l|} \hline
 \bfseries No.& \bfseries Name \\
 ... etc. ...
\end{tabular}\end{table}
```

Floating tables

- The `table` floating environment is available
- Simple example:

```
\begin{table}[htbp]
\caption{An example of floating table}
\begin{tabular}{|r|l|} \hline
 \bfseries No.& \bfseries Name \\
 ... etc. ...
\end{tabular}\end{table}
```

- Specification in optional parameter: h – here (if fits), t – top of page, b – bottom of page, p – separate page; the order of the letters determines the priority

Floating tables

- The `table` floating environment is available
- Simple example:

```
\begin{table}[htbp]
\caption{An example of floating table}
\begin{tabular}{|r|l|} \hline
 \bfseries No.& \bfseries Name \\
 ... etc. ...
\end{tabular}\end{table}
```

- Specification in optional parameter: h – here (if fits), t – top of page, b – bottom of page, p – separate page; the order of the letters determines the priority
- The `\caption` command numbers tables with connected counter `table` and places the text of the caption into file `.lot` for list of tables

Floating pictures

- The system is similar to floating tables

Floating pictures

- The system is similar to floating tables
- Environment name is `figure`

Floating pictures

- The system is similar to floating tables
- Environment name is `figure`
- The environment has the same optional parameter as `table`

Floating pictures

- The system is similar to floating tables
- Environment name is `figure`
- The environment has the same optional parameter as `table`
- The same `\caption` command may be used

Floating pictures

- The system is similar to floating tables
- Environment name is `figure`
- The environment has the same optional parameter as `table`
- The same `\caption` command may be used
- Numbering of figures is done by a `figure` counter and caption text is placed into `.lof` file for list of figures

Floating pictures

- The system is similar to floating tables
- Environment name is `figure`
- The environment has the same optional parameter as `table`
- The same `\caption` command may be used
- Numbering of figures is done by a `figure` counter and caption text is placed into `.lof` file for list of figures
- The order of the tables and the order of the figures is never broken but tables and figures may be mixed