

MiKTeX and advanced L^AT_EX

Day 2

Marko Boon

October 3, 2006

Part 2: L^AT_EX for experienced users

- customising list structures
- automatic calculation of column widths
- use non-standard headers and footers
- change chapter and section title appearance
- change table of contents appearance
- create an interactive PDF file from your LaTeX document
- create a master index
- citations and a bibliography
- including programming statements in your LaTeX document
- create a PDF slide show or poster
- Using TrueType Fonts in L^AT_EX

Special Characters – Quotation Marks

Single quotes are produced with: `' '`

Double quotes are produced with: `" "`

Avoid using the double quote character `"`

```
He said: 'Hello world'.
```

```
He said: ``Hello world''.
```

```
He said: "Hello world".
```

He said: ‘Hello world’.

He said: “Hello world”.

He said: "Hello world".

Special Characters – Hyphens and Dashes

To create – characters of different length, just repeat the - character:

```
- is called hyphen
```

```
-- is called en dash
```

```
--- is called em dash
```

- is called hyphen

– is called en dash

— is called em dash

The minus sign is obtained by entering math mode (which will be discussed later):

```
$3 - 4 = -1$
```

$3 - 4 = -1$

Special Characters – Command Characters

As mentioned before, the characters # \$ ~ _ ^ { } % are interpreted as commands.

To print them as text, give a command consisting of \ plus that character:

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

\$ ~ _ ^ { } %

To print a backslash, use the command \textbackslash: \

Special Characters – Accents

Diacritical marks or accents can be created with \LaTeX :

```
\`e \'e \^o \^o \~o \=o \textcolor{v}{s} \textcolor{c}{c}  
be\"invloeden  
het re\"ele deel  
Cura\textcolor{c}{c}ao
```

è é ô ö õ š ç
beïnvloeden
het reële deel
Curaçao

Special Characters

The package `textcomp` defines a lot of special characters. First we have to load this package in the preamble:

```
\usepackage{textcomp}
```

Now we can use all the special characters:

```
\texteuro \copyright \textcelsius
```

€ © °C

Special Characters

Special symbols can be entered directly, but only if the right input encoding is specified. The input encoding depends on the type and language of the operating system. We have to load the package `inputenc` to specify the correct encoding:

```
\usepackage[ansinew]{inputenc}
```

beïnvloeden, reëel, Curaçao

■ f © ¥ §

beïnvloeden, reëel, Curaçao

€ f © ¥ §

Please note that some of these characters also require the `textcomp` package.

The Euro Symbol: €

The package `textcomp` defined the `\texteuro` command which shows a Euro symbol: €.

Adobe created a font containing better looking euro symbols which also contains bold, italic and serif versions. To use these symbols, load the package `europs`. Now we can use the following commands:

`\EUROfc` – creates the official Euro symbol: €

`\EUR{ }` – creates a Euro symbol depending on the current text style

Bold: €

Italic: €

Sans-serif: €

Displaying Text

Changing Font Style

The following commands and declarations change the current font style:

Command	Declaration	Result
<code>\emph</code>	<code>\em</code>	<i>emphasised</i>
<code>\textrm</code>	<code>\rmfamily</code>	Roman font family
<code>\texttt</code>	<code>\ttfamily</code>	Typewriter font family
<code>\textsf</code>	<code>\sffamily</code>	Sans serif font family
<code>\textup</code>	<code>\upshape</code>	Normal, upright font shape
<code>\textit</code>	<code>\itshape</code>	<i>Italic font shape</i>
<code>\textsl</code>	<code>\slshape</code>	<i>Slanted font shape</i>
<code>\textsc</code>	<code>\scshape</code>	SMALL CAPS FONT SHAPE
<code>\textbf</code>	<code>\bfseries</code>	Boldface font weight
<code>\textmd</code>	<code>\mdseries</code>	normal (medium) font weight

Changing Font Style – Example

```
This is normal text with one \textit{italic} word.  
{\sffamily This whole \itshape line is \textbf{sans}  
serif.}
```

```
\textbf{\textit{Bold and italic}}
```

Do you see the difference?

```
\emph{emphasised}, \textit{italic}, \textsl{slanted}
```

This is normal text with one *italic* word. This whole *line is sans serif*.

