Не много кратко въведение в $\LaTeX\,2_\varepsilon$

или $\LaTeX\,2_\varepsilon$ за 132 минути

Tobias Oetiker

Hubert Partl, Irene Hyna и Elisabeth Schlegl

Версия 4.14, 04 April, 2004

Превод: Ст. Караколева, 12 март 2005 г.

Благодарности

Голяма част от включения в това въведение материал е от австрийското въведение в LaTeX 2.09, написано на немски от:

Hubert Partl <partl@mail.boku.ac.at>
Zentraler Informatikdienst der Universität für Bodenkultur Wien

Irene Hyna <Irene.Hyna@bmwf.ac.at>
Bundesministerium für Wissenschaft und Forschung Wien

Elisabeth Schlegl <noemail>
in Graz

Ако се интересувате от немския вариант на документа, можете да намерите негова версия, обновена за LaTeX 2_{ε} от Jörg Knappen, на адрес `CTAN:/tex-archive/info/lshort/german`.

Докато подготвях този документ, помолих за рецензии в `comp.text.tex`. Получих много отзиви. Следните хора ми помогнаха да подобря този документ с поправки, предложения и материали. Те вложиха много усилия да ми помогнат да доведа документа до настоящия му вид. Аз искам искрено да им благодаря. Естествено, всички грешки, които намерите в тази книга, са мои. Случайно попаднала тук правилно написана дума, навярно дължи своето появяване на реплика от един от долуизброените.

Rosemary Bailey, Marc Bevand, Friedemann Brauer, Jan Busa, Markus Brühwiler, Pietro Braione, David Carlisle, José Carlos Santos, Mike Chapman, Pierre Chardaire, Christopher Chin, Carl Cerecke, Chris McCormack, Wim van Dam, Jan Dittberner, Michael John Downes, Matthias Dreier, David Dureisseix, Elliot, Hans Ehrbar, Daniel Flipo, David Frey, Hans Fugal, Robin Fairbairns, Jörg Fischer, Erik Frisk, Mic Milic Frederickx, Frank, Kasper B. Graversen, Arlo Griffiths, Alexandre Guimond, Andy Goth, Cyril Goutte, Greg Gamble, Neil Hammond, Rasmus Borup Hansen, Joseph Hilferty, Björn Hvittfeldt, Martien Hulsen, Werner Icking, Jakob, Eric Jacoboni, Alan Jeffrey, Byron Jones, David Jones, Johannes-Maria Kaltenbach, Michael Koundouros, Andrzej Kawalec, Alain Kessi, Christian Kern, Jörg Knappen, Kjetil Kjernsmo, Maik Lehradt, Rémi Letot, Johan Lundberg, Alexander Mai, Martin Maechler, Aleksandar S Milosevic, Henrik Mitsch, Claus Malten, Kevin Van Maren, Philipp Nagele, Lenimar Nunes de Andrade, Urs Oswald, Demerson Andre Polli, Maksym Polyakov Hubert Partl, John Refling, Mike Ressler, Brian Ripley, Young U. Ryu, Bernd Rosenlecher, Chris Rowley, Risto Saarelma, Hanspeter Schmid, Craig Schlenter, Baron Schwartz, Christopher Sawtell, Geoffrey Swindale, Laszlo Szathmary, Boris Tobotras, Josef Tkadlec, Scott Veirs, Didier Verna, Fabian Wernli, Carl-Gustav Werner, David Woodhouse, Chris York, Fritz Zaucker, Rick Zaccone, и Mikhail Zotov.

Предговор

LaTeX [1] – е система за текстообработка, ориентирана за произвеждане на научни математически документи с високо типографско качество. Системата е напълно подходяща за произвеждане на други видове документи, от прости писма до напълно завършени книги. LaTeX използва TeX [2] като механизъм за текстообработка.

Това кратко въведение описва LaTeX 2_ε и трябва да бъде достатъчно за повечето приложения на LaTeX. За пълно описание на системата LaTeX четете [1, 3].

Това въведение е разбито на шест глави:

Глава 1 разказва за основната структура на LaTeX-документите. Вие ще научите малко и за историята на LaTeX. След четене на тази глава вие трябва да имате груба представа как работи LaTeX.

Глава 2 задълбочава в детайли текстообработката на вашия документ. Тя обяснява повечето съществени команди и среди в LaTeX. След прочитане на тази глава, ще можете да пишете вашите първи документи.

Глава 3 обяснява как да пишете формули в LaTeX. Многото примери демонстрират как да използвате една от най-силните страни на LaTeX. В края на тази глава има таблици с повечето достъпни в LaTeX математически символи.

Глава 4 разказва за генерацията на азбучния указател и библиографията и включването на EPS графики. Тя въвежда в създаването на PDF документи с pdfLaTeX и представя няколко удобни разширения, като пакета XY-pic.

Глава 5 показва как да се използва LaTeX за създаване на графики. Вместо рисуване на картинката с някаква графична програма, записване във файл и включването му в LaTeX, вие описвате картинката, а LaTeX я рисува вместо вас.

Глава 6 съдържа потенциално опасна информация за това, как да се изменя стандартния макет на документа, произвеждан от LaTeX. Тя ще ви разкаже, как да направите красивото извеждане в LaTeX грозно или очарователно, в зависимост от вашите способности.

Важно е да се четат главите последователно. В края на краищата, книжката не е толкова голяма. Убедете се, че внимателно сте прочели примерите, защото голямо количество от информацията е в примерите, изобилстващи в книгата.

LaTeX е достъпен за повечето компютри, от IBM PC или Mac, до големите UNIX или VMS системи. На много университетски компютърни мрежи, ще откриете, че LaTeX вече е установен и готов за работа. Информация за това как да получите достъп до локалната инсталация на LaTeX, трябва да бъде предоставена в *Local Guide* [5]. Ако имате проблеми с началото на работата, попитайте човека, който ви е дал тази книжка. Целта на този документ е *не* да ви обясни как да инсталирате и настройвате системата LaTeX, а да ви научи как да пишете вашите документи така, че те да могат да бъдат обработени с LaTeX.

Ако искате да притежавате някакъв, отнасящ се до LaTeX материал, потърсете го на един от ftp архивите Comprehensive TeX Archive Network (`CTAN`). Неговият основен адрес е `http://www.ctan.org`. Всички пакети могат също да се получат от ftp-архива `ftp://www.ctan.org` и неговите огледални сайтове по целия свят. Те могат да бъдат намерени например на `ftp://ctan.tug.org` (САЩ), `ftp://ftp.dante.de` (Германия), `ftp://ftp.tex.ac.uk` (Великобритания). Ако вие не сте в нито една от тези страни, намерете най-близкия до вас архив.
По-нататък в книгата постоянно ще намирате препратки на CTAN, особено насочващи към софтуер и документи, които може да искате да изтеглите. Вместо изписване на пълния запис на конкретен адрес, пише просто `CTAN:` и след това адреса в дървото на CTAN, където трябва да отидете.
Ако искате да имате LaTeX на вашия собствен компютър, вижте това, което е достъпно на адрес `CTAN:/tex-archive/systems`.

Ако имате идеи за нещо, което да бъде добавено, премахнато или изменено в този документ, моля, уведомете ме. Особено ме интересува обратната връзка с LaTeX-новаците по отношение на това, кои части от това въведение са лесни за разбиране и кои могат да се обяснят по-добре.

Tobias Oetiker `<oetiker@ee.ethz.ch>`

Department of Information Technology and
Electrical Engineering, Swiss Federal Institute of Technology

Текущата версия на този документ е достъпна на адрес
`CTAN:/tex-archive/info/lshort`

Съдържание

Списък на фигурите

Списък на таблиците

1 Това трябва да се знае

Първата част на тази глава съдържа кратък обзор на философията и историята на LaTeX. Втората част фокусира на основните структури на LaTeX-документите. След четене на тази глава, трябва да имате груба представа за това, как работи LaTeX, което ще ви е нужно, за да разберете останалата част от книгата.

1.1 Имена

1.1.1 TeX

TeX е компютърна програма, създадена от Доналд Кнут (Donald E. Knuth) [2]. Тя е предназначена за обработка на текст и математически формули. Кнут започва писането на TeX през 1977 година за изучаване потенциала на появяващото се по това време цифрово печатащо оборудване, надявайки се всъщност да обърне тенденцията на влошаване на типографското качество, което той виждал например при отпечатването на неговите собствени книги и статии. TeX, в този вид, в който днес се използва, е разработен през 1982 година с някои леки добавки през 1989 (по-добра поддръжка на 8-битните символи и различни езици). TeX е забележителен със своята изключителна стабилност, работа на различни типове компютри и на практика пълна липса на грешки. Номерът на версията на TeX е сходящ към π и в момента е равен на 3.14159.

TeX се произнася «тех», с "ch" като в немската дума "Ach" или в шотландската "Loch." В среда `ASCII` TeX трябва да се пише `TeX`.

1.1.2 LaTeX

LaTeX е макропакет, позволяващ на авторите да обработват и печатат документи с високо типографско качество, с помощта на предварително определени, професионални макети. LaTeX е написан от Leslie Lamport [1]. Като механизъм за обработка той използва TeX. В наши дни LaTeX се поддържа от Frank Mittelbach.

LaTeX се произнася «лейтех» или «латех». Ако се цитира LaTeX в `ASCII` среда, се пише `LaTeX`. LaTeX 2_ε се произнася «лейтех-ту-епс» и се пише `LaTeX2e`.

1.2 Основи

1.2.1 Автор, дизайнер и технически изпълнител

За да публикуват, авторите дават своите ръкописи в издателството. След това един от дизайнерите на издателството определя макета на документа (ширина на текста, шрифтове, интервали под и над заглавие и т.н.). Дизайнерът записва своите инструкции в ръкописа и го дава на техническия изпълнител, който обработва текста в съответствие с тези инструкции.

Човекът-дизайнер се опитва да разбере какво авторът е имал предвид, когато е писал своя ръкопис. Той определя заглавията на главите, цитати, примери, формули и други, използвайки своя професионален опит и съдържанието на ръкописа.

В LaTeX средата, LaTeX играе ролята на дизайнер на книгата, използвайки TeX като технически изпълнител. Но LaTeX е "само" програма, и следователно се нуждае от точни инструкции. Авторът е длъжен да предостави допълнителна информация, описваща логическата структура на неговия документ. Тази информация се записва в текста във вид на LaTeX «команди».

Това коренно се различава от WYSIWYG[1] подхода, приет в голяма част от съвременните текстови процесори, като *MS Word* или *Corel WordPerfect*. В тези приложения авторите определят макета на документа интерактивно, докато въвеждат на текста в компютъра. В процеса на работа те могат да видят на екрана как ще изглежда документа след отпечатване.

При използване на LaTeX обикновено не може да се види изходната картина по време на печатането на текста. Тя обикновено се вижда на екрана след обработка на файла с LaTeX. След това могат да се направят корекции преди изпращане на документа към принтера.

1.2.2 Диэайн на макета

Типографският дизайн е професия. Неопитните автори често допускат сериозни грешки при форматирането, предполагайки, че дизайнът на книгата е въпрос на естетика: „Ако документът изглежда красиво, следователно е с добър дизайн“. Но тъй като документът е предназначен за четене, а не за излагане в художествена галерия, удобството при четене и яснотата са много по-важни, отколкото красивия му вид. Например:

- Размерът на шрифта и номерацията на разделите трябва да се избират така, че да направят структурата на главите и разделите ясна за читателя.

- Редът трябва да бъде достатъчно къс, за да не се напрягат очите на читателя и достатъчно дълъг за красиво запълване на страницата.

С WYSIWYG системите, авторите често създават естетически приятни документи, със слабо изразена или непоследователна структура. LaTeX предотвратява такива

[1] What you see is what you get.

грешки при форматирането, заставяйки автора да обявява *логическата* структура на документа. След това вече LaTeX избира най-подходящия макет (стил) на документа.

1.2.3 Преимущества и недостатъци

Темата, която често се обсъжда, когато хора от света на WYSIWYG се срещнат с потребителите на LaTeX е «преимуществата на LaTeX пред нормалните текстови процесори» или обратното. Най-доброто, което можете да направите, когато започне такава дискусия, е да я избегнете, тъй като тя често излиза извън контрол. Но не винаги може да се отклони спора...

Ето ви оръжие. Основните преимущества на LaTeX пред обикновените текстови процесори:

- Достъпни са професионално изработени макети, които правят документите да изглеждат наистина «като издание».
- Записването на формули се прави по удобен начин
- Потребителите трябва да научат само няколко лесно разбираеми команди, които определят логическата структура на документа. Почти никога не се налага те да се грижат за макета на документа.
- Лесно се изработват дори сложни структури, като съдържание, забележки, литература и други.
- За решение на много типографски задачи, които не се поддържат директно от основния LaTeX, има свободно разпространявани допълнителни пакети. Например, съществуват пакети за включване на PostScript-графики или за оформяне на библиография в точно съответствие с конкретни стандарти. Много от тези допълнителни пакети са описани в *The LaTeX Companion* [3].
- LaTeX поощрява авторите да пишат добре структурирани документи, тъй като именно така работи LaTeX — чрез дефиниране на структури.
- TeX, форматиращото сърце на LaTeX 2_ε, е изключително мобилен и свободно достъпен. Затова системата работи фактически на всички съществуващи платформи.

LaTeX има също и някои недостатъци, но сякаш ми е трудно да намеря сред тях съществени. Но съм сигурен, че други ще намерят стотици `;-)`

- LaTeX не работи добре за хора, които са продали душите си ...
- Тъй като предопределените макети имат множество параметри за настройване, изработването на напълно нов макет на документа не е много просто и отнема много време.[1]

[1]Казват, че това е една от основните цели на бъдещия проект LaTeX3.

- Много е сложно да се пишат неструктурирани и неорганизирани документи.
- Вашата морска свиня може изобщо да не успее да разбере концепцията на логическо разбиване, въпреки първите успехи.

1.3 Изходни файлове на LaTeX

Входен файл за LaTeX е обикновен текстов `ASCII` файл. Той може да бъде създаден с какъв да е текстов редактор. Той съдържа текста на документа, заедно с командите, които указват на LaTeX как да обработва текста.

1.3.1 Интервали

«Празните» символи, такива като интервал и табулиране, се интерпретират от LaTeX еднакво – като «интервал». *Няколко последователни* празни символи се възприемат като *един* «интервал». Празни символи в началото на реда обикновено се игнорират, а единично прекъсване на реда се възприема като «интервал».

Празен ред между два реда в текста определя края на параграфа. *Няколко* празни реда се възприемат също като *един* празен ред. Долу е приведен пример. Вляво е текстът от входния файл, вдясно — форматирания изход.

```
Не е важно дали оставяте един
    или   няколко интервала
между думите.

Празен ред поставя начало
на нов параграф.
```

Не е важно дали оставяте един или няколко интервала между думите.

Празен ред поставя начало на нов параграф.

1.3.2 Специални символи

Следващите символи са резервирани символи, които или имат в LaTeX специален смисъл, или не са достъпни във всички шрифтове. Ако ги въведете в текста направо, те обикновено не се печатат, а заставят LaTeX да направи нещо, което вие изобщо не предполагате.

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

Както по-късно ще видите, тези символи могат да се използват във вашите документи, като се добави пред тях префикс «\»:

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

$ % ^ & _ { } ~

Тези символи, както и много, много други, могат да бъдат отпечатани със специални команди в математически формули или като акценти. Знакът «\» *не може* да се въвежда два пъти, тъй като тази команда (\\) се използва за прекъсване на реда.[1]

1.3.3 Команди на LaTeX

Командите на LaTeX са чувствителни към регистъра и приемат една от следните две форми:

- Те започват със символа backslash «\» и продължават с името, състоящо се само от букви. Имената на командите завършват с интервал, цифра, или всяка друга «не-буква».
- Те се състоят от «\» и точно един не-буквен символ.

LaTeX игнорира интервалите след командата. Ако искате да получите интервал след команда, трябва да поставите или «{}» и интервал, или специална команда за интервал след името на командата. «{}» не позволява на LaTeX да игнорира всички интервали след името на командата.

```
Чел съм, че Кнут е разделял
хората, работещи с \TeX{} на
\TeX{}ници и \TeX перти.\\
Днес е \today
```

Чел съм, че Кнут е разделял хората, работещи с TeX на TeXници и TeXперти.
Днес е 12 март 2005 г.

Някои команди се нуждаят от параметър, който трябва да бъде зададен между фигурни скоби «{ }» след името на командата. Някои команди поддържат незадължителни параметри, които се добавят след името на командата в квадратни скоби «[ ]». Следният пример използва някои команди на LaTeX. Не се замисляйте над тях, те ще бъдат разяснени по-късно.

```
Можете да \textsl{разчитате}
на мен!
```

Можете да *разчитате* на мен!

```
Моля, започнете нов ред
точно тук!\newline
Благодаря!
```

Моля, започнете нов ред точно тук!
Благодаря!

1.3.4 Коментари

Когато в процеса на обработка на входния файл LaTeX срещне символа %, той игнорира остатъка от текущия ред, преминаването на нов ред и всички интервали в началото на следващия ред.

[1] Вместо това използвайте командата `$\backslash$`. Тя дава ‘\’.

Това може да се използва за добавяне в изходния файл на бележки, които няма да се извеждат за печат.

```
Това е %глупав,%
%но по-скоро поучителен%
пример: Spercal%
         ifragilist%
     icexpialidocious
```

Това е пример: Spercalifragilisticexpialidocious

Знакът `%` може също да се използва за разбиване на даден ред на части там, където не се допускат интервали или нов ред.

За по-дълги коментари може също да се използва средата `comment`, предоставена от пакета `verbatim`. Това означава, че за използване на средата `comment`, трябва в преамбюла на документа да се добави командата `\usepackage{verbatim}`:

```
Това е още един
\begin{comment}
достатъчно глупав,
но полезен
\end{comment}
пример за поставяне
на коментари в документа.
```

Това е още един пример за поставяне на коментари в документа.

Забележете, че това няма да работи в сложни среди, например математика.

1.4 Структура на входния файл

Когато LaTeX 2_ε обработва входния файл, той очаква от него да следва определена структура. Така всеки входен файл трябва да започва с командата

```
\documentclass{...}
```

Тя указва какъв тип документ започвате да пишете. След това вие можете да включите команди, които влияят на стила на целия документ, или да зареждате пакети, добавящи нови възможности в системата LaTeX. За зареждане на такъв пакет се използва командата

```
\usepackage{...}
```

Когато цялата настройка е завършена,[1] започва тялото на документа след командата

```
\begin{document}
```

След тази команда се въвежда текста, заедно с някои полезни команди на LaTeX. В края на документа се добавя командата

[1] Областта между `\documentclass` и `\begin{document}` се нарича *преамбюл*.

```
\end{document}
```

Всичко, което следва след нея, се игнорира от LaTeX.

Фигура 1.1 показва съдържанието на минималния LaTeX 2_ε-файл. Малко по-сложен входен файл е даден на Фигура 1.2.[1]

```
\documentclass{article}
\usepackage{T2A]{fontenc}
\usepackage{cp1251]{inputenc}
\usepackage[bulgarian]{babel}
\begin{document}
Кратко и ясно.
\end{document}
```

Фигура 1.1: Минимален LaTeX-файл

1.5 Типична сесия при работа с LaTeX

Сигурен съм, че вече нямате търпение да изпробвате малкия пример на LaTeX-файла, приведен на страница 7. Нека да опитаме. Сам по себе си LaTeX не включва графичен интерфейс. Той е просто програма, обработваща указания файл. Някои дистрибутиви на LaTeX включват графична обвивка, където можете да изберете с мишката компилация на указания файл. В други системи трябва да набирате командите в команден режим. Следващото описание предполага, че на вашия компютър вече има работеща LaTeX-инсталация[2].

1. Редактирайте или създайте входния LaTeX-файл. Този файл трябва да бъде плосък ASCII текст. В Unix всеки редактор създава именно това. В Windows трябва да се убедите, че съхранявате файла във формат ASCII, или *Plain text*. Избирайки име на файла, добавете разширение `.tex`.

2. Компилирайте вашия входен файл с LaTeX. Ако всичко мине без грешки, ще получите файл `.dvi`. За да получите коректно съдържание и вътрешни препратки, може да се наложи да стартирате LaTeX няколко пъти. Ако във входния файл има грешка, LaTeX ще ви я покаже и ще спре обработката на входния файл. За да се върнете в командния ред, натиснете `ctrl-D`.

   ```
   latex foo.tex
   ```

[1] В дадения пример е добавен минимален набор команди, необходими за поддръжка на български език. Тук се използва пакета babel, който е стандартно средство за езикова локализация на TeX. Ако е необходимо, се консултирайте с вашия *Local Guide* [5] или с администратора. — *Бел. ред.*

[2] Това обикновено е така във всяка съвременна Unix-система.

```
\documentclass[a4paper,11pt]{article}
\usepackage{T2A}{fontenc}
\usepackage{cp1251]{inputenc}
\usepackage[bulgarian]{babel}
\begin{document}
% дефиниране на заглавието
\author{H.~Partl}
\title{Minimalizm}
\frenchspacing
\begin{document}
%генериране на заглавието
\maketitle
%поставяне на съдържание
\tableofcontents
\section{Начало}
Ето тук започва моята забележителна статия.
\section{Край}
\ldots{} и тук тя завършва.
\end{document}
```

Фигура 1.2: Пример на реалистична журнална статия

3. Сега можете да видите файла `.dvi`. Това може да се направи по различни начини. Може да се види на екрана с командата

   ```
   xdvi foo.dvi &
   ```

   Тя работи само в среда Unix и X11. Ако работите под Windows, опитайте `Yap` („yet another previewer“).