Bold and italic

Do you see the difference? *emphasised, italic, slanted*

This is an italic sentence containing an emphasised word.

Customising List Structures

The package `paralist` makes it easy to create compact list structures that can easily be configured. It defines three new environments:

```
\begin{compactdesc} \end{compactdesc}  
\begin{compactitem} \end{compactitem}  
\begin{compactenum} \end{compactenum}
```

Labels can be adjusted using the commands:

```
\setdefaultitem{level1}{level2}{level3}{level4}  
\setdefaultenum{level1}{level2}{level3}{level4}
```

Customising List Structures

Example:

```
\setdefaultitem{}{$\triangleright$}{$\blacksquare$}{}
```

```
\begin{compactitem}
```

```
\item first level
```

```
  \begin{compactitem}
```

```
    \item second level
```

```
  \end{compactitem}
```

```
\end{compactitem}
```



```
\setdefaultenum{1)}{a.}{i)}{A}
```

```
\begin{compactenum}
```

```
\item first level
```

```
  \begin{compactenum}
```

```
    \item second level
```

```
  \end{compactenum}
```

```
\end{compactenum}
```

Customising List Structures

- first level
 - ▷ second level
- i) first level
 - a. second level

Automatic Calculation of Column Widths

The `tabularx` package defines a new column type `X` which is similar to `p`, but whose width will be computed automatically.

```
\begin{tabularx}{\textwidth}{|l|c|X|X|}
...
\end{tabularx}
```

Left	Center	The width of this column will be computed automatically by the <code>tabularx</code> package	The width of this column will also be computed automatically. The width is equal to the width of the third column.
L	C	The width of this column will be computed automatically by the <code>tabularx</code> package	The width of this column will also be computed automatically.

Automatic Calculation of Column Widths

The `tabulary` package defines four new column types: `L`, `C`, `R`, `J` (= Justified). The widths of these columns will be based on their content.

```
\begin{tabulary}{\textwidth}{|l|c|L|J|}
...
\end{tabulary}
```

Left	Center	The width of this column will be computed automatically by the <code>tabulary</code> package.	The width of this column will also be computed automatically. The width is not equal to the width of the third column because this cell contains more text.
L	C	The width of this column will be computed automatically	The width of this column will also be computed automatically.

Non-standard headers and footers

The package `fancyhdr` (previously known as `fancyheadings`) defines commands that let you control headers and footers:

```
\lhead{...}      \chead{...}      \rhead{...}

\lfoot{...}      \cfoot{...}      \rfoot{...}
```

If you want to distinguish between odd and even pages, it becomes slightly more complicated:

```
\fancyhead[RO, LE]{...}

\fancyfoot[C]{...}
```

L = Left, C = left, R = right, O = odd, E = even.

You have to specify that `pagestyle` should be fancy (instead of plain or empty).

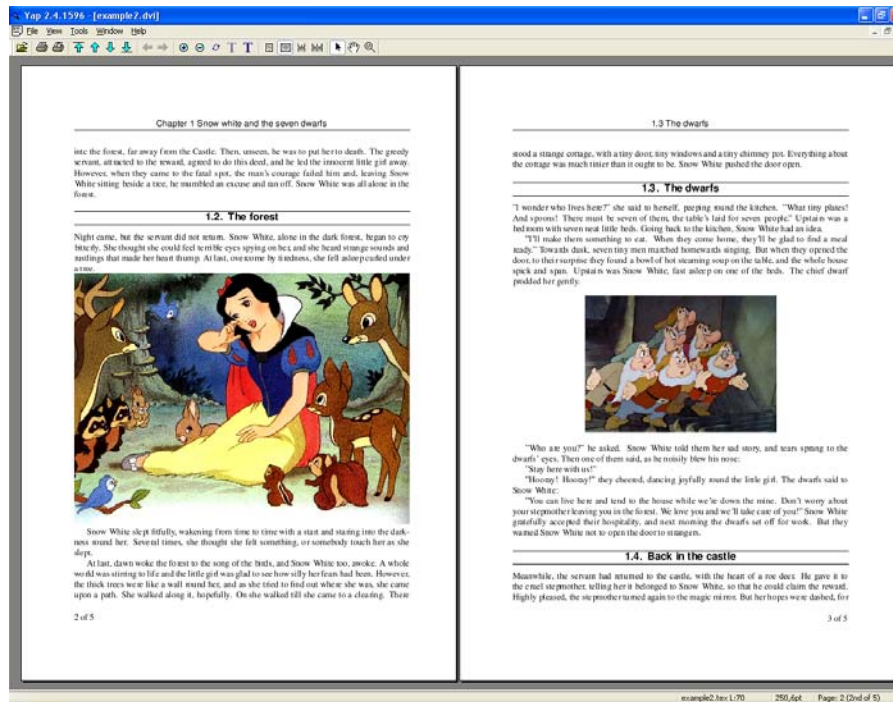
fancyhdr - Example

```
\pagestyle{fancy}
\fancyhf{}
\fancyhead[CE]{\sffamily\leftmark}
\fancyhead[CO]{\sffamily\rightmark}
\fancyfoot[RO]{\thepage\ of \pageref{LastPage}}
\fancyfoot[LE]{\thepage\ of \pageref{LastPage}}

\renewcommand\chaptermark[1]{%
  \markboth{\chaptername\ thechapter\ #1}{} }
\renewcommand\sectionmark[1]{%
  \markright{\thesection\ #1}}
\renewcommand\headrulewidth{0.4pt}
```