   Можете също да преобразувате DVI-файла в PostScript за печат или преглеждане с помощта на Ghostscript:

   ```
   dvips -Pcmz foo.dvi -o foo.ps
   ```

   Ако сте късметлия, вашата LaTeX система дори ще съдържа инструмента `dvipdf`, който позволява да преобразувате вашите `.dvi`-файлове направо в PDF.

   ```
   dvipdf foo.dvi
   ```

1.6 Макет на документа

1.6.1 Класове документи

Първото, което LaTeX трябва да знае при обработката на входния файл, е типа на създавания от автора документ. Той се задава с командата `\documentclass`.

```
\documentclass[опции]{клас}
```

Тук *клас* определя типа на създавания документ. Таблица 1.1 изброява класовете документи, разглеждани в това въведение. Дистрибуцията на LaTeX 2_ε осигурява допълнителни класове за други документи, включително писма и слайдове. Параметърът *опции* изменя поведението на класа документ. Опциите трябва да се разде́лят със запетая. В Таблица 1.2 са изброени най-употребяваните опции на стандартните класове документи.

Пример: Входният файл на LaTeX-документа може да започва с реда

```
\documentclass[11pt,twoside,a4paper]{article}
```

Той заставя LaTeX да набира документа като *статия*, с основен размер на шрифта *единадесет точки* и да форматира документа за *двустранен* печат на хартия с *формат A4*.

Таблица 1.1: Класове документи

`article` За статии и научни журнали, презентации, кратки отчети, програмна документация, покани...

`report` За по-дълги отчети, съдържащи няколко глави, неголеми книжки, дисертации...

`book` За книги

`slides` За слайдове. Използват се големи букви без пренасяне. Вместо това може да се използва FoilTEX.[a]

[a] `CTAN:/tex-archive/macros/latex/contrib/supported/foiltex`

1.6.2 Пакети

Докато пишете вашия документ, вероятно ще забележите, че има някои области, в които базовия LaTeX не може да реши вашите проблеми. Ако искате да включите в документа графика, цветен текст или изходен код на програма от външен файл, вие трябва да разширите възможностите на LaTeX. Такива разширения се наричат пакети. Пакетите се активират с командата

`\usepackage[`*опции*`]{`*пакет*`}`

където *пакет* е името на пакета, а *опции* – списък от ключови думи, включващи специални възможности на пакета. Някои пакети са включени в основния дистрибутив на LaTeX 2_ε (виж Таблица 1.3). Други се предоставят отделно. Допълнителна информация за установените пакети може да намерите във вашия *Local Guide* [5]. Основен източник на информация за LaTeX-пакетите е *The LaTeX Companion* [3]. Тя съдържа описание на стотици пакети, както и описание на това, как да пишете ваши собствени разширения за LaTeX 2_ε.

1.6.3 Стилове на страницата

LaTeX поддържа три предопределени комбинации на горния и долния колонтитул на страницата, така наречените стилове на страницата. Параметърът *стил* на командата

`\pagestyle{`*стил*`}`

определя кой от тях да се използва. Предопределените стилове на страницата са изброени в Таблица 1.4.

Таблица 1.2: Опции на класовете документи

`10pt`, `11pt`, `12pt` Установява размера на основния шрифт на документа. Ако нито една от тези опции не е зададена, се подразбира `10pt`.

`a4paper`, `letterpaper`... Определя размера на страницата. По подразбиране е `letterpaper`. Също могат да бъдат указани `a5paper`, `b5paper`, `executivepaper` и `legalpaper`.

`fleqn` Формулите на отделен ред да се отпечатват подравнени вляво, а не центрирани.

`leqno` Поставя номерата на формулите вляво, а не вдясно.

`titlepage`, `notitlepage` Определя дали трябва да започне нова страница след заглавието на документа или не. По подразбиране класът `article` не започва нова страница, а `report` и `book` — започват.

`onecolumn`, `twocolumn` Инструктира LaTeX да отпечатва документа в една колона или в две колони.

`twoside, oneside` Избира едно- или двустранен изход. По подразбиране класът `article` и `report` използват едностранен изход, класът `book` — двустранен изход. Забележете, че тези опции се отнасят само до типа на документа. Опцията `twoside` *не* заставя вашия принтер да печата двустранно.

`landscape` Сменя положението на страницата на ландшафтно.

`openright, openany` Установява главите да започват или само на дясната страница, или на първата достъпна. Това не работи с класа `article`, тъй като той не знае нищо за главите. Класът `report` по подразбиране започва глава на следващата страница, а класът `book` — на дясната.

Таблица 1.3: Някои от разпространяваните с LaTeX пакети

---

doc Позволява да се документират програми на LaTeX.
Описан в `doc.dtx`[a] и в *The LaTeX Companion* [3].

exscale Предоставя мащабирани версии на разширени математически шрифтове.
Описан в `ltexscale.dtx`.

fontenc Определя каква кодировка на шрифта трябва да използва LaTeX.
Описан в `ltoutenc.dtx`.

ifthen Предоставя команди от вида
“ако …, то изпълни …, иначе изпълни …“.
Описан в `ifthen.dtx` и в *The LaTeX Companion* [3].

latexsym За включване на шрифт със специални символи на LaTeX, трябва да се използва пакета `latexsym`. Описан в `latexsym.dtx` и в *The LaTeX Companion* [3].

makeidx Предоставя команди за генерация на указател. Описан в Раздел 4.3 и в *The LaTeX Companion* [3].

syntonly Обработва документа, без да го отпечатва.

inputenc Позволява да се указва входната кодировка, такава като ASCII, ISO Latin-1, ISO Latin-2, 437/850 IBM code pages, Apple Macintosh, Next, ANSI-Windows или определена от потребителя. Описан в `inputenc.dtx`.

---

[a] Този файл трябва да бъде инсталиран във вашата система и вие можете да получите `dvi` файл, като дадете командата `latex doc.dtx` във всеки каталог, където имате права за запис. Същото се отнася за всички подобни файлове, споменати в тази таблица.

Таблица 1.4: Предопределени стилове на LaTeX-страницата

`plain` Печата номера на страницата долу, в средата на долния колонтитул. Този стил е по подразбиране.

`headings` Печата името на текущите глава или раздел и номера на текущата страница в горния колонтитул, докато долният колонтитул остава празен. (Този стил е използван в този документ.)

`empty` Установява и долния, и горния колонтитул на страницата празни.

Възможно е да се смени стила на текущата страница с командата

`\thispagestyle{`*стил*`}`

Описание на това, как да създавате ваши собствени стилове на колонтитулите има в *The LaTeX Companion* [3], а също и в Раздел 4.4 на страница 71.

1.7 Файлове, които можете да срещнете

Работейки с LaTeX, скоро ще се окажете в лабиринт от файлове с различни разширения и вероятно – никакво указание. Следният списък обяснява различните типове файлове, използвани при работа с TeX. Моля да имате предвид, че тази таблица не е пълен списък с разширения, но ако намерите неспоменато разширение, което считате за важно, моля уведомете ме.

`.tex` Входен файл на LaTeX или TeX. Може да бъде компилиран с командата `latex`.

`.sty` Макро-пакет на LaTeX. Този файл можете да заредите във вашия LaTeX-документ с с използване на командата `\usepackage`.

`.dtx` Документиран TeX. Това е основен формат на разпространяваните стилови файлове на LaTeX. Ако обработите файла `.dtx`, ще получите документиран макро-код на LaTeX-пакета, съдържащ се в `.dtx` файла.

`.ins` Инсталатор за файловете, съдържащи се в съответния файл `.dtx`. Ако вие изтеглите пакета от Интернет, обикновено получавате файлове `.dtx` и `.ins`. Компилирайте с LaTeX файла `.ins`, за да разопаковате файла `.dtx`.

`.cls` Файл за класа, определящ вида на вашия документ. Избира се с командата `\documentclass`.

`.fd` Файл с описание на шрифт, описващ на LaTeX нов шрифт.

Следващите файлове се генерират, когато LaTeX обработва входния файл:

`.dvi` Device Independent file (файл, независещ от устройството). Това е основен резултат от компилирането с LaTeX. Съдържанието му може да се види с програмата за визуализиране DVI, или като изпрати към принтера с `dvips` или подобно приложение.

`.log` Дава детайлен отчет за това, какво се е случило при последната компилация.

`.toc` Съхранява заглавията на всички раздели. Чете се при следващата компилация и се използва при генерацията на съдържанието.

`.lof` Подобен на `.toc` списък от фигури.

`.lot` И отново същото за списък от таблици.

`.aux` Още един файл, предаващ информация от една компилация към следващата. Освен всичко друго, `.aux` файла се използва за съхраняване на информация, свързана с препратките.

`.idx` Ако вашият документ има азбучен указател, LaTeX съхранява всички думи за указателя в този файл. Обработете този файл с програмата `makeindex`. Подробно вижте в Раздел 4.3 на страница 69.

`.ind` Обработен файл `.idx`, готов за включване във вашия документ при следващата компилация.

`.ilg` Журнал за работата на `makeindex`.

1.8 Големи проекти

При работа с големи документи, може да искате да разделите входния файл на части. LaTeX има две команди, които ще ви помогнат да направите това.

`\include{`*filename*`}`

Тази команда може да се използва в тялото на документа, за да се включи в него съдържанието на файл *filename.tex*. Забележете, че LaTeX започва нова страница, преди да обработва материала, съдържащ се във *filename.tex*.

Втората команда може да се използва само в преамбюла. Тя указва на LaTeX да чете само някои от включените (с `\include`) файлове.

`\includeonly{`*filename1*`,`*filename2*`,`...`}`

След задаване в преамбюла на документа, тази команда ще изпълнява само тези команди `\include`, чиито имена на файлове са изброени в аргумента на командата

`\includeonly`. Да отбележим, че между имената на файловете и запетаите не трябва да има празно място.

Командата `\include` започва обработка на включения текст на нова страница. Това е удобно при използване на `\includeonly`, тъй като границите на страниците няма да се преместват, дори ако някои от включените файлове се пропускат. Когато това е нежелателно, се използва командата

`\input{`*filename*`}`

Тя просто включва съдържанието на указания файл.

За бърза проверка на документа използвайте пакета syntonly. Той заставя LaTeX да пробяга по документа, проверявайки синтаксиса и използваните команди, без да генерира изходен файл DVI. LaTeX работи по-бързо в този режим, икономисвайки ценното ви време. Синтаксисът е много прост:

```
\usepackage{syntonly}
\syntaxonly
```

Когато пожелаете да получите изходен файл, просто коментирайте втория ред, като добавите знак процент.

2 Обработка на текста

След прочитане на предходната глава, вие трябва да знаете от какво се състои LaTeX-документа. Във втората глава ще запълня тази структура с всичко, от което се нуждаете за изработването на истински материали.

2.1 Структура на текста и езика

Автор: Hanspeter Schmid <hanspi@schmid-werren.ch>

Основната цел на написването на текст е да се предадат на читателя идеи, информация или знания. Читателят по-добре ще разбере текста, ако тези идеи са структурирани, и най-добре той ще види и почувства тези структури, ако типографската форма отразява логическата и смислова структура на съдържанието.

LaTeX се различава от другите системи за обработка по това, че вие трябва само да му задавате логическата и смислова структура на текста. Той след това избира типографската форма на текста в съответствие с «правила», зададени във файла на класа на документа и в различни стилови файлове.

Най-важният елемент на текста в LaTeX (и в набора въобще е параграфа). Той се нарича «елемент на текста», защото параграфът е тази типографска форма, в която се формулира една свързана мисъл или идея. В следващите раздели вие ще разберете как можете да отивате на нов ред, например, с помощта на \\ и как да определите началото на параграф, например като оставите празни редове. Затова, ако започвате нова мисъл, трябва да започва нов параграф, а ако не – използвайте преминаване на нов ред. Ако се съмнявате във вашето разбиване на параграфи, помислете за текста като носител на идеи и мисли. Ако сте започнали нов параграф, продължавайки изложението на същата мисъл, то махнете началото на новия параграф. Ако в параграфа започва съвършено нова мисъл, разделете го на два.

Много хора напълно подценяват важността на разумното разбиване на текста на параграфи. Много от тях не се замислят за значението на прекъсването на параграфа или, особенно в LaTeX, започват параграфи, не знаейки това. Последната грешка се прави лесно, когато в текст се включват уравнения. Вижте следния пример и ще разберете, защо понякога в него се използват празни редове (прекъсване на параграфа), а понякога – не (ако вие още не разбирате всички команди достатъчно, за да разберете примерите, прочетете моля тази и следващите глави и след това се върнете към примерите отново.)

```
\ldots когато Айнщайн
е въвел своята формула
\begin{equation}
  e = m \cdot c^2 \; ,
\end{equation}
която е най-широко
известна и най-малко
понятна физическа формула.
```

…когато Айнщайн е въвел своята формула

$$e = m \cdot c^2 \; , \tag{2.1}$$

която е най-широко известна и най-малко понятна физическа формула.

```
\ldots откъдето следва
закона за тока на Кирхоф:
\begin{equation}
  \sum_{k=1}^{n} I_k = 0 \; .
\end{equation}

Закон за напрежението на Кирхоф
може да бъде въведен \ldots
```

…откъдето следва закона за тока на Кирхоф:

$$\sum_{k=1}^{n} I_k = 0 \; . \tag{2.2}$$

Закон за напрежението на Кирхоф може да бъде въведен …

```
\ldots който има
няколко преимущества.

\begin{equation}
  I_D = I_F - I_R
\end{equation}
е ядро на съвършенно
друг модел транзистор.
\ldots
```

…който има няколко преимущества.

$$I_D = I_F - I_R \tag{2.3}$$

е ядро на съвършенно друг модел транзистор. …

Следващата, по-малка единица на текста е изречението. В английския текст след точка, завършваща изречение, се слага по-голям интервал, отколкото след точка, стояща след съкращение. LaTeX се старае да определи, коя от тях сте имали предвид. Ако той греши, вие трябва да му подска́жете. Това е обяснено по-нататък в тази глава.

Структурата на текста обхваща дори отделни части на изречението. Голяма част от езиците използват много сложни правила на пунктуация, но в много езици (вкючително немски и английски) поставете почти всички запетаи на определени места, просто като помните, че те означават: кратко спиране в потока на изложение. Ако не сте сигурен в това, къде да поставите запетаи, прочетете изречението на глас, като се спирате на всяка запетая. Ако някъде това ви прозвучи странно, махне́те запетаята, ако почувствате необходимост да спрете на друго място, поставете запетая.[1]

[1] Всичко това е малко приложимо за български език, но може да ви помогне, ако пишете на английски. — *Бел. ред.*

Накрая, параграфите в текста трябва да бъдат също логически структурирани на по-високо ниво, обединявайки се в глави, раздели, и така нататък. Тъй като типографскияt ефект на записа, например, `\section{Структура на текста и езика}` обикновено е достатъчно очевиден, веднага става ясно, как се използват тези структури от високо ниво.

2.2 Рзбиване на редове и страници

2.2.1 Подравнени параграфи

Книгите често се оформят така, че всички редове да имат еднаква дължина. LaTeX поставя необходимите прекъсвания на редовете и интервали между думите, оптимизирайки форматирането на параграфа като цяло. Ако е необходимо, той освен това пренася думи, които не се събират на реда. От класа на документа зависи това, как ще се форматира параграфа. Обикновено първият ред на всеки параграф започва с отстъп, а допълнително разстояние между два параграфа няма. Подробности вижте в Раздел 6.3.2.

В някои случаи може да е необходимо да се укаже на LaTeX да прекъсне реда. След командите

`\\` или `\newline`

започва нов ред, без да започва нов параграф. Командата

`\\*`

забранява освен това прекъсването на страницата след поставеното прекъсване на реда. Командата

`\newpage`

започва нова страница.

Всяка от следните команди съответно

`\linebreak[`*n*`]`, `\nolinebreak[`*n*`]`, `\pagebreak[`*n*`]` и `\nopagebreak[`*n*`]`

прекъсва реда, забранява прекъсването на реда, прекъсва страницата и забранява прекъсването на страницата. Те дават възможност на автора да влияе на тяхното действие, посредством незадължителния аргумент n, който може да бъде равен на число от нула до четири. Ако дадете за n число, по-малко от 4, вие оставяте на LaTeX правото на избор, да игнорира вашите команди, ако резултатът бъде много лош. Не бъркайте командите за “прекъсване“ (break) с командите за “начало“ (new). Дори ако давате команда «прекъсване», LaTeX ще се опита да подравни дясната граница на страницата и общата височина на страницата, както е описано в следващия раздел. Ако вие действително искате да започнете «нов ред», използвайте съответните команди.

LaTeX винаги се опитва да направи най-доброто от възможните разбивания на реда. Ако не може да намери начин да разбие реда в съответствие със своите стандарти, той позволява на един ред да отстъпи от параграфа надясно. След това LaTeX извежда диагностика («overfull hbox») по време на обработката на входния файл. Най-често това се случва, когато LaTeX не може да намери подходящо място за пренасяне на думата.[1] Можете да инструктирате LaTeX да отслаби малко своите стандарти ако дадете команда `\sloppy`. Това ще предотврати такива прекалено дълги редове, чрез увеличаване на интервалите между думите, дори ако крайният резултат не е оптимален. В този случай потребителят получава предупреждение («underfull hbox»). В голяма част от случаите резултатът не изглежда много добре. Командата `\fussy` връща LaTeX към неговото поведение по подразбиране.

2.2.2 Пренасяне

LaTeX пренася думите, когато това е необходимо. Ако алгоритъмът за пренасяне не може да намери точните места за пренасяне, вие можете да се справите със ситуацията, казвайки на TeX за изключенията, с помощта на следните команди.

Командата

`\hyphenation{`*списък от думи*`}`

предизвиква пренасяне на думите, изброени в нейния аргумент, само на местата, отбелязани с «-». Аргументът на командата трябва да съдържа само думи, състоящи се от обикновени букви, или по-скоро знаци, които се разглеждат от LaTeX като нормални букви. Правилата на пренасяне се запомнят за езика, активен в момента на обръщане към командата `\hyphenation`. Това означава, че поставите командата за пренасяне в преамбюла на документа, тя ще влияе на английските преноси. Ако вие я поставите след командата `\begin{document}` и използвате при това някакъв пакет за поддръжка на национален език, например babel, то правилата за пренасяне ще бъдат активни за езика, активиран с babel.

В следващия пример се разрешава пренасяне в думата «hyphenation», както и в думата «Hyphenation», и се забранява пренасянето в думите «FORTRAN», «Fortran» и «fortran». В аргумента не се допускат специални символи.

Пример:

```
\hyphenation{FORTRAN Hy-phen-a-tion}
```

Командата `\-` поставя в думата изборен пренос. Тя също така установява единствените разрешени точки за пренасяне в тази дума. Тази команда е особено полезна за

[1] Макар, че LaTeX ви дава предупреждение и номера на реда, такива редове не винаги е лесно да се намерят. Ако вие използвате в командата `\documentclass` опция `draft`, такива редове ще бъдат отбелязани с дебели черни линии в десните полета.

думи, съдържащи специали символи (например, символи с акценти), защото LaTeX не пренася такива думи автоматично.

```
I think this is: su\-per\-cal\-%
i\-frag\-i\-lis\-tic\-ex\-pi\-%
al\-i\-do\-cious
```

I think this is: supercalifragilisticexpialidocious

Няколко думи могат да се държат заедно на един ред с командата

`\mbox{`*текст*`}`

Тя заставя LaTeX да държи аргумента цял, при всякакви обстоятелства.

```
Номерът на моя телефон скоро
ще бъде сменен. Той ще бъде
\mbox{0116 291 2319}.

Параметърът
\mbox{\emph{име на файл}} трябва
да съдържа името на файла.
```

Номерът на моя телефон скоро ще бъде сменен. Той ще бъде 0116 291 2319.

Параметърът *име на файл* трябва да съдържа името на файла.

Командата `\fbox` е аналогична на командата `\mbox`, но в добавка, около нейния аргумент се рисува видима рамка.

2.3 Специални редове

В някои от примерите на предните страници, вече сте виждали някои много прости команди на LaTeX за отпечатване на специални текстови стрингове:

Команда	Пример	Описание
`\today`	12 март 2005 г.	Текуща дата на текущия език
`\TeX`	TeX	Име на любимата ви издателска система
`\LaTeX`	LaTeX	Името на LaTeX
`\LaTeXe`	LaTeX 2_ε	Текуща инкарнация на LaTeX

2.4 Специални букви и символи

2.4.1 Знаци кавички

За набора на кавички *не трябва* да използвате знака `"`, както на пишеща машина. При обработка на текста съществуват специални знаци отварящи и затварящи кавички. В LaTeX използвайте два знака `‘` за отварящи кавички и два знака `’` - за затварящи. За единични кавички, просто използвайте един от тях.[1]

[1] При обработка обикновено се изискват две двойки кавички, при случай на «„кавички“ в кавички». При обработка на български език обикновено вместо “такива” и ‘такива’ кавички се използват «тези» и „тези“. Те се задават обикновено с командите `\flqq`, `\frqq` (или `«` и `»`) и `\glqq`, `\grqq`. — *Бел. ред.*

```
``Моля, натиснете клавиш `x' ''
```

“Моля, натиснете клавиш ‘x’ ”

Да, знам, че резултатът не е идеален. Това е всъщност акцент за отваряща кавичка и вертикална кавичка за затваряща, независимо от това, какъв шрифт ще изберете.

2.4.2 Тире и дефис

LaTeX знае за четири вида тире. Три от тях можете да получите с различен брой последователни знаци `-`. Четвъртото всъщност не е тире изобщо, а математическия знак минус:

```
dauther-in-low, X-лъчи\\
страници 13--67\\
да~--- или не?\\
$0$, $1$ и $-1$
```

dauther-in-low, X-лъчи
страници 13–67
да — или не?
0, 1 и −1

Имената на тези тирета са: `-` дефис, `--` късо тире, `---` дълго тире и `$-$` знак минус.

2.4.3 Тилда (∼)

Често срещан символ в WWW-адреси е символът тилда. За нейната генерация в LaTeX може да се използва `\~`, но резултатът (˜) няма да бъде това, което желаете. В замяна опитайте това:

```
http://www.rich.edu/\~{}bush \\
http://www.clever.edu/$\sim$demo
```

http://www.rich.edu/˜bush
http://www.clever.edu/∼demo

2.4.4 Знак за градус (◦)

Следният пример показва как да се печата знак за градус в LaTeX:

```
Температурата вече е
$-30\,^{\circ}\mathrm{C}$.
Скоро ще започне свръхпроводимост.
```

Температурата вече е $-30\,^{\circ}\mathrm{C}$. Скоро ще започне свръхпроводимост.

Пакетът textcomp позволява достъп до символа градус с командата `\textcelsius`.

2.4.5 Символ Евро (€)

Когато пишете за пари в наши дни, се нуждаете от символа на евровалутата. Много съвременни шрифтове съдържат символа Евро. След зареждане на пакета textcomp в преамбюла на документа

```
\usepackage{textcomp}
```

можете да използвате командата

```
\texteuro
```

за достъп до този символ.

Ако вашият шрифт няма собствен Евро символ, или на вас не ви харесва как той изглежда, имате още два изхода.

Първият е пакета eurosym. Той осигурява официалния символ Евро:

```
\usepackage[official]{eurosym}
```

Ако предпочитате символ Евро, който съответства на вашия шрифт, то вместо опцията `official` използвайте опция `gen`.

Ако на вашата система е инсталиран Adobe Eurofonts (той е свободно достъпен от `ftp://ftp.adobe.com/pub/adobe/type/win/all`), вие можете да използвате или пакета europs и командата `\EUR` (за Евро символ, съответстващ на текущия шрифт), или пакета eurosans и командата `\euro` (за официалния символ Евро).

Пакетът marvosym осигурява множество различни символи, в това число и Евро, под името `\EUR`. Неудобно е това, че не се предоставят наклонени и дебели варианти на символа Евро.

Таблица 2.1: Пълен набор на Евро символи

пакет	команда	roman	sans-serif	typewriter
eurosym	`\euro`	€	€	€
[gen]eurosym	`\euro`	€	€	€
europs	`\EUR`	€	€	€
eurosans	`\euro`	€	€	€
marvosym	`\EUR`	€	€	€

2.4.6 Многоточие (...)

На пишеща машина, точка или запетая заемат същото пространство, както и всяка друга буква. Напечатани в книга, тези символи заемат малко място и се разполагат много близо до предходната буква. Следователно, вие не можете да въведете многоточие просто като напечатате три точки, тъй като разстоянието между тях ще бъде неправилно. Вместо това, за многоточие има специална команда, наричана

```
\ldots
```

```
не така ... а така:\\
Ню Йорк, Токио, Будапеща, \ldots
```

не така ... а така:
Ню Йорк, Токио, Будапеща, …

2.4.7 Лигатури

Някои комбинации от букви се отпечатват не просто като се наберат различни букви една след друга, а чрез използване на специални символи.[1]

ﬀ ﬁ ﬂ ﬃ… вместо ff fi fl ffi …

Тези така наречени лигатури могат да бъдат забранени с поставянето на командата `\mbox{}` между две съответстващи букви. Това се налага за сложни думи, състоящи се от две думи.

```
\Large Не ‘‘shelfful’’\\
а ‘‘shelf\mbox{}ful’’
```

Не “shelﬀul”
а “shelfful”

2.4.8 Акценти и специални символи

LaTeX поддържа използването на акценти и специални символи за много езици. Таблица 2.2 показва всевъзможните акценти, приложени към буквата o. Ясно е, че на нейно място могат да бъдат и други букви.

За да се постави знак за акцент над буквите i или j, точките над тях трябва да бъдат премахнати. Това става с набора `\i` и `\j`.

```
H\^otel, na\"\i ve, \'el\`eve,\\
sm\o rrebr\o d, !`Se\~norita!,\\
Sch\"onbrunner Schlo\ss{}
Stra\ss e
```

Hôtel, naïve, élève,
smørrebrød, ¡Señorita!,
Schönbrunner Schloß Straße

[1] Лигатурите широко се използват при обработка на текстове на английски език. Доколкото ми е известно, при набор на български език, лигатури не се използват. Обаче, TeX използва механизма на лигатурите за генерация, например, на кавички и тире. — *Бел. ред.*

Таблица 2.2: Акценти и специални символи

ò	\`o	ó	\'o	ô	\^o	õ	\~o
ō	\=o	ȯ	\.o	ö	\"o	ç	\c c
ŏ	\u o	ǒ	\v o	ő	\H o	ǫ	\c o
ọ	\d o	o̲	\b o	o͡o	\t oo		
œ	\oe	Œ	\OE	æ	\ae	Æ	\AE
å	\aa	Å	\AA				
ø	\o	Ø	\O	ł	\l	Ł	\L
ı	\i	ȷ	\j	¡	!\`	¿	?\`

2.5 Поддръжка на чужди езици

Когато пишете документ на език, различен от английски език, LaTeX трябва да бъде конфигуриран по подходящ начин в три насоки:

1. Всички автоматично генерирани текстови стрингове[1] трябва да бъдат преведени на другия език. За много езици тези изменения се постигат с използване на пакета babel (автор Johannes Braams).

2. LaTeX трябва да знае правила за пренасяне за новия език. Включването на правилата за пренасяне в LaTeX е малко по-сложно. То включва обновяване на форматния файл с други разрешени образци на пренасяне. Вашият *Local Guide* [5] трябва да съдържа повече информация за това.

3. Специфичните за езика типографски правила. Например, за френския език, преди всеки символ двоеточие (:) трябва да стои интервал.

Ако вашата система вече е конфигурирана по съответния начин, можете да активирате пакета babel с добавяне на командата

```
\usepackage[език]{babel}
```

след командата `\documentclass`. Списък на всички *езици*, които поддържа вашата LaTeX система, се изобразява всеки път, когато се стартира компилатора. `Babel` автоматично активира правилата за пренасяне на избрания от вас език. Ако форматния файл на вашия LaTeX не поддържа пренасянето за избрания език, `babel` ще работи, но ще забрани пренасянето, което ще се отрази негативно на външния вид на документа.

[1] Съдържание, Списък на фигурите, Библиография ...

Babel освен това въвежда нови команди за някои езици, които опростяват въвеждането на специални символи. Например, немският език съдържа много знаци umlaut (äöü). С използване на babel, вие можете да въведете ö, печатайки `"o` вместо `\"o`.

Ако извикате babel, посочвайки няколко езика,

```
\usepackage[език1,език2]{babel}
```

то за избор на текущия език се използва командата

```
\selectlanguage{език1}
```

Голяма част от съвременните компютърни системи позволяват да се въвеждат специални символи директно от клавиатурата. За да обработва различните входни кодировки, използвани за различните групи езици и/или на различни компютърни платформи, LaTeX използва пакета inputenc:

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

Когато използвате този пакет, вие трябва да имате предвид, че е възможно, други хора да не могат да видят коректно вашите входни файлове на своя компютър, поради използването на друга кодировка. Например, немският акцент ä в система OS/2 се кодира като 132, на Unix системи, използващи ISO-LATIN 1, се кодира като 228, докато в cyrillic encoding cp1251 за Windows, този символ не съществува изобщо. Следователно, трябва да използвате тези възможности с внимание. В зависимост от типа на използваната от вас система, могат да се използват следните кодировки:[1]

Операционна система	кодировка western latin	кодировка cyrillic
Mac	`applemac`	`macukr`
Unix	`latin1`	`koi8-ru`
Windows	`ansinew`	`cp1251`
DOS, OS/2	`cp850`	`cp866nav`

Ако пишете многоезиков документ с конфликтни входни кодировки, можете да превключите на Unicode с помощта на пакета ucs:

```
\usepackage{ucs}
\usepackage[utf8]{inputenc}
```

Това ще ви позволи да създавате входни LaTeX-файлове в `utf8`, многобайтова кодировка, където всеки символ се кодира от един до четири байта.

[1] За по-задълбочено изучаване на поддържаните входни кодировки за езици, основани на Латиница и Кирилица, прочетете съответната документация `inputenc.dtx` и `cyinpenc.dtx`. Раздел 4.6 обяснява как се получава печатна версия на документацията.

Друг случай е кодировката на шрифта. Тя определя, в каква позиция вътре в TEX-шрифта, се намира всяка буква. Множеството входни кодировки може да се изобрази в една кодировка на шрифта, което намалява броя на необходимите набори шрифт. Кодировките на шрифтовете се обработват с пакета fontenc:

```
\usepackage[кодировка]{fontenc}
```

където *кодировка* е изискваната кодировка на шрифта. Могат едновременно да се зареждат няколко кодировки.

Кодировката на LATEX по подразбиране е `OT1`, кодировката на оригиналния шрифт на TEX-Computer Modern. Тя съдържа само 128 символи от 7-битния набор символи ASCII. Когато се изискват акцентирани символи, TEX ги създава, комбинирайки нормален символ с акцент. Въпреки, че резултатът изглежда идеално, този подход спира автоматичното пренасяне вътре в дума, съдържаща акцентирани символи. Освен това, някои латински букви не могат да се създадат чрез комбиниране на нормален символ с акцент, да не говорим за буквите на не-латинските азбуки, като Гръцката азбука или Кирилицата.

За преодоляване на тези ограничения, са били създадени няколко 8-битни набори шрифтове, подобни на CM. *Extended Cork* (EC) шрифтовете в кодировка `T1` съдържат букви и препинателни знаци за повечето европейските езици, основани на Латиница. Наборът шрифтове LH съдържа необходимите букви за отпечатване на документи на езици, използващи Кирилица. Поради големия брой глифове на кирилица, те са организирани в четири кодировки на шрифта: `T2A`, `T2B`, `T2C` и `X2`.[1] Пакетът CB съдържа шрифтове в кодировка `LGR` за набор на гръцки текстове.

Използвайки тези шрифтове, вие можете да подобрите, или въобще да направите възможно пренасянето в не-английски документи. Още едно преимущество при използването на новите CM-подобни шрифтове е, че те предоставят шрифтове от фамилията CM от всички плътности, форми и оптически мащабирани размери на шрифта.

2.5.1 Поддръжка на португалски език

Автор: Demerson Andre Polli <polli@linux.ime.usp.br>

За да включите пренасянето и да смените автоматично генерираните текстове на португалски, използвайте командата

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

Ако сте в Бразилия, заместете езика с `brazilian`.

[1] Езиците, поддържани от всяка от тези кодировки, са изброени в [11].

Таблица 2.3: Преамюл за португалски документи

```
\usepackage[portugese]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
```

Тъй като в португалския език има много акценти, удобно е да използвате

```
\usepackage[latin1]{inputenc}
```

за да бъдат те коректно въведени, както и

```
\usepackage[T1]{fontenc}
```

за включване на правилата за пренасяне.

Вижте в Таблица 2.3 преамбюла, необходим за работа с португалски език. Забележете, че тук се използва кодировка latin1, така че този преамбюл няма да работи на Mac или в DOS. Просто, заместете правилната кодировка за вашата система.

2.5.2 Поддръжка на френски език

Автор: Daniel Flipo <daniel.flipo@univ-lille1.fr>

Ето няколко съвета за това как да се създават в LaTeX френски документи. Можете да заредите поддръжката на френски език с командата

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

Да отбележим, че по исторически причини името на опцията в babel за френски език е *frenchb* или *francais*, но не *french*.

Това включва френските правила за пренасяне, ако вашата LaTeX-система е правилно настроена. Освен това, всички автоматични текстове стават на френски: `\chapter` отпечатва Chapitre, `\today` отпечатва текущата дата на френски и така нататък. Става достъпен също и набор от нови команди, позволяващи леко да се набират френски текстове. За пояснение, вижте Таблица 2.4.

Когато превключите на френски език, ще забележите също и изменение във вида на списъците. За повече информация за това, какво прави `frenchb` опцията на babel и как можете да измените нейното поведение, компилирайте с LaTeX файла `frenchb.dtx` и прочетете получения файл `frenchb.dvi`.

Таблица 2.4: Специални команди за френски език

`\og guillemets \fg{}`	guillemets
`M\up{me}, D\up{r}`	M^{me}, D^{r}
`1\ier{}, 1\iere{}, 1\ieres{}`	1er, 1re, 1res
`2\ieme{} 4\iemes{}`	2^{e} 4es
`\No 1, \no 2`	№1, n^{o} 2
`20~\degres C, 45\degres`	20 °C, 45°
`\bsc{M. Durand}`	M. Durand
`\nombre{1234,56789}`	1,234.567,89

2.5.3 Поддръжка на немски език

Няколко съвета за тези, които искат да създават с LaTeX немски документи. Можете да заредите поддръжката на немски език с командата

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

Това включва немските правила за пренасяне, ако вашата система LaTeX е правилно настроена. Освен това, автоматично генерираните текстове ще станат на немски, например, "Chapter" ще стане "Kapitel". Става достъпен и набор от нови команди, които ви дават възможност да пишете на немски по-бързо, дори когато не използвате пакета inputenc. За пояснение, вижте Таблица 2.5. С използване на inputenc всичко това става ненужно, но вашият текст ще може да се превключва на различни кодировки.

Таблица 2.5: Специални символи за немски език

`"a`	ä	`"s`	ß
``"` ``	„	`"'`	"
`"<` или `\flqq`	«	`">` или `\frqq`	»
`\flq`	‹	`\frq`	›
`\dq`	"		

В немските книги често се срещат френските знаци кавички («guillemets»). Немските типографи обаче, ги използват по друг начин. Цитат в немска книга обикновено изглежда »така«. В немскоговорящите части на Швейцария, типографите използват «guillemets» така, както и във Франция.

Главният проблем е свързан с команди от типа `\flq`. Ако използвате шрифтовете OT1 (а те се включват по подразбиране), тези кавички изглеждат като математическия символ "$\ll$", от което на типографите им прилошава. Шрифтовете в кодировка T1, от друга страна, съдържат необходимите символи. Така че, ако са ви необходими такива кавички, убедете се, че сте включили кодировка T1 (`\usepackage[T1]{fontenc}`).

2.5.4 Поддръжка на корейски език[1]

Използвайки LaTeX за набор на корейски език, трябва да разрешим три проблема:

1. Трябва да можем да редактираме корейски входни файлове. Те трябва да бъдат във формат на плосък текст, но тъй като символите на корейския език не влизат в набора US-ASCII, те ще изглеждат твърде странно в стандартните редактори ASCII. Двете най-широко използвани кодировки за корейски текстови файлове са EUK-KR и нейното съвместимо разширение, използвано в корейската система MS Windows, CP949/Windows-949/UHC. Тези кодировки включват ASCII като подмножество, аналогично на кодировките ISO-8859-x, EUC-JP, Shift_JIS и Big5. От друга страна, слоговете Hangul, знаците Hanjas (използвани в Корея китайски символи), Hangul Jamos, Hirakanas, Katakanas, Greek, Cyrillic и други символи от KS X 1001 се представят чрез два последователни байта. Първият байт има установен старши бит. До средата на 90-те години, установяването на корейското обкръжение в нелокализирана (не-корейска) операционна система, е изисквало значителни усилия и време. Вие можете да прегледате вече доста остарелия `http://jshin.net/faq`, за да получите представа какво е да се използва корейски език в не-корейска операционна система в средата на 90-те. В наше време, всичките три основни операционни системи (Mac OS, Unix, Windows), включват напълно прилична многоезикова поддръжка и възможност за интернационализация, така че редактирането на корейски текстови файлове вече не е проблем, дори в нелокализирана операционна система.

2. TeX и LaTeX са били първоначално създадени за писменост, имаща в азбуката си не повече от 256 символи. За да работят те с езици, в които има много повече символи, такива като корейския[2] и китайския, е бил разработен механизъм на

[1] Разглеждат се няколко въпроса, с които трябва да се справят корейските потребители на LaTeX. Този раздел е написан от Karnes KIM от името на корейския екип преводачи на lshort. Разделът е преведен на английски от SHUN Jungshik и съкратен от Tobi Oetiker.

[2] Корейската азбука Hangul е буквена писменост с 14 основни съгласни и 10 основни гласни (Jamos). За разлика от Латиницата или Кирилицата, отделните символи трябва да бъдат разположени в правоъгълни блокове приблизително със същия размер, като китайските символи. Всеки блок представлява слог. От тези краен брой гласни и съгласни могат да се формират неограничен брой слогове. Обаче, съвременните орфографически стандарти (както в Южна,

под-шрифтове. Той дели единния шрифт CJK, с хиляди или десетки хиляди знаци, на набор от под-шрифтове с 256 знака във всеки. За корейския език има три широко използвани пакета: HLaTeX (автор Un Koanghi), hLaTeXp (автор Cha Jaechoon) и пакет CJK (автор Werner Lemberg).[1] HLaTeX и hLaTeXp са специфични за корейския език и предлагат корейска локализация, както и шрифтова поддръжка. И двата обработват входните файлове в кодировка EUK-KR. HLaTeX може да обработва файлове в кодировки CP949/Windows-949/UHC и UTF-8, ако се използва заедно с Λ, Ω.

Пакетът CJK не е специфичен за корейския език. Той обработва входните файлове в UTF-8, а също в различни кодировки CJK, включително EUK-KR и CP949/Windows-949/UHC. Той може да се използва за набор на документи с многоезично съдържание (особено китайски, японски и корейски). Пакетът CJK не съдържа корейска локализация, аналогична на тази в HLaTeX, и освен това няма толкова много специални корейски шрифтове като HLaTeX.

3. Крайна цел на използването на програма за набор, като TeX и LaTeX, е да се получат документи, отпечатани «естетически» удовлетворително. Може да се твърди, че най-важният елемент на текстo-обработката е набора с добре проектирани шрифтове. Дистрибутивът HLaTeX включва UHC PostScript шрифтове от десет различни семейства и TrueType шрифт Munhwabu[2] от пет различни семейства. Пакетът CJK работи с набор шрифтове, използващи се в старите версии на HLaTeX, а също може да се използва TrueType шрифт cyberbit на компанията Bitstream.

така и в Северна Корея), налагат известни ограничения при формирането на тези блокове. Следователно, съществуват само краен брой орфографически правилни слогове. Корейските кодировки на символите определят индивидуални кодове за всеки от тези слогове (KS X 1001:1998 и KS X 1002:1992). По такъв начин, азбуката Hangul, бидейки буквена, се представя по същия начин, както писмеността на Китай и Япония, с хиляди идеографически/логографически символи. Стандартът ISO 10646/Unicode предлага двата способа за представяне на Hangul, използвани в *съвременна* Корея, кодирайки в допълнение към всички орфографически разрешени слогове Hangul (`http://www.unicode.org/charts/PDF/UAC00.pdf`), още и Conjoining Hangul Jamos (азбука: `http://www.unicode.org/charts/PDF/U1100.pdf`). Едно от най-плашещите предизвикателства в набора на корейски текст с LaTeX и аналогични системи, е поддръжката на средновековните корейски (а вероятно и бъдещи корейски) слогове, които могат да се представят като последователност от знаци Jamos в Unicode. Има надежда, че бъдещите варианти на TeX, като Ω и Λ, рано или късно ще предоставят решение на тази задача, така че корейските лингвисти и историци да успеят да избягат от MS Word, където вече има доста добра поддръжка на средновековния корейски.

[1] Те могат да бъдат получени от `CTAN:/tex-archive/language/korean/HLaTeX/` `CTAN:/tex-archive/language/korean/CJK/` и `http://knot.kaist.ac.kr/htex/`

[2] Министерство на културата на Корея

За използване на пакета HLaTeX за писане на корейски текст, поставете следната декларация в преамбюла на документа:

```
\usepackage{hangul}
```

Тази команда включва корейската локализация. Заглавията на глави, раздели, под-раздели, съдържание и списък на фигурите ще бъдат преведени на корейски, и форматирането на документа ще се измени така, че да следва приетите в Корея правила. Пакетът също така предоставя автоматичен «избор на частици». В корейския език съществуват двойки постфиксни частици, граматически еквивалентни, но различни по форма. Коя частица от двойката е коректна, зависи от това, завършва ли предходния слог на гласна или съгласна. (Всъщност, правилото е доста по-сложно, но това ви дава общата идея.) За корейците не е проблем да изберат правилната частица, но е невъзможно предварително да се определи, какви частици да се използват в препратките и в автоматично генерираните текстове, изменящи се при редактирането на документа. Ръчното поставяне на правилните частици при всяко добавяне/премахване на препратки в текста, или просто при разместване на отделни части на документа, коства доста усилия. HLaTeX облекчава своите потребители в този отегчителен и рискован процес.