Please note that the `LastPage` reference is only available if you load the package `lastpage`. You have to `\TeX` your document twice before it works.

fancyhdr - Example



Chapter and section title appearance

The package `titlesec` lets you control the appearance of chapter, section, subsection and subsubsection headings. The general command is:

```
\titleformat{cmd}[shape]{fmt}{label}{sep}{bef}[aft]
```

- `cmd` the heading command name (`\chapter`, `\section`, ...).
- `shape` The basic shape for the heading. Predefined shapes are `hang` (section default), `runin` (paragraph default), `display` (chapter default), `frame` (display, framed), `leftmargin`, `rightmargin`.
- `fmt` the declarations that are applied to the whole title.
- `label` the formatting of the label (the heading number).
- `sep` the distance between the label and title text.
- `bef` code executed immediately preceding the heading text.
- `aft` optional code to be executed after formatting the heading.

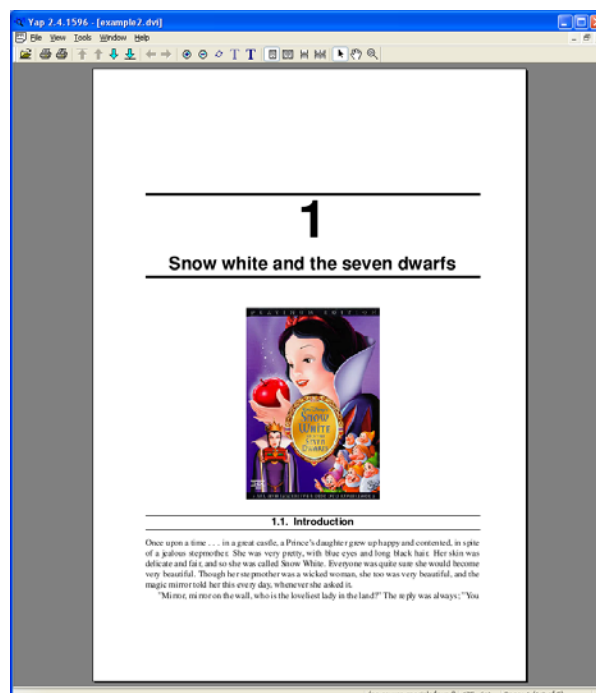
titlesec - Example

```

\usepackage{titlesec}
\titleformat{\chapter}[display]
  {\normalfont\sffamily\bfseries%
  \centering\huge}
  {{\titlerule[2pt]}\vspace{2mm}%
  \fontsize{72pt}{72pt}\selectfont%
  \thechapter}
  {0mm}{}[{\titlerule[2pt]}]
\titleformat{\section}[hang]
  {\normalfont\sffamily\bfseries%
  \centering\large\titlerule}
  {\thesection.}{1ex}{}[{\titlerule}]

```

titlesec - Example



Change table of contents appearance

Adding a line to the table of contents manually:

`\addcontentsline{ext}{type}{text}`

where **ext** is the extension of the file (toc, lof, lot), **type** is the kind of contents entry (chapter, section, figure, table) and **text** is the text to be written.