В случай, че не ви е нужна корейска локализация, а просто искате да наберете корейски текст, можете да поставите вместо това в преамбюла следните редове:

```
\usepackage{hfont}
```

За повече детайли относно текстo-обработката на корейски текстове с HLaTeX, вижте *Ръководство по HLaTeX*. Посетете WEB-сайта на групата на корейските TeX-потребители (KTUG) на адрес http://www.ktug.or.kr/. Съществува също корейски превод на този документ.

2.5.5 Поддръжка на Кирилица

Автор: Maksym Polyakov <polyama@myrealbox.com>

Пакетът babel, започвайки от версия 3.7, включва поддръжка на кодировка T2* и поддръжка на обработката на български, руски и украински текстове с използване на кирилически букви.

Поддръжката на Кирилицата е основана на стандартния механизъм на LaTeX: пакетите fontenc и inputenc. Но, ако трябва да използвате Кирилица в математически режим, трябва да заредите пакета mathtext преди пакета fontenc:[1]

```
\usepackage{mathtext}
\usepackage[T1,T2A]{fontenc}
\usepackage[cp1251]{inputenc}
\usepackage[english,bulgarian,russian,ukranian]{babel}
```

Казано въобще, babel автоматично избира кодировката на шрифта по подразбиране. За трите изброени езици това е `T2A`. Обаче, документите не са ограничени само с една кодировка на шрифта. За многоезични документи, използващи езици, основани се на Кирилица и Латиница, има смисъл явно да се включи латинската кодировка на шрифтовете. babel се грижи за превключването на съответната кодировка, когато се избират различни езици в документа.

В допълнение към включването на правилата за пренасяне, превеждането на автоматично генерираните стрингове и активирането на някои специфични за езика типографски правила (например, `\frenchspacing`), babel предоставя няколко команди за обработка в съответствие със стандартите на български, руски или украински езици.

И за трите езика се предоставя специфична пунктуация: кирилическо тире за текста (то е малко по-дълго от латинското тире и е оградено с малки интервали), тире за пряка реч, кавички и команди за управление на пренасянето, вж. Таблица 2.6.

Таблица 2.6: Допълнителни дефиниции, въвеждани с опциите на babel за български, руски и украински езици

`"|`	забранява лигатура в тази позиция.
`"-`	дефис, разрешаващ да се пренася остатъка от думата.
`"---`	кирилическо тире в обикновен текст.
`"--~`	кирилическо тире в съставни фамилии.
`"--*`	кирилическо тире за означаване на пряка реч.
`""`	като `"-`, но не генерира знака дефис (за съставни думи с дефис, например `x-""y` или друг знак като "disable/enable").
`"~`	означава съставна дума без разделяне.
`"=`	означава съставна дума с разделяне, с разрешаване пренасянето в думите-компоненти.
`",`	малък интервал за инициали с разрешаване на пренасянето в следващата фамилия.
``"` ``	немски леви двойни кавички (изглежда така: „).
`"'`	немски десни двойни кавички (изглежда така: “).
`"<`	френски леви двойни кавички (изглежда така: $\ll$).
`">`	френски десни двойни кавички (изглежда така: $\gg$).

[1] Ако използвате пакетите $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX, заредете ги също преди пакетите fontenc и babel.

Руската и украинска опции на пакета babel въвеждат команди `\Asbuk` и `\asbuk`, които работят аналогично на командите `\Alph` и `\alph`, но генерират малки и главни букви от руската или украинската азбуки (в зависимост от текущия език в документа). Българската опция на babel въвежда команди `\enumBul` и `\enumLat` (`\enumEng`), които заставят `\Alph` и `\alph` да генерират букви или от българската, или от латинската (английската) азбуки. По подразбиране се генерират български букви.

2.6 Интервали между думите

За получаване на равен десен край на извеждане, LaTeX поставя различни интервали между думите. В края на изречението той слага малко по-голям интервал, което прави текста по-удобен за четене.[1] LaTeX предполага, че изречението завършва с точка, въпросителен или възклицателен знак. Ако точката следва след буква в горен регистър, тя не се счита за край на изречението, тъй като точки след букви в горен регистър обикновено се използват за съкращение.

Всяко изключение от тези предположения трябва да бъде явно определено от автора. Знакът «`\`» пред интервал генерира интервал, който не може да бъде увеличен. Знакът тилда «`~`» генерира интервал, който не може да бъде увеличен и който, освен това, забранява прекъсването на реда. Командата `\@` пред точка определя, че тази точка е край на изречение, дори ако тя стои след буква в горен регистър.

```
Mr.~Smith беше щастлив да я види\\
вж.~Fig.~5\\
Аз харесвам BASIC\@. А ти?
```

Mr. Smith беше щастлив да я види
вж. Fig. 5
Аз харесвам BASIC. А ти?

Допълнителен интервал след точка може да бъде забранен с командата

```
\frenchspacing
```

която указва на LaTeX да *не* въвежда по-голям интервал след точка, отколкото интервал след обикновен символ. Това е често срещано в езици, различни от английския, освен в библиографията. Ако използвате `\frenchspacing`, командата `\@` не е необходима.

2.7 Заглавия, глави и раздели

За да помогнете на читателя да се ориентира във вашия документ, вие трябва да го разделите на глави, раздели и под-раздели. LaTeX поддържа това със специални команди, които имат за аргумент заглавието на раздела. Вашата задача е да ги използвате в точна последователност.

[1] В съответствие с традициите, приети в английския език. — *Бел. ред.*

В класа `article` са достъпни следните команди за секциониране:

```
\section{...}
\subsection{...}
\subsubsection{...}
\paragraph{...}
\subparagraph{...}
```

Ако искате да разбиете вашия документ на части, без изменение на номерацията на разделите/главите, използвайте командата

```
\part{...}
```

Когато работите в класовете `report` или `book`, можете да използвате допълнителна команда за секциониране

```
\chapter{...}
```

Тъй като класът `article` не знае нищо за главите (chapters), статиите доста лесно се добавят в книга като глави. Интервалите между разделите, номерацията и размера на шрифта на заглавията се установяват от LaTeX автоматично.

Две от командите за секциониране са особени:

- Командата `\part` не влияе на последователността на номериране на главите.
- Командата `\appendix` няма аргумент. Тя просто започва да номерира главите с букви вместо цифри.[1]

LaTeX създава Съдържание, като взема заглавията на разделите и номерата на страниците от предходния цикъл на компилация на документа. Командата

```
\tableofcontents
```

извежда Съдържанието на това място, където тя се е появила. Нов документ трябва да бъде компилиран с LaTeX два пъти, за да се получи правилно Съдържание. Понякога може да се наложи да се компилира документа и трети път. LaTeX ще ви каже, когато това е необходимо.

Всички изброени по-горе команди за секциониране съществуват също и във варианти със звездичка. Вариант със звездичка се получава чрез добавяне на `*` към името на командата. Така се генерират заглавия на раздели, които не се номерират и не се появяват в Съдържанието. Например, командата `\section{Помощ}` става `\section*{Помощ}`.

Обикновено заглавията на разделите се появяват в съдържанието точно в същия вид, в който те са въведени в текста. Понякога това е невъзможно поради това,

[1] В класа `article` се изменя номерацията на разделите.

че заглавието е твърде дълго, за да се постави в съдържанието. Тогава, елементът на съдържанието може бъде определен като незадължителен аргумент пред самото заглавие.

```
\chapter[Заглавие за съдържанието]{Това е твърде дълго
 и отегчително заглавие, появяващо се в текста}
```

Заглавната страница на целия документ се генерира с командата

```
\maketitle
```

Нейното съдържание трябва да бъде определено с командите

`\title{...}`, `\author{...}` и `\date{...}`

преди извикване на `\maketitle`. Аргументът на командата `\author` може да съдържа няколко имена, разделени с командите `\and`.

Пример на някои от посочените команди може да бъде намерен на Фигура 1.2 на страница 8.

Освен описаните по-горе команди за секциониране, LaTeX 2_ε въвежда три допълнителни команди за използване с класа `book`. Те са полезни за деление на вашата публикация. Тези команди изменят заглавията на главите и номерацията на страниците така, както това се очаква от книга:

`\frontmatter` трябва да бъде първата команда след `\begin{document}`. Тя превключва номерацията на страниците с използване на римски цифри, а разделите не са номерирани. Това е подобно на използването на команди за секциониране със звездички (например, `\chapter*{Предговор}`), но основната разлика е, че заглавията на разделите ще се появяват в Съдържанието.

`\mainmatter` се поставя непосредствено преди първата глава на книгата. Тя включва номерацията на страниците с арабски цифри и изчиства брояча на страниците.

`\appendix` отбелязва началото на допълнителния материал във вашата книга. След тази команда главите ще се номерират с букви.

`\backmatter` трябва да бъде въведена преди последните части на вашата книга, такива като Библиография и Азбучен указател. В стандартните класове документи, няма видим ефект от това.

2.8 Препратки

В книги, доклади и статии често се срещат препратки към фигури, таблици и специални части на текста. LaTeX предоставя следните команди за препратки:

`\label{`*етикет*`}`, `\ref{`*етикет*`}` и `\pageref{`*етикет*`}`

където *етикет* е идентификатор, избран от потребителя. LaTeX заменя `\ref` с номера на раздела, под-раздела, фигурата, таблицата или теоремата, след която е била използвана съответната команда `\label`. `\pageref` печата номера на тази страница, на която се е появила съответната команда `\label`.[1] Така, както и в случая със заглавията на раздели, тук също се използват номерата от предишната компилация.

```
Препратката на този
раздел~\label{sec:this}
изглежда  така: <<вж.
Раздел~\ref{sec:this} на
стр.~\pageref{sec:this}.>>
```

Препратката на този раздел изглежда така: «вж. Раздел 2.8 на стр. 37.»

2.9 Забележки под печатното поле

С командата

`\footnote{`*Текст на забележката*`}`

се печата забележка под линия на текущата страница. Забележката под печатното поле трябва винаги да се поставя след думата или изречението, към което се отнася. Следователно, забележки, които се отнасят към изречение, или част от него, трябва да се поставят след точката или запетаята.[2]

```
Потребителите на \LaTeX{}
често употребяват
забележки под поле.\footnote{%
  Това е забележка.}
```

Потребителите на LaTeX често употребяват забележки под поле.[a]

[a] Това е забележка.

[1] Забележете, че тези команди не знаят, на какво точно се препращат. `\label` просто съхранява последния автоматично генериран номер.

[2] Да отбележим, че забележките отвличат читателя от основния текст на документа. Освен това всички четат забележките, защото ние хората сме любопитни създания. Затова, най-добре е да се постараете да включите всичко, което искате да кажете, в тялото на документа.[3]

[3] Не винаги пътепоказателят сочи там, накъдето трябва :-).

2.10 Подчертани думи

В ръкописи, напечатани на пишеща машина, `важните думи се отделят с подчертаване`:

```
\underline{текст}
```

В печатните издания обаче, тези думи се подчертават като се отпечатват в наклонен шрифт (*курсив*). LaTeX предоставя командата

```
\emph{текст}
```

за *подчертаване* на текст. Нейният аргумент е текста за подчертаване . Какво всъщност прави тази команда със своя аргумент, зависи от контекста:

```
Ако използвате
\emph{подчертаване вътре във
вече подчертан текст, то \LaTeX{}
използва \emph{прав} шрифт.}
```

Ако използвате *подчертаване вътре във вече подчертан текст, то LaTeX използва* прав *шрифт.*

Забележете разликата между командите за *подчертаване* и смяна на *шрифта*:

```
\textit{Можете също така да
  \emph{подчертаете} текст,
  ако го наберете в italic,}
\textsf{ в sans-serif}
  \emph{шрифт}
\texttt{или в
  \emph{typewriter} стил.}
```

Можете също така да подчертаете *текст, ако го наберете в italic,* в sans-serif *шрифт* или в *typewriter* стил.

2.11 Среди

За обработка на специални видове текст, LaTeX дефинира множество среди за различни видове форматиране:

```
\begin{среда}  текст  \end{среда}
```

където *среда* определя името на средата. Среда може да се извиква от друга среда, като се съблюдава реда на отваряне и затваряне на средата:

```
\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}
```

В следващите под-раздели се разказва за всички важни среди.

2.11.1 Списък, изброяване и описание

Средата `itemize` е подходяща за прости списъци, средата `enumerate` – за номерирани списъци, а средата `description` – за описание.

```
\flushleft
\begin{enumerate}
\item Можете по ваш вкус да
смесвате средите на списъците:
\begin{itemize}
\item Но това може да изглежда
глупаво.
\item[-] с тире.
\end{itemize}
\item Затова запомнете:
\begin{description}
\item[Глупавите] неща няма да
станат по-значителни от това,
че са в списък.
\item[Умните] неща обаче,
могат да бъдат представени
по-добре в списък.
\end{description}
\end{enumerate}
```

1. Можете по ваш вкус да смесвате средите на списъците:
   - Но това може да изглежда глупаво.
   - \- с тире.
2. Затова запомнете:

   Глупавите неща няма да станат по-значителни от това, че са в списък.

   Умните неща обаче, могат да бъдат представени по-добре в списък.

2.11.2 Подравняване вляво, вдясно и центриране

Средите `flushleft` и `flushright` генерират параграфи, които са подравнени вляво или вдясно. Средата `center` генерира центриран текст. Ако не използвате `\\` за определяне на местата за прекъсване на редовете, LaTeX ги определя автоматично.

```
\begin{flushleft}
  Този текст\\ е подравнен вляво.
  \LaTeX{} не се старае да направи
  всички редове с еднаква дължина.
\end{flushleft}
```

Този текст
е подравнен вляво. LaTeX не се старае да направи всички редове с еднаква дължина.

```
\begin{flushright}
Този текст\\ е подравнен вдясно.
  \LaTeX{} не се старае да направи
  всички редове с еднаква дължина.
\end{flushright}
```

Този текст
е подравнен вдясно. LaTeX не се старае да направи всички редове с еднаква дължина.

```
\begin{center}
  В центъра\\на Земята
\end{center}
```

В центъра
на Земята

2.11.3 Цитати и стихове

Средата `quote` е полезна за цитати, важни фрази и примери.

```
Типографско правило за
дължина на реда:
\begin{quote}
  По правило, реда трябва
  да съдържа не повече от
  66~символа.
\end{quote}
Затова \LaTeX{} прави такива
широки полета на страницата.
Затова във вестниците често се
използва набор в няколко колони.
```

Типографско правило за дължина на реда:

> По правило, реда трябва да съдържа не повече от 66 символа.

Затова LaTeX прави такива широки полета на страницата. Затова във вестниците често се използва набор в няколко колони.

Съществуват още две подобни среди: `quotation` и `verse`. Средата `quotation` е полезна за по-дълги цитати, състоящи се от няколко параграфа, защото тя прави отстъп на първия ред от всеки параграф. Средата `verse` се използва за стихотворения, при които са важни прекъсванията на редовете. Редовете се отделят чрез въвеждане на \\ в края на реда и празен ред след всяка строфа.

```
Аз знам само едно английско
стихотворение наизуст. То е
за Humpty Dumpty:
\begin{flushleft}
\begin{verse}
Humpty Dumpty sat on a wall:\\
Humpty Dumpty had a great fall.\\
All the King’s horses and all
the King’s men\\
Couldn’t put Humpty together again.
\end{verse}
\end{flushleft}
```

Аз знам само едно английско стихотворение наизуст. То е за Humpty Dumpty:

> Humpty Dumpty sat on a wall:
> Humpty Dumpty had a great fall.
> All the King’s horses and all the King’s
> men
> Couldn’t put Humpty together again.

2.11.4 Резюме

В научните публикации е прието да се започва с резюме, което дава на читателя бърз преглед на това, какво да очаква. LaTeX осигурява за тази цел средата `abstract`. Обикновено `abstract` се използва в документи, които се обработват в класа article.

```
\begin{abstract}
Това е резюме.
\end{abstract}
```

Това е резюме.

2.11.5 Буквално възпроизвеждане

Текст, заграден с `\begin{verbatim}` и `\end{verbatim}` ще бъде директно напечатан, като набран на пишеща машина, с всички интервали и нови редове, без изпълняване на каквито и да било команди на LaTeX.

Вътре в параграфа аналогична функция изпълнява командата

`\verb+`*текст*`+`

Тук «+» е само пример на символ-ограничител. Вие можете да използвате всеки символ, освен букви, «*» или интервал. Много LaTeX-примери в тази книжка са набрани с тази команда.

```
Командата \verb|\ldots| \ldots

\begin{verbatim}
10 PRINT "HELLO WORLD ";
20 GOTO 10
\end{verbatim}
```

Командата `\ldots` …

```
10 PRINT "HELLO WORLD ";
20 GOTO 10
```

```
\begin{verbatim*}
Варианта на средата
verbatim         със
звездичка подчертава
интервалите    в  текста
\end{verbatim*}
```

```
Варианта␣на␣средата
verbatim␣␣␣␣␣␣␣␣␣със
звездичка␣подчертава
интервалите␣␣␣␣в␣␣текста
```

Командата `\verb` също може да се използва по аналогичен начин със звездичка:

```
\verb*|ето   така :-) |
```

```
ето␣␣␣така␣:-)␣
```

Средата `verbatim` и командата `\verb` не могат да се използват вътре в параметри на други команди.

2.11.6 Таблици

Средата `tabular` може да се използва за отпечатване на красиви таблици, със или без хоризонтални и вертикални линии. LaTeX определя ширината на колоните автоматично.

Аргументът *спецификация* на командата

`\begin{tabular}[`*позиция*`]{`*спецификация*`}`

дефинира формáта на таблицата. Използвайте `l` за колона с ляво-подравнен текст, `r` – за дясно-подравнен текст и `c` за центриран текст, `p{`*ширина*`}` за колона, съдържаща свързан текст с пренасяне на думите, и `|` за вертикална линия.

Ако текстът в колоната е твърде широк за страницата, LaTeX няма да може автоматично да го пренесе. Използвайки `p{`*ширина*`}`, вие можете да дефинирате специален тип колона, в която текстът ще се пренася като в нормален параграф.

Аргументът *позиция* определя вертикалното положение на таблицата относно базовата линия на околния текст. Използвайте една от буквите `t`, `b` и `c` за да позиционирате таблицата горе, долу или в центъра спрямо основната линия.

Вътре в средата `tabular` знакът «`&`» прехвърля към следващата колона, командата `\\` започва нов ред, а `\hline` въвежда хоризонтална линия. Можете да добавяте непълни линии с помощта на командата`\cline{j-i}`, където `j` и `i` са номерата на колоните, над които трябва да се простира линията.

```
\begin{tabular}{|r|l|}
\hline
7C0 & шестнадесетично \\
3700 & осмично \\ \cline{2-2}
11111000000 & двоично \\
\hline \hline
1984 & десетично \\
\hline
\end{tabular}
```

7C0	шестнадесетично
3700	осмично
11111000000	двоично
1984	десетично

```
\begin{tabular}{|p{4.7cm}|}
\hline Добре дошли в параграф
в рамка. Надяваме се, всичко
това да ви хареса.\\
\hline
\end{tabular}
```

Добре дошли в параграф в рамка. Надяваме се, всичко това да ви хареса.

Разделителят на колоните може да се задава с конструкцията `@{...}`. Тази команда премахва интервала между колоните и го заменя с това, което е включено между фигурните скоби. Едно от честите приложения на тази команда е показано по-долу, в задачата за подравняване по десетичната точка. Друго възможно приложение е премахването на водещия интервал в таблица с `@{}`:

```
\begin{tabular}{@{} l @{}}
\hline
няма водещ интервал\\
\hline
\end{tabular}
```

няма водещ интервал

```
\begin{tabular}{l}
\hline
водещ интервал вляво и вдясно\\
\hline
\end{tabular}
```

водещ интервал вляво и вдясно

Тъй като няма вграден способ за подравняване на числови колони по десетичната точка,[1] ние можем да «надхитрим» TEX и да направим това с помощта на две колони: дясно-подравнена цяла част и ляво-подравнена дробна. Командата `@{.}` в реда `\begin{tabular}` заменя нормалния интервал между колоните просто с «.», давайки ефект на една колона, подравнена по десетичната точка. Не забравяйте да замените във вашите числа десетичната точка с разделител между колони (`&`)! Надпис над колона може да се постави над нашата числова «колона» с командата `\multicolumn`:

```
\begin{tabular}{c r @{.} l}
Израз с $\pi$      &
\multicolumn{2}{c}{Стойност} \\
\hline
$\pi$              & 3&1416  \\
$\pi^{\pi}$        & 36&46   \\
$(\pi^{\pi})^{\pi}$ & 80662&7 \\
\end{tabular}
```

Израз с π	Стойност
π	3.1416
π^{π}	36.46
$(\pi^{\pi})^{\pi}$	80662.7

```
\begin{tabular}{|c|c|}
\hline
\multicolumn{2}{|c|}{Ene} \\
\hline
Mene & Muh! \\
\hline
\end{tabular}
```

<table><tr><td colspan="2">Ene</td></tr><tr><td>Mene</td><td>Muh!</td></tr></table>

Материал, набиран в средата `tabular`, винаги е неделим и се разполага на една страница. Ако искате да набирате дълги таблици, може да видите пакетите supertabular или longtable.

2.12 Плаващи обекти

В наши дни повечето публикации съдържат много фигури и таблици. Тези елементи се нуждаят от специално отношение, защото те не могат да бъдат разбити между страниците. Един изход би бил да се започва нова страница всеки път, когато се срещне фигура или таблица, твърде голяма, за да се събере на текущата страница. Този подход би довел до това, страниците да остават частично празни, което изглежда много лошо.

Решението на този проблем е: всяка фигура или таблица, която не се събира на текущата страница да „плава“, премествайки се на следващата страница, а в същото време текущата страница да се запълва с текст. LATEX предлага две среди за плаващите обекти, една за таблици и друга – за фигури. За да се възползвате напълно от тези две среди, е важно да разберете приблизително, как LATEX обработва плаващите обекти. Иначе те могат да станат източник на разочарование поради това, че LATEX ги поставя не там, където вие искате.

[1] Ако на вашата система е установен комплектът ‘tools’, обърнете внимание на пакета dcolumn.

Нека първо да разгледаме командите, предлагани от LaTeX за плаващите обекти:

Всеки материал, включен в средата `figure` или `table`, се определя като «плаващ» обект. И двете среди имат незадължителен параметър

`\begin{figure}[`*спецификация на позицията*`]` или
`\begin{table}[`*спецификация на позицията*`]`

наречен *спецификация на позицията*. Този параметър се използва за указване на LaTeX, къде може да се премества плаващия обект. *Спецификация на позицията* се конструира като поредица от *ключове за разрешени позиции на плаващия обект*, (вж. Таблица 2.7).

Например, дадена таблица може да започва със следния ред:

```
\begin{table}[!hbp]
```

Спецификацията на позицията `[!hbp]` позволява на LaTeX да постави таблицата точно на това място (`h`) или долу на същата страница (`b`), или на отделна страница (`p`) с плаващи обекти, дори всичко това да не изглежда много добре (`!`). Ако не е зададена никаква спецификация на позицията, стандартните класове предполагат `[tbp]`.

LaTeX разполага всеки срещнат плаващ обект в съответствие със зададената от автора спецификация. Ако обектът не може да се събере на текущата страница, той се отстранява и се поставя на опашка от фигури или на опашка от таблици.[1] Когато започне нова страница, LaTeX проверява, може ли да запълни специална страница с плаващи обекти от опашките. Ако това не е възможно, първият обект от всяка опашка се разглежда като току що появил се в текста; LaTeX отново се опитва да

[1]Тези опашки се подчиняват на дисциплината *FIFO*: 'пръв влязъл – пръв излязъл'.

Таблица 2.7: Ключове за разрешени позиции на плаващ обект

Ключ	Разрешава да се разположи обекта ...
`h`	*тук*, на същото място в текста, където се е появил плаващият обект. Обикновено се използва за малки обекти.
`t`	*горе* на страницата
`b`	*долу* на страницата
`p`	на *специална страница*, съдържаща само плаващи обекти.
`!`	без разглеждане на повечето вътрешни параметри,[a] които могат да попречат за разполагането на този обект.

Забележка: `pt` и `em` са единици за измерване на дължина в TeX. Подробности за единиците и размерностите вижте в Таблица 6.5 на страница 109.

[a]Такива, като максимален брой плаващи обекти, разрешени на една страница

ги постави в съответствие с тяхната спецификация (с изключение на „h“, което вече е невъзможно). Всеки нов плаващ обект, който се появи в текста, се поставя на съответната опашка. LaTeX стриктно спазва реда, в който са били срещнати плаващи обекти от съответния тип. Затова фигура, която не може да се позиционира, отхвърля всички следващи фигури към края на документа. Следователно:

> Ако LaTeX не позиционира плаващите обекти както вие очаквате, то често това се дължи само на един обект, който е «задръстил» едната или другата опашка.

Макар и да е възможно да се задават в LaTeX единични спецификации на позицията на плаващ обект, това може да доведе до проблеми. Ако обектът не се събира на посоченото място, той остава на опашката, като блокира следващите плаващи обекти. В частност, никога не трябва да използвате опцията `[h]`; това е толкова лошо, че в съвременните версии на LaTeX той автоматично се заменя на `[ht]`.

След обясненията на тези трудности остава да отбележим още няколко неща за средите `table` и `figure`. Чрез командата

`\caption{`*текст на заглавието*`}`

вие можете да задавате заглавие на плаващия обект. Нарастващият номер и надписът «Фигура» или «Таблица», се добавят от LaTeX.

Двете команди

`\listoffigures` и `\listoftables`

работят аналогично на командата `\tableofcontents`, печатайки съответно списък на фигурите или таблиците. Тези списъци включват целия аргумент на командата `\caption`, така че, ако сте склонен да използвате дълги заглавия, трябва да предоставите техен кратък вариант за включване в списъците. Това се осъществява чрез въвеждане на кратка версия в квадратни скоби след командата `\caption`.

```
\caption[Кратко]{Дддддъъъъъъъъъъъълго}
```

С помощта на `\label` и `\ref` могат да се правят препратки от вашия текст към плаващия обект.

Следният пример рисува квадрат и го поставя в документа. Тази техника може да се използва, за да се остави в документа място за изображение, което ще се постави по-късно.

```
Фигура~\ref{white} е пример на Поп-Арт.
\begin{figure}[!hbp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{Пет на пет в сантиметри} \label{white}
\end{figure}\\
```

В горния пример LaTeX *много силно* (`!`) ще се старае да позиционира фигурата точно *на място* (`h`).[1] Ако това е невъзможно, той ще се опита да я разположи *долу на страницата* (`b`). Ако не му се отдаде да постави фигурата на текущата страница, той ще провери, може ли да създаде страница с плаващи обекти, съдържаща тази фигура и, вероятно, някои таблици от опашката с таблици. Ако няма достатъчно материал за специална страница с плаващи обекти, LaTeX започва нова страница и отново разглежда фигурата, все едно че тя току що се е появила в текста.

В определени случаи може да е необходимо да се използва команда

`\clearpage` или дори `\cleardoublepage`

Командата `\clearpage` заповядва на LaTeX, незабавно да постави всички плаващи обекти, останали в опашките, след което да започне нова страница. `\cleardoublepage`, освен това, започва нова дясно-стояща страница.

По нататък в това въведение, вие ще разберете, как да включвате във вашите документи LaTeX-рисунки във формат PostScript.

2.13 Защита на крехки команди

Текст, зададен в аргумента на команди, подобни на `\caption` или `\section`, може да се появява в документа повече от един път (например, в съдържанието, в колонтитулите и в тялото на документа). Някои команди не работят, ако се използват в аргумента на команда от типа `\section`. Те се наричат «крехки» команди. Например, «крехки» са командите `\footnote` или `\phantom`. Тези «крехки» команди се нуждаят от защита. Вие можете да ги защитите, като поставите пред тях командата `\protect`.

Командата `\protect` се отнася само за командата, непосредствено следваща след нея; дори не и към нейните аргументи. В голяма част от случаите, излишна команда `\protect` няма да навреди.

```
\section{Аз съм  внимателен
     \protect\footnote{и защитавам своите бележки под поле}}
```

[1]Предполага се, че опашката от фигури е празна

3 Набор на математически формули

Ето сега сме готови! В тази глава ще се срещнем с най-голямата сила на TEX: математическата обработка. Но имайте предвид, че тази глава не навлиза дълбоко в същността на темата. Макар, за много хора, изложените тук неща да са достатъчно, не се отчайвайте, ако не успеете да намерите решение, съответстващо на вашите нужди за обработка на математически формули. Твърде вероятно е, вашият проблем да се решава в AMS-LaTeX[1].

3.1 Общи сведения

LaTeX има специален режим за обработка на математика. Математически формули могат да бъдат набрани вътре в параграфа, или параграфът може да бъде разбит на части, за да се отпечатат те отделно. Математически текст вътре в параграф се въвежда между `\(` и `\)`, между `$` и `$` или между `\begin{math}` и `\end{math}`.

```
Събирайки $a$ на квадрат с
$b$ на квадрат, получаваме
$c$ на квадрат. Или
на езика на математиката:
$c^{2}=a^{2}+b^{2}$
```

Събирайки a на квадрат с b на квадрат, получаваме c на квадрат. Или на езика на математиката: $c^2 = a^2 + b^2$

```
\TeX{} се произнася като
\(\tau\epsilon\chi\).\\[6pt]
100~м$^{3}$ вода.\\[6pt]
Това излиза от моето
\begin{math}\heartsuit\end{math}
```

TEX се произнася като $\tau\epsilon\chi$.

100 м^3 вода.

Това излиза от моето $\heartsuit$

Когато искате да «отделите» големи математически уравнения или формули от останалата част от параграфа, е за предпочитане те да се поставят на отделни редове, без да се прекъсва параграфа. За да направите това, можете да ги оградите с `\[` и `\]` или с `\begin{displaymath}` и `\end{displaymath}`.

[1] *Американското математическо общество* е изработило мощно разширение на LaTeX. Много примери в тази глава използват това разширение. То е включено във всички съвременни дистрибутиви на TEX. Ако във вашето то отсъства, можете да го получите от адрес `CTAN:/tex-archive/macros/latex/required/amslatex`.

```
Събирайки $a$ на квадрат с
$b$ на квадрат, получаваме
$c$ на квадрат. Или
на езика на математиката:
\begin{displaymath}
  c^{2}=a^{2}+b^{2}
\end{displaymath}
или можете да изразите това
по-кратко: \[a+b=c\]
```

Събирайки a на квадрат с b на квадрат, получаваме c на квадрат. Или на езика на математиката:

$$c^{2}=a^{2}+b^{2}$$

или можете да изразите това по-кратко:

$$a+b=c$$

Ако искате LaTeX да номерира вашите уравнения, можете да използвате средата `equation`. Вие можете при това да поставите на уравнението етикет `\label` и да правите препратки към него от произволно място в текста с командите `\ref` или `\eqref`:

```
\begin{equation}
  \label{eq:eps}
  \epsilon > 0
\end{equation}
От (\ref{eq:eps})
заключаваме \ldots{}
От \eqref{eq:eps}
ние правим същото.
```

$$\epsilon > 0 \tag{3.1}$$

От (3.1) заключаваме … От (3.1) ние правим същото.

Забележете разликата в стила на отпечатване на уравнения вътре в параграф и когато те са «отделени»:

```
$\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}$
```

$\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$

```
\begin{displaymath}
  \lim_{n \to \infty}
  \sum_{k=1}^n \frac{1}{k^2}
  = \frac{\pi^2}{6}
\end{displaymath}
```

$$\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

Има разлики между *математически режим* и *текстов режим*. Например, в *математически режим*:

1. Повечето интервали и прекъсвания на редове не се вземат под внимание, тъй като всички интервали или произлизат от логиката на математическите изрази, или трябва да се задават в явен вид с команди като `\,` или `\quad`, или `\qquad`.

2. Празни редове не се допускат. Всяка формула заема само един параграф.

3. Всяка буква се разглежда като име на променлива и се отпечатва като такава. Ако искате да въведете нормален текст вътре във формула (нормален прав шрифт с нормални интервали), то е необходимо да го въведете с командите `\textrm{...}` (вж. също Раздел 3.7 на стр. 56).

```
\begin{equation}
  \forall x \in \mathbf{R}:
  \qquad x^{2} \geq 0
\end{equation}
```

$$\forall x \in \mathbf{R}: \qquad x^2 \geq 0 \tag{3.2}$$

```
\begin{equation}
  x^{2} \geq 0\qquad
  \textrm{за всички }x\in
  \mathbf{R}
\end{equation}
```

$$x^2 \geq 0 \qquad \textrm{за всички } x \in \mathbf{R} \tag{3.3}$$

Математиците са много строги по отношение на използваните символи: прието е да се използват "ажурни плътни символи", които се получават с командата `\mathbb` от пакетите amsfonts или amssymb. Последният пример сега изглежда така:

```
\begin{displaymath}
  x^{2} \geq 0\qquad
  \textrm{за всяко }x\in
  \mathbb{R}
\end{displaymath}
```

$$x^2 \geq 0 \qquad \textrm{за всяко } x \in \mathbb{R}$$

3.2 Групиране в математически режим

Повечето команди в математически режим действат само на следващия символ. Така че, ако искате командата да влияе на няколко символа, трябва да ги групирате заедно с помощта на фигурни скоби: `{...}`.

```
\begin{equation}
  a^x+y \neq a^{x+y}
\end{equation}
```

$$a^x + y \neq a^{x+y} \tag{3.4}$$

3.3 Елементи на математическите формули

В този раздел ще бъдат описани най-важните команди, използвани в математическата обработка. Подробен списък на командите за набор на математически символи вижте в Раздел 3.10 на страница 60.

Малки гръцки букви се въвеждат като `\alpha`, `\beta`, `\gamma`, …, главните букви се въвеждат като `\Gamma`, `\Delta`, …[1]

```
$\lambda,\xi,\pi,\mu,%
\Phi,\Omega$
```

$\lambda, \xi, \pi, \mu, \Phi, \Omega$

Горни и долни индекси се въвеждат с помощта на символите «`^`» и «`_`».

```
$a_{1}$ \qquad $x^{2}$ \qquad
$e^{-\alpha t}$ \qquad
$a^{3}_{ij}$\\
$e^{x^2} \neq {e^x}^2$
```

$a_{1} \qquad x^{2} \qquad e^{-\alpha t} \qquad a^{3}_{ij}$
$e^{x^2} \neq {e^x}^2$

Квадратен корен се въвежда като `\sqrt`, корен n-ти се генерира с `\sqrt[`n`]`. Размерът на знака за корен се избира от LaTeX автоматично. Ако ви е нужен само знак за корен, използвайте `\surd`.

```
$\sqrt{x}$ \qquad
$\sqrt{ x^{2}+\sqrt{y} }$
\qquad $\sqrt[3]{2}$\\[3pt]
$\surd[x^2 + y^2]$
```

$\sqrt{x} \qquad \sqrt{ x^{2}+\sqrt{y} } \qquad \sqrt[3]{2}$
$\surd[x^2 + y^2]$

Командите `\overline` и `\underline` създават **хоризонтални линии** непосредствено над или под израза.

```
$\overline{m+n}$
```

$\overline{m+n}$

Командите `\overbrace` и `\underbrace` създават дълги **хоризонтални фигурни скоби** непосредствено над или под израза.

```
$\underbrace{ a+b+\cdots%
  +z }_{26}$
```

$\underbrace{ a+b+\cdots+z }_{26}$

За добавяне към променливите на знаци за математически акценти, такива, като малки стрелки или знак тилда, можете да използвате командите, изброени в Таблица 3.1 на страница 60. Широки «шапки» и тилди, обхващащи няколко символи, се генерират с командите `\widetilde` и `\widehat`. Символът «`'`» дава знак за производна.

```
\begin{displaymath}
y=x^{2}\qquad y'=2x \qquad y''=2
\end{displaymath}
```

$$y=x^{2}\qquad y'=2x \qquad y''=2$$

[1] В LaTeX 2_ε не се дефинира главна буква «алфа», защото тя изглежда по същия начин, както и латинската «A». При новата кодировка на математиката това ще бъде изменено.

Векторите често се означават чрез добавяне на малък знак стрелка над променливата. Това се прави с командата `\vec`. За означаване на вектор от A до B са полезни двете команди `\overrightarrow` и `\overleftarrow`.

```
\begin{displaymath}
  \vec a\quad\overrightarrow{AB}
\end{displaymath}
```

$$\vec a\quad\overrightarrow{AB}$$

Обикновено знакът точка, означаващ операция умножение, не се набира в явен вид. Обаче, понякога той се записва, за да помогне на читателя да групира формулата. В тези случаи се използва `\cdot`:

```
\begin{displaymath}
v = {\sigma}_1 \cdot {\sigma}_2
    {\tau}_1 \cdot {\tau}_2
\end{displaymath}
```

$$v = {\sigma}_1 \cdot {\sigma}_2 {\tau}_1 \cdot {\tau}_2$$

Имената на функции от типа lg често се записват в прав шрифт, а не в наклонен като променливите. Затова LaTeX предлага следните команди за набор на имена на най-важните функции:

`\arccos`	`\cos`	`\csc`	`\exp`	`\ker`	`\limsup`	`\min`
`\arcsin`	`\cosh`	`\deg`	`\gcd`	`\lg`	`\ln`	`\Pr`
`\arctan`	`\cot`	`\det`	`\hom`	`\lim`	`\log`	`\sec`
`\arg`	`\coth`	`\dim`	`\inf`	`\liminf`	`\max`	`\sin`
`\sinh`	`\sup`	`\tan`	`\tanh`			

```
\[\lim_{x \rightarrow 0}
\frac{\sin x}{x}=1\]
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x}=1$$

За функцията модул има две команди: `\bmod` за бинарен оператор «$a \bmod b$» и `\pmod` за изрази от вида «$x \equiv a \pmod{b}$».

```
$a\bmod b$\\
$x\equiv a \pmod{b}$
```

$a\bmod b$
$x\equiv a \pmod{b}$

Обикновена **дроб** се записва с командата `\frac{...}{...}`. Често се предпочита форма с ко́са черта 1/2, защото тя се вижда по-добре при малко количество „дробен материал“.

```
$1\frac{1}{2}$~часа
\begin{displaymath}
  \frac{ x^{2} }{ k+1 }\qquad
  x^{ \frac{2}{k+1} }\qquad
  x^{ 1/2 }
\end{displaymath}
```

$1\frac{1}{2}$ часа

$$\frac{ x^{2} }{ k+1 }\qquad x^{ \frac{2}{k+1} }\qquad x^{ 1/2 }$$

За отпечатване на биномни коефициенти или аналогични структури може да се използва командата `\binom` от пакета `amsmath`.

```
\begin{displaymath}
\binom{n}{k}\qquad\mathrm{C}_n^k
\end{displaymath}
```

$$\binom{n}{k}\qquad\mathrm{C}_n^k$$

За бинарни отношения е удобно да се разполагат символи един над друг. Командата `\stackrel` набира символа, зададен като първи аргумент, в шрифт с размер на индекс, и го разполага над втория аргумент, отпечатван в обичайна позиция:

```
\begin{displaymath}
\int f_N(x) \stackrel{!}{=} 1
\end{displaymath}
```

$$\int f_N(x) \stackrel{!}{=} 1$$

Оператор за интеграл се генерира с командата `\int`, **оператор за сума** – с командата `\sum`, **оператор за произведение** – с командата `\prod`. Горните и долните граници се указват с помощта на знаците «^» и «_», като горни и долни индекси[1].

```
\begin{displaymath}
\sum_{i=1}^{n} \qquad
\int_{0}^{\frac{\pi}{2}} \qquad
\prod_\epsilon
\end{displaymath}
```

$$\sum_{i=1}^{n} \qquad \int_{0}^{\frac{\pi}{2}} \qquad \prod_\epsilon$$

За да имате по-голям контрол при разполагането на индексите в сложни изрази, `amsmath` предоставя още два инструмента: командата `\substack` и средата `subarray`:

```
\begin{displaymath}
\sum_{\substack{0<i<n \\ 1<j<m}}
   P(i,j) =
\sum_{\begin{subarray}{l} i\in I\\
         1<j<m
      \end{subarray}}     Q(i,j)
\end{displaymath}
```

$$\sum_{\substack{0<i<n \\ 1<j<m}} P(i,j) = \sum_{\begin{subarray}{l} i\in I\\ 1<j<m \end{subarray}} Q(i,j)$$

TEX предоставя много символи за **скоби** и други ограничители (например $[\ \langle\ \|\ \updownarrow$). Кръгли и квадратни скоби могат да се въвеждат със съответните клавиши, фигурни скоби – с `\{`, но всички други ограничители се генерират със специални команди (например `\updownarrow`). Списък на достъпните ограничители вижте в Таблица 3.8 на страница 62.

```
\begin{displaymath}
{a,b,c}\neq\{a,b,c\}
\end{displaymath}
```

$$a,b,c \neq \{a,b,c\}$$

[1] $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX, освен това, поддържа многоредови горни и долни индекси.

Ако поставите пред отварящ ограничител командата `\left` или пред затварящ – `\right`, то TEX автоматично избира точния размер на ограничителя. Забележете, че трябва да затваряте всяко `\left` със съответното `\right` и че размерът на ограничителите се определя коректно само ако двата се отпечатват на един и същ ред. Ако не искате да имате десен ограничител, използвайте невидимия ограничител „`\right.`“!

```
\begin{displaymath}
1 + \left( \frac{1}{ 1-x^{2} }
   \right) ^3
\end{displaymath}
```

$$1 + \left(\frac{1}{1-x^{2}} \right)^3$$

В някои случаи е необходимо да се определи ръчно точният размер на математическия ограничител. Това може да се направи чрез командите `\big`, `\Big`, `\bigg` и `\Bigg`, използвани като префикси към повечето команди за ограничители.[1]

```
$\Big( (x+1) (x-1) \Big) ^{2}$\\
$\big(\Big(\bigg(\Bigg($\quad
$\big\}\Big\}\bigg\}\Bigg\}$\quad
$\big\|\Big\|\bigg\|\Bigg\|$
```

$\Big((x+1) (x-1) \Big) ^{2}$
$\big(\Big(\bigg(\Bigg($ $\quad$ $\big\}\Big\}\bigg\}\Bigg\}$ $\quad$ $\big\|\Big\|\bigg\|\Bigg\|$

Има няколко команди за въвеждане на **три точки** във формула. `\ldots` отпечатва точките на основната линия, а `\cdots` ги центрира. Освен това, съществуват команди `\vdots` за вертикални и `\ddots` за диагонални точки. В Раздел 3.5 ще намерите друг пример.