Example:

```
\clearpage
\phantomsection
\appendix
\addcontentsline{toc}{chapter}{Appendix}
```

This code adds the word "Appendix" to the table of contents. The command `\phantomsection` is only necessary if you use the package `hyperref`. It fixes a bug in `hyperref` (reference to the wrong page).

Change table of contents appearance

The command `\contentsname` defines the title of the table of contents (default: Contents). You can change it by redefining this command:

```
\renewcommand{\contentsname}{Table of contents}
```

Change table of contents appearance

The package `titletoc` provides commands to change the appearance of the table of contents:

```
\titlecontents{type}[left]{above}{format1}
               {format2}{pagenr}
```

- type** the type of contents line (chapter, section, ...).
- left** space left of the chapter/section name.
- above** code to be executed before the entry (usually adding vertical space).
- format1** format for numbered entries (regular chapters). Usually this ends with the command `\contentslabel{width}` where **width** is the width of the column with numbers.
- format2** format for numberless entries (like appendix).
- pagenr** the filler and page number. You can use the `\titlerule` command for dotted fillers.

titletoc - Example

```
\titlecontents{chapter}[0.7cm]
  {\vspace{0.3cm}}
  {\sffamily\bfseries\Large\contentslabel{0.7cm}}
  {\sffamily\bfseries\Large\hspace{-0.7cm}}
  {\sffamily\bfseries\Large\hfill\contentspage}

\titlecontents{section}[2cm]
  {}
  {\sffamily\contentslabel{1.3cm}}
  {}
  {\titlerule*[1mm]{.}\sffamily\contentspage}
```

titletoc - Example

Table of contents	
1	Snow white and the seven dwarfs 3
1.1	Introduction 3
1.2	Snow-white's youth 4
1.3	The forest 4
1.4	The dwarfs 5
1.5	Back at the castle 6
1.6	The first plan 6
1.7	The second plan 7
1.8	The death of Snow-white 8
1.9	The prince 8
1.10	The resurrection 8
	Appendix 10
	A Illustrations 10

Frontmatter and mainmatter

In books it is very common to start the "real" page numbering at the page of the first chapter. The pages before the first chapter have roman page numbering (i, ii, iii, iv, ...). The **book** document class defines two commands to ensure this.

Put this command right before `\begin{document}` to start roman page numbering:

```
\frontmatter
```

Put this command right before the first chapter to start regular page numbering (starts at 1):

```
\mainmatter
```

You can manually set the page number with the command

```
\setcounter{page}{17}
```

Create an interactive PDF file

If you load the package **hyperref**, the DVI and PDF file created from your \LaTeX document will be interactive. Added interactivity is:

- all internal references will be clickable (including the table of contents).
- you can create external links (e.g. to web pages).
- you can create a list of bookmarks in your PDF document.

Usage is simple: just add the line

```
\usepackage{hyperref}
```

to your document (always load this package after all the other packages!).

A useful new command, similar to the HTML ` ` command is:

```
\href{URL}{description}
```

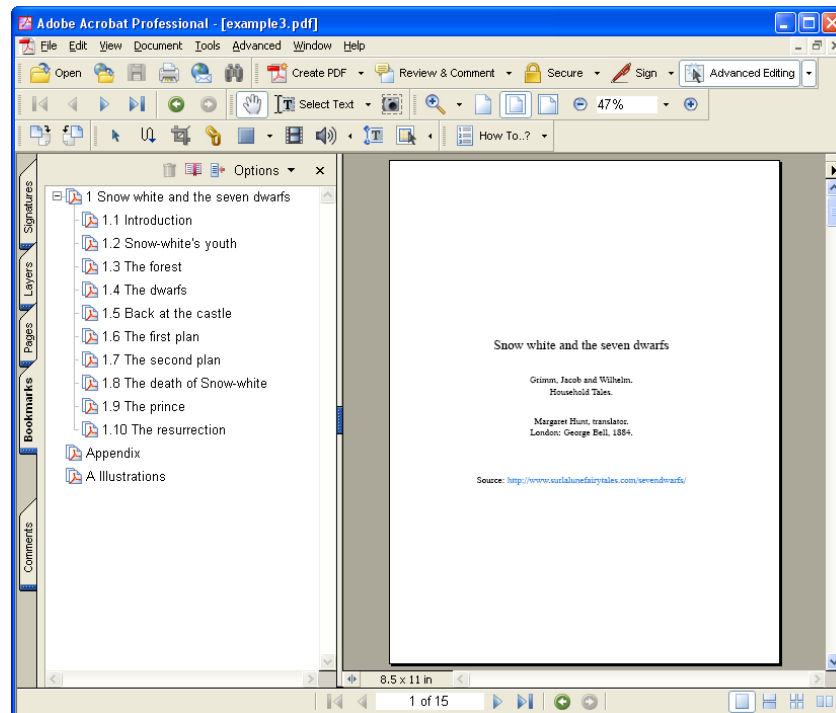
Customizing hyperref

You can customize the appearance of hyperlinks in your document using the command **\hypersetup**:

```
\definecolor{mycolour}{rgb}{0.19,0.54,0.92}
\hypersetup{
  colorlinks=true,
  linkcolor=mycolour, urlcolor=mycolour,
  pdfpagemode=UseOutlines,
  bookmarksopen=true,
  bookmarksnumbered=true,
  plainpages=false, pdfpagelabels,
  pdftitle={Snow white and the seven dwarfs},
  pdfauthor={Grimm, Jacob and Wilhelm.}
}
```

You have to include the package **color** to use **\definecolor**. You can use WinEdt's colour-picker to find RGB values of a selected colour.

hyperref - Example



Distinguishing between \LaTeX and PDF \LaTeX

The command `\hypersetup` also affects the DVI and PostScript file. Usually you want the printed version of your document to be black and not interactive, while the PDF version supports interactivity and has coloured links. The package `ifpdf` defines a command `\ifpdf` that can be used to distinguish between \LaTeX and PDF \LaTeX .

```
\ifpdf
\hypersetup{
  colorlinks=true,
  linkcolor=mycolour,
  urlcolor=mycolour
}
\else
\hypersetup{
  colorlinks=false
}
\fi
```


Creating a master index

To create a master index, just follow these steps:


1. include the package `makeidx`.
2. add the command `\makeindex` before `\begin{document}`
3. add words to the index with the command `\index{word}`. Please note that this command does not display the word. It might be useful to define a command:

```
\newcommand{\idx}[1]{#1\index{#1}}
```

4. put the following commands at the location where you want the index:

```
\newpage\cleardoublepage
\printindex
```

This makes sure that the index will start on an odd page.

5. run \LaTeX twice, then run `makeindex`  and run \LaTeX again.

Special index formats

Use this to point to another word:

```
\index{looking-glass|see{mirror}}
```

Use this to make the page number bold:

```
\index{forest|textbf}
```

Use this to make sub categories:

```
\index{plan!first}
\index{plan!second}
```

Use this to span multiple pages:

```
\index{Snow white|()}
\index{Snow white|})}
```

Citations and a bibliography

Step 1: the bibliography file

1. create a new empty file in WinEdt and save it with the **.bib** extension.
2. add all books, articles etc. to this file using
Insert → BibTeX Items → book/article/report/...
3. fill in all fields (only the capitalized fields are required)

Please note that this is a general file that can be used from any \LaTeX document.

The bibliography file - Example

```
@BOOK{snowwhitedisney,  
  AUTHOR =      {Liza Baker},  
  TITLE =      {Snow White and the Seven Dwarfs},  
  PUBLISHER =   {Disney Press},  
  YEAR =       {1999}  
}  
@ARTICLE{artayres,  
  AUTHOR =      {{Brenda von} Ayres},  
  TITLE =      {The Poisonous Apple in Snow White:  
                Disney's Kingdom of Gender},  
  JOURNAL =     {The Emperor's Old Groove:  
                Decolonizing Disney's Magic Kingdom},  
  YEAR =       {2003},  
  volume =     {11},  
  pages =      {39 - 50}  
}
```

Citations and a bibliography

Step 2: the bibliography style

Use the command `\bibliographystyle{style}` in your \LaTeX document to define the style. This style will determine the appearance of the bibliography. Valid styles are:

unsrt a simple bibliography style that uses numbers as references.

plain the same as **unsrt**, but the bibliography will be sorted.

abbrv the same as **plain**, but the author's first name is abbreviated.

alpha creates a sorted bibliography. References are first 3 letters of the author's last name plus the year (e.g. Ayro3).

natbib creates a sorted bibliography using customizable author-date references. Requires the package **natbib**.

chapterbib allows separate bibliographies for each chapter.