```
\begin{displaymath}
x_{1},\ldots,x_{n}\qquad
x_{1}+\cdots+x_{n}
\end{displaymath}
```

$$x_{1},\ldots,x_{n}\qquad x_{1}+\cdots+x_{n}$$

3.4 Математически интервали

Ако избраните от TEX интервали вътре във формула не ви удовлетворяват, можете да ги промените чрез използване на специални команди за интервали. Има няколко команди за малки интервали: `\,` за $\frac{3}{18}$quad (⊔), `\:` за $\frac{4}{18}$quad (⊔) и `\;` за $\frac{5}{18}$quad (⊔). Екранният символ за интервал `\␣` дава среден размер интервал, а `\quad` (⊔) и `\qquad` (⊔) генерират по-големи интервали. Размерът `\quad` примерно съответства на ширината на буквата ‘M’ в текущия шрифт. Командата `\!` произвежда отрицателен интервал с размер $-\frac{3}{18}$ quad (⊔).

[1] Тези команди не работят както се очаква, ако се използват команди за смяна на размера на шрифта или ако е указана опция `11pt` или `12pt`. За коригиране на това поведение, използвайте пакетите exscale или amsmath.

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\int\!\!\!\int_{D} g(x,y)
  \, \ud x\, \ud y
\end{displaymath}
вместо
\begin{displaymath}
\int\int_{D} g(x,y)\ud x \ud y
\end{displaymath}
```

$$\int\!\!\!\int_{D} g(x,y)\,\mathrm{d}x\,\mathrm{d}y$$

вместо

$$\int\int_{D} g(x,y)\mathrm{d}x\mathrm{d}y$$

Забележете, че ‘d’ в диференциала обикновено се отпечатва в прав шрифт.

$\mathcal{AMS}$-LaTeX поддържа друг начин за тънка настройка на интервалите между няколко знака за интеграл: командите `\iint`, `\iiint`, `\iiiint` и `\idotsint`. Със зареждане на пакета `amsmath` горният пример може да бъде набран така:

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\iint_{D} \, \ud x \, \ud y
\end{displaymath}
```

$$\iint_{D} \, \mathrm{d}x \, \mathrm{d}y$$

За повече подробности вижте електронния документ textmath.tex (разпространява се с $\mathcal{AMS}$-LaTeX) или Глава 8 на *The LaTeX Companion* [3].

3.5 Вертикално разположен материал

За отпечатване на **матрици** се използва средата `array`. Тя работи подобно на средата `tabular`. Командата `\\` се използва за прекъсване на реда.

```
\begin{displaymath}
\mathbf{X} = \left(
\begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

$$\mathbf{X} = \left(
\begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)$$

Средата `array` може също да се използва за отпечатване на изрази, които имат един голям ляв ограничител и се поставя «`.`» като невидим десен ограничител:

```
\begin{displaymath}
y = \left\{
\begin{array}{ll}
 a & \textrm{ако $d>c$}\\
 b+x & \textrm{сутрин}\\
 l & \textrm{през другото време}
  \end{array} \right.
\end{displaymath}
```

$$y = \left\{
\begin{array}{ll}
 a & \textrm{ако $d>c$}\\
 b+x & \textrm{сутрин}\\
 l & \textrm{през другото време}
\end{array} \right.$$

Точно както и в средата `tabular`, можете да рисувате и линии в средата `array`, например, разделящи елементите на матрица:

```
\begin{displaymath}
\left(\begin{array}{c|c}
 1 & 2 \\
\hline 3 & 4
\end{array}\right)
\end{displaymath}
```

$$\left(\begin{array}{c|c} 1 & 2 \\ \hline 3 & 4 \end{array}\right)$$

За формули, заемащи няколко реда или за системи уравнения можете да използвате средите `eqnarray` и `eqnarray*` вместо `equation`. В `eqnarray` всеки ред получава отделен номер на уравнение. В `eqnarray*` номера́ не се поставят никъде.

Средите `eqnarray` и `eqnarray*` работят подобно на таблици с три стълба с формат `{rcl}`, където средният стълб се използва за знак за равенство, знак за неравенство, или друг подходящ знак. Командата `\\` прекъсва реда.

```
\begin{eqnarray}
f(x) & = & \cos x     \\
f'(x) & = & -\sin x   \\
\int_{0}^{x} f(y)dy &
 = & \sin x
\end{eqnarray}
```

$$f(x) = \cos x \tag{3.5}$$
$$f'(x) = -\sin x \tag{3.6}$$
$$\int_{0}^{x} f(y)dy = \sin x \tag{3.7}$$

Забележете, че от двете страни на средната колона, около знаците за равенство, има твърде много свободно място. То може да бъде редуцирано с установяването `\setlength\arraycolsep{2pt}`, както в следващия пример.

Дълги уравнения не се разбиват автоматично на равни части. Авторът трябва да посочи, къде да бъдат разбити и как да се подравнят. Най-често се използват следните два метода:

```
{\setlength\arraycolsep{2pt}
\begin{eqnarray}
\sin x & = & x -\frac{x^{3}}{3!}
     +\frac{x^{5}}{5!}-{}
                   \nonumber\\
 & & {}-\frac{x^{7}}{7!}+{}\cdots
\end{eqnarray}}
```

$$\begin{aligned} \sin x = x - \frac{x^{3}}{3!} + \frac{x^{5}}{5!} - {} \\ {} - \frac{x^{7}}{7!} + \cdots \end{aligned} \tag{3.8}$$

```
\begin{eqnarray}
\lefteqn{ \cos x = 1
     -\frac{x^{2}}{2!} +{} }
                   \nonumber\\
 & & {}+\frac{x^{4}}{4!}
     -\frac{x^{6}}{6!}+{}\cdots
\end{eqnarray}
```

$$\begin{aligned} &\cos x = 1 - \frac{x^{2}}{2!} + {} \\ &\qquad + \frac{x^{4}}{4!} - \frac{x^{6}}{6!} + \cdots \end{aligned} \tag{3.9}$$

Командата `\nonumber` заставя LaTeX да не генерира номер за това уравнение.

С тези методи може би е сложно да се получат добре изглеждащи вертикално подравнени уравнения; по-мощна алтернатива предоставя пакета amsmath (вж. средите `align`, `flalign`, `gather`, `multiline` и `split`).

3.6 Фантоми

Ние не можем да видим привидения, но те, все пак заемат своето място в умовете на някои хора. LaTeX позволява използване на фантоми (невидими обекти) за някои интересни пространствени трикове при разполагане на видимите обекти.

Когато LaTeX разполага текст по вертикала с помощта на командите `^` и `_`, той понякога проявява доста интелект. Чрез команда `\phantom` можете да запазите пространство за символи, които всъщност няма да се появят на страницата. По-добре това ще бъде разбрано от следните примери.

```
\begin{displaymath}
{}^{12}_{\phantom{1}6}\textrm{C}
\qquad \textrm{versus} \qquad
{}^{12}_{6}\textrm{C}
\end{displaymath}
```

$$ {}^{12}_{6}\textrm{C} \qquad \textrm{versus} \qquad {}^{12}_{6}\textrm{C} $$

```
\begin{displaymath}
\Gamma_{ij}^{\phantom{ij}k}
\qquad \textrm{versus} \qquad
\Gamma_{ij}^{k}
\end{displaymath}
```

$$ \Gamma_{ij}^{k} \qquad \textrm{versus} \qquad \Gamma_{ij}^{k} $$

3.7 Размер на математическия шрифт

В математически режим TeX избира размера на шрифта в зависимост от контекста. Индекси, например, се отпечатват с по-малък шрифт. Ако искате да добавите към уравнение обикновен текст, не използвайте командата `\textrm`, тъй като механизмът на превключване на размера няма да работи, защото `\textrm` временно влиза в текстов режим. За да работи този механизъм, използвайте командата `\mathrm`. Но имайте предвид, `\mathrm` ще работи добре само с къси елементи. Интервалите в `\mathrm` не са активни и акцентирани символи не работят.[1]

```
\begin{equation}
2^{\textrm{nd}} \quad
2^{\mathrm{nd}}
\end{equation}
```

$$ 2^{\textrm{nd}} \quad 2^{\mathrm{nd}} \tag{3.10} $$

[1] При включване на AMS-LaTeX (пакет amsmath) командата `\textrm` започва да работи с изменение на размера.

Всъщност, понякога се налага да укажете на LaTeX точния размер на шрифта. В математически режим, размерът се установява с четири команди:

`\displaystyle` ($\displaystyle 123$), `\textstyle` ($\textstyle 123$), `\scriptstyle` ($\scriptstyle 123$) и `\scriptscriptstyle` ($\scriptscriptstyle 123$).

Смяната на стиловете влияе също и на начина на изобразяване на границите.

```
\begin{displaymath}
\mathop{\mathrm{corr}}(X,Y)=
 \frac{\displaystyle
   \sum_{i=1}^n(x_i-\overline x)
   (y_i-\overline y)}
  {\displaystyle\biggl[
 \sum_{i=1}^n(x_i-\overline x)^2
\sum_{i=1}^n(y_i-\overline y)^2
\biggr]^{1/2}}
\end{displaymath}
```

$$\mathop{\mathrm{corr}}(X,Y)=\frac{\displaystyle\sum_{i=1}^n(x_i-\overline x)(y_i-\overline y)}{\displaystyle\biggl[\sum_{i=1}^n(x_i-\overline x)^2\sum_{i=1}^n(y_i-\overline y)^2\biggr]^{1/2}}$$

Това е един от примерите, когато са необходими по-големи скоби, отколкото предоставяните стандартни `\left[ \right]`. Използват се командите `\biggl` и `\biggr` съответно за лява и дясна скоби.

3.8 Теореми, закони, . . .

При писане на математически документи, вероятно ви е необходим способ за набиране на «леми», «определения», «аксиоми» и аналогични структури. LaTeX поддържа това с командите

`\newtheorem{`*име*`}[`*брояч*`]{`*текст*`}[`*раздел*`]`

Аргументът *име* е кратка ключова дума, използвана за идентификация на «теоремите». Аргументът *текст* определя истинското име на «теоремата», което ще се отпечатва в документа.

Аргументите в квадратни скоби са незадължителни. И двата се използват за определяне на това, как да се номерира «теоремата». С аргумента *брояч* може да се укаже *име* на по-рано обявена «теорема». Новата «теорема» тогава ще се номерира в същата последователност. Аргументът *раздел* позволява да определите раздел, вътре в който да се номерира вашата «теорема».

След като поставите в преамбюла на документа на всички команди `\newtheorem`, можете използвате следната команда вътре в текста:

> `\begin{`*име*`}[`*текст*`]`
> Това е интересна теорема.
> `\end{`*име*`}`

Тази теория трябва да е достатъчно. Следващите примери трябва да разсеят всички съмнения, и окончателно да ви убедят, че средата `\newtheorem` не е чак толкова сложна за разбиране.

```
% определения за
% преамбюла на документа
\newtheorem{law}{Law}
\newtheorem{jury}[law]{Jury}
% в тялото на документа
\begin{law} \label{law:box}
Don't hide in the witness box
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law~\ref{law:box}\end{jury}
\begin{law}No, No, No\end{law}
```

Law 1 *Don't hide in the witness box*

Jury 2 (The Twelve) *It could be you! So beware and see law 1*

Law 3 *No, No, No*

Теоремата «Jury» използва същия брояч, както и теоремата «Law». Следователно, тя получава пореден номер заедно с другите теореми «Law». Аргументът в квадратни скоби в текста, указва заглавията на теоремите, или нещо подобно.

```
\flushleft
\newtheorem{mur}{Murphy}[section]
\begin{mur}
Ако съществуват два или повече
начина да се направи нещо, и
един от тях може да доведе до
катастрофа, то някой непременно
ще направи точно това.
\end{mur}
```

Murphy 3.8.1 *Ако съществуват два или повече начина да се направи нещо, и един от тях може да доведе до катастрофа, то някой непременно ще направи точно това.*

Теоремата «Murphy» получава номер, свързан с номера на текущия раздел. Можете да използвате и друга структурна единица, например, глава или под-раздел.

3.9 Дебели (bold) символи

В LaTeX далеч не е просто да се получат дебели символи. Това, вероятно, е направено умишлено, защото е непрофесионално твърде често да се злоупотребява с тях. Командата за смяна на шрифта `\mathbf` дава дебели символи, но те са roman (прав) шрифт, докато математическите символи обикновено са наклонени. Съществува команда `\boldmath`, но *тя може да бъде използвана само извън математически режим*. Същото се отнася и за символите.

```
\begin{displaymath}
\mu, M \qquad \mathbf{M} \qquad
\mbox{\boldmath $\mu, M$}
\end{displaymath}
```

$$\mu, M \qquad \mathbf{M} \qquad \boldsymbol{\mu, M}$$

Забележете, че запетаята също е дебела, което може да е нежелателно.

Пакетът amsbsy (включван с пакета amsmath), както и пакетът bm (от инструментите tools), включват командата \boldsymbol.

```
\begin{displaymath}
\mu, M \qquad
\boldsymbol{\mu}, \boldsymbol{M}
\end{displaymath}
```

$$\mu, M \qquad \boldsymbol{\mu}, \boldsymbol{M}$$

3.10 Списък на математическите символи

В следните таблици ще намерите всички символи, достъпни обикновено в *математически режим*.

За достъп до символите, изброени в Таблици 3.12–3.16[1] в преамбюла на документа трябва да бъде зареден пакета amssymb и в системата трябва да бъдат установени математическите шрифтове AMS. Ако пакетът и шрифтовете AMS не са установени във вашата система, вижте на `CTAN:/tex-archive/macros/latex/required/amslatex`. Още по-пълен списък на символите можете да намерите на адрес `CTAN:/tex-archive/info/symbols/comprehensive`.

Таблица 3.1: Акценти в математически режим

$\hat{a}$	`\hat{a}`	$\check{a}$	`\check{a}`	$\tilde{a}$	`\tilde{a}`	$\acute{a}$	`\acute{a}`
$\grave{a}$	`\grave{a}`	$\dot{a}$	`\dot{a}`	$\ddot{a}$	`\ddot{a}`	$\breve{a}$	`\breve{a}`
$\bar{a}$	`\bar{a}`	$\vec{a}$	`\vec{a}`	$\widehat{A}$	`\widehat{A}`	$\widetilde{A}$	`\widetilde{A}`

Таблица 3.2: Малки гръцки букви

α	`\alpha`	θ	`\theta`	o	`o`	υ	`\upsilon`
β	`\beta`	ϑ	`\vartheta`	π	`\pi`	ϕ	`\phi`
γ	`\gamma`	ι	`\iota`	ϖ	`\varpi`	φ	`\varphi`
δ	`\delta`	κ	`\kappa`	ρ	`\rho`	χ	`\chi`
ϵ	`\epsilon`	λ	`\lambda`	ϱ	`\varrho`	ψ	`\psi`
ε	`\varepsilon`	μ	`\mu`	σ	`\sigma`	ω	`\omega`
ζ	`\zeta`	ν	`\nu`	ς	`\varsigma`		
η	`\eta`	ξ	`\xi`	τ	`\tau`		

Таблица 3.3: Главни гръцки букви

Γ	`\Gamma`	Λ	`\Lambda`	Σ	`\Sigma`	Ψ	`\Psi`
Δ	`\Delta`	Ξ	`\Xi`	Υ	`\Upsilon`	Ω	`\Omega`
Θ	`\Theta`	Π	`\Pi`	Φ	`\Phi`		

[1] Тези таблици са били получени от `symbols.tex` (автор David Carlisle) и по-късно силно изменени по съвет на Josef Tkadlec

Таблица 3.4: Бинарни отношения

Можете да получите съответните отрицания чрез добавяне пред следните символи на командата `\not`.

$<$	`<`	$>$	`>`	$=$	`=`
$\leq$	`\leq` или `\le`	$\geq$	`\geq` или `\ge`	$\equiv$	`\equiv`
$\ll$	`\ll`	$\gg$	`\gg`	$\doteq$	`\doteq`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubset$	`\sqsubset` [1]	$\sqsupset$	`\sqsupset` [1]	$\Join$	`\Join` [1]
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni` , `\owns`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\perp$	`\perp`
$\smile$	`\smile`	$\frown$	`\frown`	$\asymp$	`\asymp`
$:$	`:`	$\notin$	`\notin`	$\neq$	`\neq` или`\ne`

[1]За достъп до този символ ползвайте пакета latexsym.

Таблица 3.5: Бинарни оператори

$+$	`+`	$-$	`-`		
$\pm$	`\pm`	$\mp$	`\mp`	$\triangleleft$	`\triangleleft`
$\cdot$	`\cdot`	$\div$	`\div`	$\triangleright$	`\triangleright`
$\times$	`\times`	$\setminus$	`\setminus`	$\star$	`\star`
$\cup$	`\cup`	$\cap$	`\cap`	$\ast$	`\ast`
$\sqcup$	`\sqcup`	$\sqcap$	`\sqcap`	$\circ$	`\circ`
$\vee$	`\vee` , `\lor`	$\wedge$	`\wedge` , `\land`	$\bullet$	`\bullet`
$\oplus$	`\oplus`	$\ominus$	`\ominus`	$\diamond$	`\diamond`
$\odot$	`\odot`	$\oslash$	`\oslash`	$\uplus$	`\uplus`
$\otimes$	`\otimes`	$\bigcirc$	`\bigcirc`	$\amalg$	`\amalg`
$\bigtriangleup$	`\bigtriangleup`	$\bigtriangledown$	`\bigtriangledown`	$\dagger$	`\dagger`
$\lhd$	`\lhd` [1]	$\rhd$	`\rhd` [1]	$\ddagger$	`\ddagger`
$\unlhd$	`\unlhd` [1]	$\unrhd$	`\unrhd` [1]	$\wr$	`\wr`

Таблица 3.6: Големи оператори

$\sum$	`\sum`	$\bigcup$	`\bigcup`	$\bigvee$	`\bigvee`	$\bigoplus$	`\bigoplus`
$\prod$	`\prod`	$\bigcap$	`\bigcap`	$\bigwedge$	`\bigwedge`	$\bigotimes$	`\bigotimes`
$\coprod$	`\coprod`	$\bigsqcup$	`\bigsqcup`			$\bigodot$	`\bigodot`
$\int$	`\int`	$\oint$	`\oint`			$\biguplus$	`\biguplus`

Таблица 3.7: Стрелки

$\leftarrow$	`\leftarrow` или `\gets`	$\longleftarrow$	`\longleftarrow`	$\uparrow$	`\uparrow`
$\rightarrow$	`\rightarrow` или `\to`	$\longrightarrow$	`\longrightarrow`	$\downarrow$	`\downarrow`
$\leftrightarrow$	`\leftrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`	$\updownarrow$	`\updownarrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`	$\Uparrow$	`\Uparrow`
$\Rightarrow$	`\Rightarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Downarrow$	`\Downarrow`
$\Leftrightarrow$	`\Leftrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`	$\Updownarrow$	`\Updownarrow`
$\mapsto$	`\mapsto`	$\longmapsto$	`\longmapsto`	$\nearrow$	`\nearrow`
$\hookleftarrow$	`\hookleftarrow`	$\hookrightarrow$	`\hookrightarrow`	$\searrow$	`\searrow`
$\leftharpoonup$	`\leftharpoonup`	$\rightharpoonup$	`\rightharpoonup`	$\swarrow$	`\swarrow`
$\leftharpoondown$	`\leftharpoondown`	$\rightharpoondown$	`\rightharpoondown`	$\nwarrow$	`\nwarrow`
$\rightleftharpoons$	`\rightleftharpoons`	$\iff$	`\iff` (голям интервал)	$\leadsto$	`\leadsto` [1]

[1]За достъп до този символ ползвайте пакета latexsym.

Таблица 3.8: Ограничители

<table>
<tr><td>$($</td><td><code>(</code></td><td>$)$</td><td><code>)</code></td><td>$\uparrow$</td><td><code>\uparrow</code></td><td>$\Uparrow$</td><td><code>\Uparrow</code></td></tr>
<tr><td>$[$</td><td><code>[</code> или <code>\lbrack</code></td><td>$]$</td><td><code>]</code> или <code>\rbrack</code></td><td>$\downarrow$</td><td><code>\downarrow</code></td><td>$\Downarrow$</td><td><code>\Downarrow</code></td></tr>
<tr><td>$\{$</td><td><code>\{</code> или <code>\lbrace</code></td><td>$\}$</td><td><code>\}</code> или <code>\rbrace</code></td><td>$\updownarrow$</td><td><code>\updownarrow</code></td><td>$\Updownarrow$</td><td><code>\Updownarrow</code></td></tr>
<tr><td>$\langle$</td><td><code>\langle</code></td><td>$\rangle$</td><td><code>\rangle</code></td><td>$\vert$</td><td><code>|</code> или <code>\vert</code></td><td>$\Vert$</td><td><code>\|</code> или <code>\Vert</code></td></tr>
<tr><td>$\lfloor$</td><td><code>\lfloor</code></td><td>$\rfloor$</td><td><code>\rfloor</code></td><td>$\lceil$</td><td><code>\lceil</code></td><td>$\rceil$</td><td><code>\rceil</code></td></tr>
<tr><td>$/$</td><td><code>/</code></td><td>$\backslash$</td><td><code>\backslash</code></td><td></td><td></td><td></td><td></td></tr>
</table>

Таблица 3.9: Големи ограничители

$\lgroup$	`\lgroup`	$\rgroup$	`\rgroup`	$\lmoustache$	`\lmoustache`	$\rmoustache$	`\rmoustache`
$\arrowvert$	`\arrowvert`	$\Arrowvert$	`\Arrowvert`	$\bracevert$	`\bracevert`		

Таблица 3.10: Други символи

$\dots$	\dots	$\cdots$	\cdots	$\vdots$	\vdots	$\ddots$	\ddots
$\hbar$	\hbar	$\imath$	\imath	$\jmath$	\jmath	ℓ	\ell
$\Re$	\Re	$\Im$	\Im	$\aleph$	\aleph	$\wp$	\wp
$\forall$	\forall	$\exists$	\exists	$\mho$	\mho [1]	∂	\partial
$'$	'	$\prime$	\prime	$\emptyset$	\emptyset	∞	\infty
∇	\nabla	$\triangle$	\triangle	$\Box$	\Box [1]	$\Diamond$	\Diamond [1]
$\bot$	\bot	$\top$	\top	$\angle$	\angle	$\surd$	\surd
$\diamondsuit$	\diamondsuit	$\heartsuit$	\heartsuit	$\clubsuit$	\clubsuit	$\spadesuit$	\spadesuit
$\neg$	\neg или \lnot	$\flat$	\flat	$\natural$	\natural	$\sharp$	\sharp

[1] За достъп до този символ ползвайте пакета latexsym.

Таблица 3.11: Не-математически символи

Тези символи могат да се използват и в текстов режим.

†	\dag	§	\S	©	\copyright	®	\textregistered
‡	\ddag	¶	\P	£	\pounds	%	\%

Таблица 3.12: Ограничители AMS

$\ulcorner$	\ulcorner	$\urcorner$	\urcorner	$\llcorner$	\llcorner	$\lrcorner$	\lrcorner
$\lvert$	\lvert	$\rvert$	\rvert	$\lVert$	\lVert	$\rVert$	\rVert

Таблица 3.13: Букви от гръцки и иврит AMS

$\digamma$	\digamma	$\varkappa$	\varkappa	$\beth$	\beth	$\gimel$	\gimel	$\daleth$	\daleth

Таблица 3.14: Бинарни отношения AMS

$\lessdot$	\lessdot	$\gtrdot$	\gtrdot	$\doteqdot$	\doteqdot или \Doteq
$\leqslant$	\leqslant	$\geqslant$	\geqslant	$\risingdotseq$	\risingdotseq
$\eqslantless$	\eqslantless	$\eqslantgtr$	\eqslantgtr	$\fallingdotseq$	\fallingdotseq
$\leqq$	\leqq	$\geqq$	\geqq	$\eqcirc$	\eqcirc
$\lll$	\lll или \llless	$\ggg$	\ggg или \gggtr	$\circeq$	\circeq
$\lesssim$	\lesssim	$\gtrsim$	\gtrsim	$\triangleq$	\triangleq
$\lessapprox$	\lessapprox	$\gtrapprox$	\gtrapprox	$\bumpeq$	\bumpeq
$\lessgtr$	\lessgtr	$\gtrless$	\gtrless	$\Bumpeq$	\Bumpeq
$\lesseqgtr$	\lesseqgtr	$\gtreqless$	\gtreqless	$\thicksim$	\thicksim
$\lesseqqgtr$	\lesseqqgtr	$\gtreqqless$	\gtreqqless	$\thickapprox$	\thickapprox
$\preccurlyeq$	\preccurlyeq	$\succcurlyeq$	\succcurlyeq	$\approxeq$	\approxeq
$\curlyeqprec$	\curlyeqprec	$\curlyeqsucc$	\curlyeqsucc	$\backsim$	\backsim
$\precsim$	\precsim	$\succsim$	\succsim	$\backsimeq$	\backsimeq
$\precapprox$	\precapprox	$\succapprox$	\succapprox	$\vDash$	\vDash
$\subseteqq$	\subseteqq	$\supseteqq$	\supseteqq	$\Vdash$	\Vdash
$\Subset$	\Subset	$\Supset$	\Supset	$\Vvdash$	\Vvdash
$\sqsubset$	\sqsubset	$\sqsupset$	\sqsupset	$\backepsilon$	\backepsilon
$\therefore$	\therefore	$\because$	\because	$\varpropto$	\varpropto
$\shortmid$	\shortmid	$\shortparallel$	\shortparallel	$\between$	\between
$\smallsmile$	\smallsmile	$\smallfrown$	\smallfrown	$\pitchfork$	\pitchfork
$\vartriangleleft$	\vartriangleleft	$\vartriangleright$	\vartriangleright	$\blacktriangleleft$	\blacktriangleleft
$\trianglelefteq$	\trianglelefteq	$\trianglerighteq$	\trianglerighteq	$\blacktriangleright$	\blacktriangleright

Таблица 3.15: Стрелки AMS

$\dashleftarrow$	\dashleftarrow	$\dashrightarrow$	\dashrightarrow	$\multimap$	\multimap
$\leftleftarrows$	\leftleftarrows	$\rightrightarrows$	\rightrightarrows	$\upuparrows$	\upuparrows
$\leftrightarrows$	\leftrightarrows	$\rightleftarrows$	\rightleftarrows	$\downdownarrows$	\downdownarrows
$\Lleftarrow$	\Lleftarrow	$\Rrightarrow$	\Rrightarrow	$\upharpoonleft$	\upharpoonleft
$\twoheadleftarrow$	\twoheadleftarrow	$\twoheadrightarrow$	\twoheadrightarrow	$\upharpoonright$	\upharpoonright
$\leftarrowtail$	\leftarrowtail	$\rightarrowtail$	\rightarrowtail	$\downharpoonleft$	\downharpoonleft
$\leftrightharpoons$	\leftrightharpoons	$\rightleftharpoons$	\rightleftharpoons	$\downharpoonright$	\downharpoonright
$\Lsh$	\Lsh	$\Rsh$	\Rsh	$\rightsquigarrow$	\rightsquigarrow
$\looparrowleft$	\looparrowleft	$\looparrowright$	\looparrowright	$\leftrightsquigarrow$	\leftrightsquigarrow
$\curvearrowleft$	\curvearrowleft	$\curvearrowright$	\curvearrowright		
$\circlearrowleft$	\circlearrowleft	$\circlearrowright$	\circlearrowright		

Таблица 3.16: Отрицателни бинарни отношения и стрелки AMS

$\nless$	`\nless`	$\ngtr$	`\ngtr`	$\varsubsetneqq$	`\varsubsetneqq`
$\lneq$	`\lneq`	$\gneq$	`\gneq`	$\varsupsetneqq$	`\varsupsetneqq`
$\nleq$	`\nleq`	$\ngeq$	`\ngeq`	$\nsubseteqq$	`\nsubseteqq`
$\nleqslant$	`\nleqslant`	$\ngeqslant$	`\ngeqslant`	$\nsupseteqq$	`\nsupseteqq`
$\lneqq$	`\lneqq`	$\gneqq$	`\gneqq`	$\nmid$	`\nmid`
$\lvertneqq$	`\lvertneqq`	$\gvertneqq$	`\gvertneqq`	$\nparallel$	`\nparallel`
$\nleqq$	`\nleqq`	$\ngeqq$	`\ngeqq`	$\nshortmid$	`\nshortmid`
$\lnsim$	`\lnsim`	$\gnsim$	`\gnsim`	$\nshortparallel$	`\nshortparallel`
$\lnapprox$	`\lnapprox`	$\gnapprox$	`\gnapprox`	$\nsim$	`\nsim`
$\nprec$	`\nprec`	$\nsucc$	`\nsucc`	$\ncong$	`\ncong`
$\npreceq$	`\npreceq`	$\nsucceq$	`\nsucceq`	$\nvdash$	`\nvdash`
$\precneqq$	`\precneqq`	$\succneqq$	`\succneqq`	$\nvDash$	`\nvDash`
$\precnsim$	`\precnsim`	$\succnsim$	`\succnsim`	$\nVdash$	`\nVdash`
$\precnapprox$	`\precnapprox`	$\succnapprox$	`\succnapprox`	$\nVDash$	`\nVDash`
$\subsetneq$	`\subsetneq`	$\supsetneq$	`\supsetneq`	$\ntriangleleft$	`\ntriangleleft`
$\varsubsetneq$	`\varsubsetneq`	$\varsupsetneq$	`\varsupsetneq`	$\ntriangleright$	`\ntriangleright`
$\nsubseteq$	`\nsubseteq`	$\nsupseteq$	`\nsupseteq`	$\ntrianglelefteq$	`\ntrianglelefteq`
$\subsetneqq$	`\subsetneqq`	$\supsetneqq$	`\supsetneqq`	$\ntrianglerighteq$	`\ntrianglerighteq`
$\nleftarrow$	`\nleftarrow`	$\nrightarrow$	`\nrightarrow`	$\nleftrightarrow$	`\nleftrightarrow`
$\nLeftarrow$	`\nLeftarrow`	$\nRightarrow$	`\nRightarrow`	$\nLeftrightarrow$	`\nLeftrightarrow`

Таблица 3.17: Бинарни оператори AMS

$\dotplus$	`\dotplus`	$\centerdot$	`\centerdot`	$\intercal$	`\intercal`
$\ltimes$	`\ltimes`	$\rtimes$	`\rtimes`	$\divideontimes$	`\divideontimes`
$\Cup$	`\Cup` или `\doublecup`	$\Cap$	`\Cap` или `\doublecap`	$\smallsetminus$	`\smallsetminus`
$\veebar$	`\veebar`	$\barwedge$	`\barwedge`	$\doublebarwedge$	`\doublebarwedge`
$\boxplus$	`\boxplus`	$\boxminus$	`\boxminus`	$\circleddash$	`\circleddash`
$\boxtimes$	`\boxtimes`	$\boxdot$	`\boxdot`	$\circledcirc$	`\circledcirc`
$\leftthreetimes$	`\leftthreetimes`	$\rightthreetimes$	`\rightthreetimes`	$\circledast$	`\circledast`
$\curlyvee$	`\curlyvee`	$\curlywedge$	`\curlywedge`		

Таблица 3.18: Други символи AMS

$\hbar$	`\hbar`	$\hslash$	`\hslash`	$\Bbbk$	`\Bbbk`
$\square$	`\square`	$\blacksquare$	`\blacksquare`	$\circledS$	`\circledS`
$\vartriangle$	`\vartriangle`	$\blacktriangle$	`\blacktriangle`	$\complement$	`\complement`
$\triangledown$	`\triangledown`	$\blacktriangledown$	`\blacktriangledown`	$\Game$	`\Game`
$\lozenge$	`\lozenge`	$\blacklozenge$	`\blacklozenge`	$\bigstar$	`\bigstar`
$\angle$	`\angle`	$\measuredangle$	`\measuredangle`	$\sphericalangle$	`\sphericalangle`
$\diagup$	`\diagup`	$\diagdown$	`\diagdown`	$\backprime$	`\backprime`
$\nexists$	`\nexists`	$\Finv$	`\Finv`	$\varnothing$	`\varnothing`
$\eth$	`\eth`	$\mho$	`\mho`		

Таблица 3.19: Математически азбуки

Пример	Команда	Изискван пакет
ABCdef	`\mathrm{ABCdef}`	
ABCdef	`\mathit{ABCdef}`	
$\mathnormal{ABCdef}$	`\mathnormal{ABCdef}`	
$\mathcal{ABC}$	`\mathcal{ABC}`	euscript с опция: mathcal
$\mathscr{ABC}$	`\mathscr{ABC}`	mathrsfs
$\mathfrak{ABCdef}$	`\mathfrak{ABCdef}`	eufrak
$\mathbb{ABC}$	`\mathbb{ABC}`	amsfonts или amssymb

4 Специални възможности

При работа с голям документ, LaTeX ще ви помогне с някои особености, като генерация на азбучен указател, управление на библиография и други подобни. По-подробно описание на специалните възможности и разширения на LaTeX има в *LaTeX Manual* [1] и в *The LaTeX Companion* [3].

4.1 Включване на Encapsulated PostScript графики

LaTeX дава основните средства за работа с плаващи обекти, такива, като фигури и таблици, с помощта на средите `figure` и `table`.

Съществуват също няколко способа за създаване на собствена графика със средствата на базовия LaTeX или негови разширения. За съжаление, повечето потребители ги определят като трудни за разбиране, затова тук тези способи не се разглеждат. Допълнителна информация е дадена в *The LaTeX Companion* [3] и в *LaTeX Manual* [1].

Много по-лесен начин за получаване на графики в документа е те да се създадат със специализирани програмни пакети[1] и да се включват в документа готовите графики. Също и тук, LaTeX-пакетите предлагат множество начини да се прави това, но в това въведение се обсъжда само използване на графики във формат Encapsulated PostScript (EPS), тъй като това се прави лесно и е широко разпространено. За да се използват картинки във формат EPS, трябва да има на разположение PostScript принтер за печат.[2]

Добър набор команди за включване на графики има в пакета graphicx (автор D. P. Carlisle). Той е част от цяло семейство пакети, наречени комплект “graphics”.[3]

При предположение, че работите в система с достъпен за извеждане PostScript-принтер и с инсталиран пакет graphicx, можете да използвате следната постъпкова инструкция за включване на картинка във вашия документ:

1. Експортирайте картинката от вашата графична програма във формат EPS.[4]

---

[1] Такива, като XFig, CorelDraw!, Freehand, Gnuplot, ...

[2] Друга възможност за извеждане на PostScript е чрез използване на програмата GhostScript, достъпна от `CTAN:/tex-archive/support/ghostscript`. Потребителите на Windows и OS/2 могат да обърнат внимание на програмата GSview.

[3] `CTAN:/tex-archive/macros/latex/required/graphics`

[4] Ако вашата програма няма експорт в EPS, може да опитате да установите драйвер на PostScript-принтера (например, някакъв Apple LaserWriter) и да печатате във файл чрез този драйвер. Ако ви провърви, в този файл ще се получи EPS. Забележете, че EPS трябва да съдържа не повече от една страница. Някои принтерни драйвери трябва в явен вид да се настройват за генерация на EPS.

2. В преамбюла на документа заредете пакета graphicx с командата

   `\usepackage[`*драйвер*`]{graphicx}`

   където *драйвер* е името на вашата конвертираща програма от DVI в PostScript. Най-широко използвания конвертор се нарича `dvips`. Името на драйвера се изисква затова, защото не съществува стандарт за включване на графика в TeX. Знаейки името на *драйвера*, graphicx може да избере точния метод за включване на информация за графиката в `.dvi` файла така, че принтерът да я разбере и да успее коректно да включи `.eps` файла.

3. Използвайте командата

   `\includegraphics[`*опция*=*стойност*, ...`]{`*файл*`}`

   за включване на *файл* във вашия документ. Незадължителният параметър допуска списък на разделени със запетаи *опции* и съответните им *стойности*. *Опциите* могат да се използват за изменение на ширината, височината или ъгъла на завъртане на включваната графика. Таблица 4.1 изброява най-важните опции.

Таблица 4.1: Имена на опциите на пакета graphicx

`width`	мащабира графиката до указаната ширина
`height`	мащабира графиката до указаната височина
`angle`	завърта графиката обратно на часовата стрелка
`scale`	мащабира графиката

Следният пример може да направи нещата по-ясни:

```
\begin{figure}
\begin{center}
\includegraphics[angle=90,width=0.5\textwidth]{test}
\end{center}
\end{figure}
```

Тук се включва графика, записана във файл `test.eps`. Тя *отначало* се завърта на 90 градуса и *след това* се мащабира до крайна ширина 0.5 от ширината на стандартния параграф. Пропорциите се съхраняват, защото не е указана конкретна височина. Параметрите височина и ширина могат също да бъдат указани в абсолютни размерности. Вижте Таблица 6.5 на страница 109 за повече информация. Ако искате да знаете за това повече, прочетете [9] и [13].

4.2 Библиография

Библиография се генерира в средата `thebibliography`. Всеки елемент започва с

`\bibitem[`*етикет*`]{`*маркер*`}`

След това *маркер* се използва за цитиране на книгата, статията или труда в документа.

`\cite{`*маркер*`}`

Ако не използвате опцията *етикет*, елементите на библиографията се номерират автоматично. Параметърът след командата `\begin{thebibliography}` определя колко място да се резервира за номера́та на етикетите. В следващия пример `{99}` указва на LaTeX, че нито един от номера́та на етикетите няма да е по-широк от числото '99'.

```
Partl~\cite{pa} е
предложил \ldots

\begin{thebibliography}{99}
\bibitem{pa} H.~Partl:
\emph{German \TeX},
TUGboat Volume~9, Issue~1 (1988)
\end{thebibliography}
```

Partl [1] е предложил ...

Библиография

[1] H. Partl: *German TEX*, TUGboat Volume 9, Issue 1 (1988)

За големи проекти можете да изберете програмата BibTEX. BibTEX се включва в повечето дистрибутиви на TEX. Тя позволява да поддържате библиографска база данни и да извличате от нея елементи, свързани с това, което цитирате във вашия документ. Визуалното представяне на библиографията, генерирана с BibTEX, е основано на концепцията на стиловете, което ви позволява да създавате библиография, следвайки широк асортимент от установени стилове на библиографии.

4.3 Указатели

Полезна особеност на много книги е техният азбучен указател. С помощта на LaTeX и съпровождащата програма `makeindex`, указателят може да се създава твърде лесно.[1]

[1] На системи, неподдържащи имена на файлове над 8 символа, програмата може да се нарича `makeidx`.

Таблица 4.2: Примерен синтаксис на ключове за указателя

Пример	Вид на указателя	Коментар
`\index{hello}`	hello, 1	обикновен елемент
`\index{hello!Peter}`	Peter, 3	Подчинен на 'hello' елемент
`\index{Sam@\textsl{Sam}}`	*Sam*, 2	форматиран ключ
`\index{Lin@\textbf{Lin}}`	**Lin**, 7	същото като горното
`\index{Jenny|textbf}`	Jenny, **3**	форматирана страница
`\index{Joe|textit}`	Joe, *5*	същото като горното
`\index{eolienne@\'eolienne}`	éolienne, 4	акцентирани букви

В това въведение са изложени само базовите команди за генерация на указателя. По-задълбочено изложение вижте в *The LaTeX Companion* [3].

За включване на възможностите на LaTeX, в преамбюла на документа трябва да се зареди пакетът makeidx:

```
\usepackage{makeidx}
```

и трябва да бъдат разрешени специалните команди на указателя чрез поставяне в преамбюла на командата

```
\makeindex
```

Съдържанието на указателя се определя с командата

```
\index{ключ}
```

където *ключ* е елемент на указателя. Команди на указателя се въвеждат на онези места в текста, където трябва да сочи съответният елемент на указателя. Таблица 4.2 обяснява синтаксиса на аргумента *ключ* с няколко примера.

При обработката на входния файл с LaTeX, всяка команда `\index` записва в специален файл съответния елемент на указателя, заедно с номера на текущата страница. Файлът има същото име като входния LaTeX-файл, но друго разширение (`.idx`). Този `.idx`-файл след това се обработва с програмата `makeindex`.

```
makeindex filename
```

Програмата `makeindex` генерира сортиран указател със същото име, но този път – с разширение `.ind`.

Ако сега отново се компилира входния файл, този сортиран указател се включва в документа на това място, където LaTeX намира командата

```
\printindex
```

Пакетът showidx, влизащ в LaTeX 2_ε, печата всички елементи на указателя в лявото поле на текста. Това е особено полезно при проверка на текста и сверяване с указателя.

Да отбележим, че командата `\index`, ако се използва невнимателно, може да повлияе на вида на отпечатване.

```
Моята дума \index{дума}. За разлика
от дума\index{дума}. Забележете
положението на точката.
```

Моята дума . За разлика от дума. Забележете положението на точката.

4.4 Настройка на колонтитули

Пакетът fancyhdr,[1] написан от Piet van Oostrum, предоставя няколко прости команди, които позволяват да настройвате горния и долния колонтитули на документа. Ако сега погледнете горе на тази страница, ще видите едно от възможните приложения на този пакет.

Сложността в настройката на колонтитулите е в това, че се включват неща, касаещи заглавията на разделите или главите. LaTeX реализира това чрез двуетапен подход. При дефинирането на колонтитулите се използват командите `\rightmark` и `\leftmark` за определяне съответно на заглавията на текущите Раздел и Глава. Стойностите на тези две команди се изменят при обработка с командите `\chapter` или `\section`.

За по-голяма гъвкавост, командата `\chapter` и нейните подобни не предефинират `\rightmark` и `\leftmark` сами, а извикват друга команда, наречена `\chaptermark`, `\sectionmark` или `\subsectionmark`, отговаряща за преопределяне на `\rightmark` и `\leftmark`.

Така че, ако искате да измените вида на названията на главите в горния колонтитул, вие просто преопределяте командата `\chaptermark`.

Фигура 4.1 показва, как може да се настрои пакета fancyhdr така, че колонтитулите да изглеждат почти така, както изглеждат в този буклет. Във всеки случай, препоръчвам ви да се запознаете с документацията към пакета на адреса, отбелязан в забележката под поле.

[1] Достъпен от
`CTAN:/tex-archive/macros/latex/contrib/supported/fancyhdr`.