Citations and a bibliography

Step 3: the citations

In your \LaTeX document you can use the `\cite{ref}` command to insert a citation. Only bibliography entries that are being referred to, will be included in the bibliography. You can add entries that are not being referred to using the `\nocite{ref}` command.

If you want all references to be right:

1. \LaTeX ,
2. Bib \LaTeX ,
3. \LaTeX (2 \times).

The command `\bibliography{filename}` will print the bibliography.
Please notice: no extension `.bib` in the filename!

For more details, see `\cite{gevers93,jcw86}`.

```
\bibliographystyle{plain}
\bibliography{referenc}
```

`bibliographystyle` formatting instructions

`bibliography` reference to external file containing the actual references.

Natbib

The package `natbib` offers new bibliography styles and citation commands.

```
\citet[post-note]{key-list}
\citet[pre-note][post-note]{key-list}
```

`\citet` creates a reference for textual citation

`\citep` creates a reference for parenthetical citation:

```
\citet[chapter 2]{snowwhitedisney}
\citep[chapter 2]{snowwhitedisney}
```

Baker (1999, chapter 2)

↔

(Baker 1999, chapter 2)

Bibliography styles defined in `natbib`:

`agsm` popular style with round parentheses.

`chicago` same as `agsm`, but allows ambiguous citations.

(Goossens et al. (1997) can refer to different books with Goossens
as first author among other authors)

Using External Bibliography Managers

- Search engines like [MathSciNet](#) can search for articles and generate BibTeX format.
- Several programs exist to manage citations:
 - J BibtexManager (written in Java)
 - BibTexMng (for Windows)
 - EndNote (Windows, commercial)
 - Reference Manager (Windows, TU/e has a license)

Export from EndNote to BibTeX

EndNote is a popular, but commercial, reference manager with online search tool, which can create bibliographies. It is written for Windows and especially for usage with Word. But it also has a BibTeX export functionality:

- **Edit \implies Output Styles \implies Open Style Manager**
- Check BibTeX Export
- Close Window
- **Edit \implies Output Styles \implies BibTeX Export**

Now you can Copy-Paste, or Export to plain text in BibTeX format.

Export from Reference Manager to BibTeX

Reference Manager can be installed from the [TU/e Library Web Site](#).

- Mark all items that you want to export
- **Bibliography** \implies **Generate From Reference List**
- Output Style: click the Browse button
- Select `BiBTeX.os`
- Save as RTF file, which can be opened in Word and exorted to plain text.

Including programming statements

The package `listings` formats listings. It defines the following commands:

- `\lstlisting{...}` for inline programming statements.
- `\begin{lstlisting} ... \end{lstlisting}` for multi-line listings.
- `\lstinputlisting{filename}` imports a complete source file

Customizing listings

Using the command `\lstset` you can customize the language and appearance of the listing:

```
\lstset{
  language=Java,
  basicstyle=\color{black}\ttfamily,
  commentstyle=\color{green}\itshape\ttfamily,
  keywordstyle=\color{blue}\bfseries\ttfamily,
  showstringspaces=false,
  frame=single, % boxed listings
  backgroundcolor=\color{white}
}
```

Supported languages: too many to mention. Included are Basic, C, C++, Delphi, Fortran, HTML, Java, Mathematica, Matlab, Pascal, Perl, PHP, SAS, SQL, TeX, VBScript, XML.

Customizing listings

Alternatively, you can specify options like this:

```
\definecolor{myyellow}{rgb}{1.00,1.00,0.50}
\begin{lstlisting}[language=Pascal,
                   backgroundcolor=\color{myyellow}]

  readln(N);
  for i := 1 to N do
  begin
    writeln(random)
  end
\end{lstlisting}
```

```
readln(N);
for i := 1 to N do
begin
  writeln(random)
end
```

Create a PDF slide show or poster

The package `pdfscreen` was written for PDF slide show presentations. Unfortunately this package contained some bugs, so another package was written: `tuepdfscreen`. This package can be used to create PDF slide shows. The default appearance is in the TU/e style (colours, fonts) but this can be modified. In fact, any Powerpoint style can be converted to PDF which makes it suitable for TU/ePDFScreen.

Very important: TU/ePDFScreen can only be used with PDF \LaTeX , **not** with \LaTeX !

Detailed information about TU/ePDFScreen can be found in the file

`C:\MiKTeX\localtexmf\examples\presentatie\slides.tex`

This file contains information about all features, like navigation buttons, page numbering, page transitions, step-by-step appearance of lists, including pictures and movies.

This style can also be used for posters. See:

`C:\MiKTeX\localtexmf\examples\presentatie\poster.tex`

Converting your \LaTeX document to a slide show

1. make sure that your document runs with PDF \LaTeX .
2. include the line

`\usepackage{tuepdfscreen}`
3. specify the background image using the `\overlay{file}` command.
4. use the `slide` or `slidetop` environments to divide your text into different slides.
5. if want pages to appear in multiple steps, use the `\pause` command to define breaks.
6. run PDF \LaTeX on the file.
7. if you used the `\pause` command, run the program `AddPause` that can be found in the MiK \TeX Start Menu program group.

Converting your \LaTeX document to a slide show

```
\documentclass{article}

\begin{document}

\section{Mathematics}

\begin{eqnarray*}
\lim_{x \rightarrow 0} \frac{\sin x}{x} &=& 1 \\
\sum_{k=0}^{\infty} x^k &=& \frac{1}{1-x} \quad (|x| < 1)
\end{eqnarray*}

\end{document}
```

Converting your \LaTeX document to a slide show

```
\documentclass{article}
\usepackage{tuepdfscreen}
\overlay{tuebgkuk}
\begin{document}
\begin{slide}
\section{Mathematics}

\begin{eqnarray*}
\lim_{x \rightarrow 0} \frac{\sin x}{x} &=& 1 \\
\sum_{k=0}^{\infty} x^k &=& \frac{1}{1-x} \quad (|x| < 1)
\end{eqnarray*}
\end{slide}
\end{document}
```

Presentations using the Beamer class

The **beamer** class is a class that has become kind of a standard for beamer presentations. It has a huge number of features, including animations, easy creation of handouts, displaying slides in several steps, two displays etc.. It is highly customizable using themes. Beamer has five kind of themes:

1. presentation themes: the overall theme that loads the other themes
2. color themes: define the colors
3. font themes: define the fonts
4. inner themes: define how elements are typeset, like enumerations, block environments, theorems, table of contents
5. outer themes: define the “border” of the slides: head and footlines.

A theme **tue** has been written for **beamer**, so you can create presentations in the TU/e style using the **beamer** class. See the beamer package documentation or the TU/e beamer documentation for more information.

Using TrueType Fonts

The MiKTeX CD-ROM contains a program called TTF2LTX that converts TrueType fonts to Type 1 PostScript fonts that can be used in L^AT_EX.

1. start TTF2LTX and select a font
2. choose a name for this font in L^AT_EX
3. in your L^AT_EX document, use `\fontfamily{name}\selectfont` to use this font.

Warnings:

- not all font shapes (bold, italic) might be available,
- not all font sizes might be available,
- for some fonts the conversion might fail completely!

Using TrueType Fonts - Example

Install the fonts Old English Text MT (oldengl.ttf) and Kunstler Script (kunstler.ttf) in Windows:

go to Control Panel → Fonts.

Then select File → Install new font. Select the TTF files and install them.

Convert these fonts to L^AT_EX fonts, and give them the following names: **oldenglish** and **kunstler**. Renew the font family defaults:

```
\renewcommand{\rmdefault}{kunstler}  
\renewcommand{\sfdefault}{oldenglish}
```

Don't forget to load the package **fontenc** with the **T1** option!

Using TrueType Fonts - Example