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
% С това сме сигурни, че заглавията на  глави и
% раздели са в долен регистър.
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{}  % изтрива текущите установки за колонтитулите
\fancyhead[LE,RO]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\addtolength{\headheight}{0.5pt} % оставя място за линия
\fancypagestyle{plain}{%
   \fancyhead{} % изчиства колонтитулите на обикновените страници
   \renewcommand{\headrulewidth}{0pt} % и линията
}
```

Фигура 4.1: Пример за настройка на fancyhdr

4.5 Пакет verbatim

По-горе в тази книга се запознахте със *средата* `verbatim`. В този раздел ще узнаете за *пакета* verbatim. Пакетът verbatim представлява повторна реализация на средата `verbatim` с поправки на някои негови ограничения. Само по себе си това не е забележително,[1] но освен това, в него е добавена и известна функционалност, поради което тук споменавам този пакет. Пакетът verbatim предоставя командата

`\verbatiminput{`*файл*`}`

която позволява да се включва текстов файл в документа, така, както ако неговото съдържание би се намирало вътре в средата `verbatim`.

Тъй като пакетът verbatim е част от комплекта ‘tools’, вие трябва да го намерите инсталиран на повечето системи. Ако искате да знаете за този пакет повече, обезателно прочетете [10].

[1] За американците това не е забележително. За нас е интересно това, че пакетът verbatim включва командата `\verbatim@font`, позволяюваща да се използва произволен шрифт, например, кирилица, което е невъзможно в средата `verbatim` без модификация на стандартното поведение на LaTeX. — *Бел. ред.*

4.6 Изтегляне и инсталиране на LaTeX-пакети

Повечето дистрибутиви на LaTeX включват голям набор установени стилеви пакети, но много повече са достъпни в мрежата. Основно място за търсене на стилеви пакети в Интернет е CTAN (`http://www.ctan.org/`).

Пакети като geometry, hyphenat и много други, обикновено се състоят от два файла: файл с разширение `.ins` и друг – с разширение `.dtx`. Често към тях се прилага файл `readme.txt` с кратко описание на пакета. Разбира се, първо трябва да се прочете този файл.

Във всеки случай, веднага след като сте копирали файловете от пакета на вашия компютър, трябва да ги обработите така, че (а) вашият TeX да узнае за наличието на нов пакет и (б) да получите необходимата документация. Ето как се изпълнява първата част:

1. Обработете с LaTeX файла `.ins`. Като резултат ще получите файл `.sty`.
2. Преместете файла `.sty` там, където вашият дистрибутив търси тези файлове. Обикновено това са под-каталози на каталога `.../`*`localtexmf`*`/tex/latex`. (Потребителите на Windows и OS/2 трябва да попълнят липсващото в директорията).
3. Обновете базата с имената на файловете във вашия дистрибутив. Командата зависи от използвания LaTeX-дистрибутив: за teTeX, fpTeX – `texhash`, за web2c – `maktexlsr`, за MikTeX – `initexmf-update-fndb` или с използване на графичния интерфейс (GUI).

Сега можете да получите документацията от файла `.dtx`:

1. Обработете с LaTeX файла `.dtx`. Като резултат ще получите файл `.dvi`. Забележете, че може да се наложи да пуснете LaTeX няколко пъти, за да получите правилни препратки.
2. Проверете, дали LaTeX е генерирал файл `.idx`, освен всички други получени файлове. Ако не видите такъв файл, преминете на точка 5.
3. За да генерирате азбучния указател, напишете командата:
   `makeindex -s gind.ist` *име*
   (където *име* е името на основния файл без разширение).
4. Отново обработете с LaTeX файла `.dtx`
5. Накрая, за удобство при четене, генерирайте файла `.ps` или `.pdf`.

Ще забележите понякога, че се е генерирал и файл `.glo` (глосарий). В този случай между стъпките 4 и 5 изпълнете командата:
`makeindex -s gglo.ist -o` *име*`.gls` *име*`.glo`
и поне още веднъж компилирайте с LaTeX файла `.dtx` преди преминаване на стъпка 5.

4.7 Работа с pdfLaTeX

Автор: Daniel Flipo <Daniel.Flipo@univ-lille1.fr>

PDF е формат на хипертекстови документи. Така, както и на web страниците, някои думи в документа се означават като хипервръзки. Те водят към други точки в документа или дори към други документи. Ако кли́кнете с мишката върху такава хипервръзка, ще попаднете в точката-цел на връзката. В контекста на LaTeX това означава, че всички срещнати в текста команди `\ref` и `\pageref` стават хипервръзки. Освен това, съдържанието, азбучният указател и други аналогични структури стават набор от елементи на хипервръзки.

Голяма част от web-страниците днес се пишат на езика HTML *(HyperText Markup Language)*. За научни документи този формат има два сериозни недостъка:

1. Включването в HTML на математически формули, най-общо казано, не се поддържа. Независимо от наличието на такъв стандарт, повечето използвани днес Интернет - навигатори не го поддържат или нямат необходимите шрифтове.

2. Печатане на HTML документи е възможен, но резултатът силно зависи от използваната платформа и навигатора. Резултатът дори малко не прилича на това качество, което сме свикнали да очакваме в света на LaTeX.

Имало е много опити да се създадат транслатори от LaTeX в HTML. Някои от тях са били дори твърде успешни, в този смисъл, че те могат да генерират коректни web - страници от обикновени файлове на LaTeX. Но всички те правят множество опростявания, за да получат резултата. Веднага щом започнете да използвате сложни възможности на LaTeX и външни пакети, нещата започват да се разпадат и транслаторът дава грешки. Авторите, желаещи да съхранят уникалното типографско качество на своите документи, публикувани дори на WWW, се обръщат към PDF *(Portable Document Format)*, който съхранява вида на документа и позволява да се използва хипертекстова навигация. Повечето съвременни web - навигатори имат вградени средства за изобразяване на документи във формат PDF.

Въпреки че има средства за преглеждане на формат DVI и PostScript почти на всяка платформа, много по-разпространени са Acrobat Reader и xpdf за преглеждане на PDF-документи. Затова, предоставяйки PDF-версии на вашите документи, вие ги правите много по-достъпни за потенциалните читатели.

4.7.1 PDF-документи за WWW

Създаването на PDF-файл от изходния TeX-файл, е много просто, благодарение на програмата pdfTeX, разработена от Hàn Th´ê Thành. pdfTeX генерира PDF по аналогичен начин, както TeX генерира DVI. Съществува също pdfLaTeX, генериращ PDF от изходния LaTeX-текст.

И двете програми, pdfTeX и pdfLaTeX, се инсталират автоматично при повечето съвременни дистрибуции на TeX, като teTeX, fpTeX, MikTeX, TeXLive и CMacTeX.

За генерация на PDF вместо DVI е достатъчно вместо командата `latex file.tex` да се използва командата `pdflatex file.tex`. Там, където LaTeX не се стартира от командния ред, можете да намерите специален бутон в TeXControlCenter.

В LaTeX можете да определите размера на хартията с добавяне на незадължителен параметър в командата `\documentclass`, например, `a4paper` или `letterpaper`. Този механизъм работи и в pdfLaTeX, но освен това, pdfTeX се нуждае и от физическия размер на хартията, за да определи физическия размер на страницата в pdf-файла. Ако използвате пакета hyperref (вж. стр. 77), размерът на хартията ще бъде установен автоматично. В други случаи трябва да направите това ръчно, като поставите в преамбюла на документа следните редове:

```
\pdfpagewidth=\paperwidth
\pdfpageheight=\paperheight
```

Следващият раздел по-детайлно разглежда разликата между «нормалния» LaTeX и pdfLaTeX. Основните разлики засягат три области: използваните шрифтове, форматите на включваните изображения и ръчното оформяне на хипервръзки.

4.7.2 Шрифтове

pdfLaTeX може да се справя с всякакви видове шрифтове (PK bitmaps, TrueType, PostScript type 1...), но основния за LaTeX шрифтов формат, PK bitmaps, дава много лош резултат, когато документът се изобразява на екрана с Acrobat Reader. Най-добре е да се използват шрифтове PostScript Type 1, особено за генериране на добре изглеждащи документи. Модерните TeX дистрибуции се установяват така, че това става автоматично. Най добре е да опитате. Ако това е така, можете да прескочите този параграф.

PostScript Type 1 версии на шрифтовете Computer Modern и AMSFonts са произведени от компаниите Blue Sky Research и Y&Y, Inc., които след това са предали авторските права върху тях на Американското Математическо Общество (AMS). В началото на 1997 година тези шрифтове са били направени публично достъпни, а в днешно време са включени в повечето дистрибутиви на TeX.

Обаче, ако използвате LaTeX за създаване на документи на езици, различни от английски, може би ще искате да използвате шрифтове EC, LH или CB (вж. дискусията за шрифтовете `OT1` на стр. 27). Владимир Волович е създал шрифтовия пакет cm-super, който обхваща пълните набори шрифтове EC/TC, EC Concrete, EC Bright и LH. Той е достъпен на адрес `CTAN:/tex-archive/fonts/ps-type1/cm-super`, а освен това е включен в дистрибутивите TeXLive7 и MikTeX. Аналогично, гръцките шрифтове CB във формат Type 1, създадени от Apostolos Syropoulos, са достъпни на адрес `CTAN:/tex-archive/fonts/greek/cb`. За съжаление, тези два набора шрифтове не са със същото типографско качество като Type 1 CM шрифтовете на Blue Sky/Y&Y. Хинти (hints) в тях са генерирани автоматично, и затова документите може да изглеждат на екрана не толкова добре, както при използване на Type 1 CM шрифтове на Blue Sky/Y&Y. На устройства с високо разрешение те дават резултати, идентични на оригиналните растрови шрифтове EC/LH/CB.

Ако създавате документи на латиница, имате няколко други възможности:

- Да използвате пакета aeguill, известен също като *Almost European Computer Modern with Guillemets*. Просто поставете в преамбюла реда `\usepackage{aeguill}` за да използвате виртуални шрифтове AE вместо шрифтове EC.

- Алтернативно, може да използвате пакета mltex, който обаче работи само ако вашият pdfTeX е компилиран с опция mltex.

Виртуалният набор шрифтове AE, аналогично на системата MlTeX, заставя TeX да вярва, че разполага с пълен набор шрифт с 256-символа, като създава повечето липсващи знаци за букви от шрифта CM и пренареждайки ги в кодировка EC. Този подход дава възможност да се използват висококачествени CM шрифтове във формат Type 1, достъпни на повечето системи. Тъй като шрифта сега има кодировка T1 за европейските езици, основани на латиница, работи нормално механизма на пренасяне. Единственият недостатък на този подход е в това, че изкуствените символи AE не работят във функция на търсене на Acrobat Reader, следователно вие не можете да търсите думи с акцентирани символи в крайния PDF файл.

За руски език аналогично решение е използването на виртуални шрифтове C1, достъпни на адрес
`ftp://ftp.vsu.ru/pub/tex/font-packs/c1fonts`. Тези шрифтове комбинират стандартните шрифтове CM Type 1 от колекцията Bluesky и шрифтовете CMCYR Type 1 от колекцията Paradissa и BaKoMa, достъпни на CTAN. Тъй като шрифтовете Paradissa съдържат само руски букви, шрифтовете C1 нямат други глифове за кирилица.

Друго решение е да се превключи на други PostScript Type 1 шрифтове. Всъщност, някои от тях дори се включват във всяко копие на Acrobat Reader. Тъй като тези шрифтове имат друга метрика на символите, отпечатването на текста на страницата ще се измени. Обикновено текста започва да заема повече място, тъй като CM шрифтовете са много по-компактни. Освен това, общата визуална съгласуваност на документа ще се влоши, тъй като шрифтовете Times, Helvetica и Courier (основни кандидати за такава замяна) не са били проектирани да работят в хармония в един документ.

За тази цел има два готови набора шрифтове: пакета pxfonts, който е базиран основно на *Palatino* като основен шрифт в текста, и най-добре – пакета txfonts, базиращ се на шрифта *Times*. За да използвате тези пакети, е достатъчно да поставите в преамбюла следните редове:

```
\usepackage[T1]{fontenc}
\usepackage{pxfonts}
```

Забележка: След компилация на изходния текст, може да видите в `.log`-файла редове, наподобяващи следния:

```
Warning: pdftex (file eurmo10): Font eurmo10 at ... not found
```

Те означават, че не е бил намерен някакъв шрифт, използван в документа. Вие трябва да отстраните този проблем, тъй като полученият PDF-документ може *изобщо да не показва страниците с отсъстващите символи.*

Всички тези проблеми с шрифтовете, особено отсъствието на добър набор шрифтове EC, равен по качество на шрифтовете CM във формат Type 1, заемат умовете на много хора, така че постоянно се появяват нови решения.

4.7.3 Използване на графики

Включването на графики в документ работи най-добре с пакета graphicx (вж. стр. 67). С използване на стойност `pdftex` в опцията *driver*, този пакет успешно работи с pdfLaTeX:

```
\usepackage[pdftex]{color,graphicx}
```

В горния пример е включен също пакета color, тъй като използването на цвят в документи, изобразявани в мрежата, е напълно естествено.

Толкова за добрите новини. Лошата новина е, че графики във формат Encapsulated PostScript не работят с pdfLaTeX. Ако не зададете разширение на файла в командата `\includegraphics`, пакетът graphicx избира подходящия файл, основавайки се на стойността на опцията *driver*. За `pdftex` това са формати `.png`, `.pdf`, `.jpg`, `.mps` (METAPOST) и `.tif`, но *не* и `.eps`.

Прост начин за заобикаляне на този проблем е да се конвертират вашите EPS-файлове във формат PDF с инструмента `epstopdf`, присъстващ в много системи. За векторни графики това е забележително решение. За растери (фотографии, сканирани документи) това не е идеално, защото форматът PDF сам по себе си поддържа включване на растрови изображения във формат PNG и JPEG. Форматът PNG е добър за снимки на екрана и други изображения с малко количество цветове. Форматът JPEG е добър за фотографии поради своята компактност.

Често е по-добре да не се рисуват някои геометрични фигури, а да се описват на езика на специалните команди, такъв, като METAPOST, включен в повечето дистрибутиви на TeX. В комплекта има разширено ръководство за ползване.

4.7.4 Хипер-връзки

Пакетът hyperref превръща всички вътрешни препратки в документа в хипер-връзки. За да работи това както трябва, е необходима малко магия, в частност, командата `\usepackage[pdftex]{hyperref}` трябва да бъде *последна* команда в преамбюла на документа. За настройка на поведението на пакета hyperref можете да използвате:

- или списък от разделени със запетая опции на пакета hyperref, след опцията pdftex
  `\usepackage[pdftex]{hyperref}`
- или отделни редове с командата `\hypersetup{`*опции*`}`.

Единствената задължителна опция е `pdftex`, останалите само позволяват да се изменя поведението на hyperref по подразбиране.[1] В следващия списък, стойностите по подразбиране са в прав шрифт:

`bookmarks (=true,`*`false`*`)` Показва или пречи на полето със заглавия при изобразяване на документа;

`unicode (=false,`*`true`*`)` Позволява да се използват нелатински символи в заглавията на Acrobat;

`pdftoolbar (=true,`*`false`*`)` Показва или пречи на линията с инструментите на Acrobat;

`pdfmenubar (=true,`*`false`*`)` Показва или пречи на менюто на Acrobat;

`pdffitwindow (=true,`*`false`*`)` Изменя началното увеличаване на документа (размера на страницата в «прозореца» на Acrobat);

`pdftitle (={text})` Дефинира заглавие, което се изобразява от Acrobat в прозореца `Document Info`;

`pdfauthor (={text})` Името на автора на PDF-документа;

`pdfnewwindow (=true,`*`false`*`)` Определя дали да се отваря нов прозорец, ако хипер-връзката води извън текущия документ;

`colorlinks (=false,`*`true`*`)` Огражда връзките с цветни рамки (`false`) или изменя цвета на хипер-връзките (`true`). Цветът може да се настройва със следните опциии (указаните стойности са по подразбиране):

- `linkcolor (=red)` Цвят на вътрешните хипер-връзки (раздели, страници и др.);
- `citecolor (=green)` Цвят на връзки към Литературата;
- `filecolor (=magenta)` Цвят на връзки към файлове;
- `urlcolor (=cyan)` Цвят на връзки към URL (електронна поща, WWW).

Ако ви устройват стойностите по подразбиране, използвайте:

```
\usepackage[pdftex]{hyperref}
```

За да се отвори полето на заглавията и да се оцветят хипер-връзките (стойностите `=true` могат да се изпускат):

[1] Важно е да се отбележи, че пакетът hyperref не е ограничен за работа с pdfTEX. Той може да се конфигурира за вграждане на специфична PDF информация в нормалното DVI извеждане на стандартния LaTeX, която след това попада в PS-файла, генериран с програмата `dvips` и накрая се избира Adobe Distiller за преобразуване на PS-файла във формат PDF.

```
\usepackage[pdftex,bookmarks,colorlinks]{hyperref}
```

Когато PDF-документът е предназначен за печат, по-добре е да не се използват цветни препратки, защото те ще се окажат на хартията сиви, което затруднява четенето. По-добре е да се използват цветни рамки, които не се печатат:

```
\usepackage{hyperref}
\hypersetup{colorlinks=false}
```

или да се направят препратките черни:

```
\usepackage{hyperref}
\hypersetup{colorlinks,%
            citecolor=black,%
            filecolor=black,%
            linkcolor=black,%
            urlcolor=black,%
            pdftex}
```

Ако искате да дадете информация за частта `Document Info` на PDF-файла:

```
\usepackage[pdfauthor={Pierre Desproges}%
            pdftitle={Des femmes qui tombent},%
            pdftex]{hyperref}
```

В допълнение към автоматично генерираните хипер-връзки, е възможно да се правят връзки в явен вид с командата

`\href{`*url*`}{`*text*`}`

Кодът

```
Web-страницата \href{http://www.ctan.org}{CTAN}.
```

генерира на изхода “CTAN”; при кликване с мишката върху думата “CTAN”, попадете на web-страницата CTAN.

Ако препратката води не на URL, а на локален файл, можете да използвате командата `\href`:

```
Пълният текст на документа се намира \href{manual.pdf}{тук}.
```

което генерира текст “Пълният текст на документа се намира тук.” Кликването върху думата “тук” отваря файла `manual.pdf`. (Името на файла се разглежда относно положето на текущия документ.)

Авторът на статията може да облегчи читателя при изпращане на писма с мнения, използвайки командата `\href` вътре в командата `\author` на заглавната страница на документа:

```
\author{Mary Oetiker $<$\href{mailto:mary@oetiker.ch}%
       {mary@oetiker.ch}$>$
```

Да отбележим, че препратката с пощенски адрес допълва адреса, приведен на самата страница. Това е направено затова, защото препратката
`\href{mailto:mary@oetiker.ch}{Mary Oetiker}`
е удобна в Acrobat Reader, но ще бъде невидима след печат на документа на хартия.

4.7.5 Проблеми с препратките

Когато се изчистват броячи, например, когато командата `\mainmatter` от класа документи `book` изчиства брояча за номер на страници и го променя на 1, се генерира съобщение подобно на такова:

```
! pdfTeX warning (ext4): destination with the same identifier
  (name{page.1}) has been already used, duplicate ignored
```

Това е така, защото във въвеждащата част на книгата вече е имало страница с номер 1 и всички препратки на "страница 1" вече няма да бъдат уникални; оттук и съобщението "`duplicate` has been `ignored`."

Да се избавим от това можем, като зададем на `hyperref` опция `plainpages=false`. За съжаление, това ще помогне само по отношение брояча на страниците. Още по-радикални мерки могат да се вземат с използване на опцията `hypertexnames=false`, но след това ще престанат да работят препратките към страниците от предметния указател.

4.7.6 Проблеми със заглавията

Текстът в заглавията не винаги изглежда така, както на вас би ви се искало. В заглавията са допустими доста по-малък набор символи, отколкото в нормалния LaTeX, тъй като те са "просто текст". Обикновено hyperref забелязва свързаните с това проблеми и извежда предупреждение:

```
Package hyperref Warning: Token not allowed in a
PDFDocEncoded string:
```

Вие можете да се справите с този проблем, предоставяйки текстов вариант за заглавието в замяна на проблемния текст:

`\texorpdfstring{`*текст на TeX*`}{`*текст на заглавието*`}`

Математическите изрази са първите кандидати за такава замяна:

```
\section{\texorpdfstring{$E=mc^2$}%
        {E\ =\ mc\texttwosuperior}}
```

което превръща `\section{$E=mc^2$}` в "E=mc2" за извеждане в заглавието.

Смяната на цвета също върви лошо в заглавията:

```
\section{\textcolor{red}{Red !}}
```

извежда “redRed!”. Командата `\textcolor` се игнорира, но нейният аргумент (red) се печата.

По-добър резултат се получава така:

```
\section{\texorpdfstring{\textcolor{red}{Red !}}{Red\ !}}
```

4.8 Съвместимост на изходните текстове в LaTeX и pdfLaTeX

В идеалния случай вашият документ трябва еднакво добре да се компилира както с LaTeX, така и с pdfLaTeX. Основният проблем тук е включването на графики. Просто решение е *системното премълчаване* на разширенията на файловете в командата `\includegraphics`. В този случай компилаторът автоматично избира файл с изисквания формат от текущия каталог. Всичко, което трябва да се направи е да се създадат подходящи версии на графичните файлове. LaTeX ще избира `.eps`, а pdfLaTeX ще се постарае да включи файлове с разширения `.png`, `.pdf`, `.jpg`, `.mps` или `.tif` (в указания ред).

За случаите, когато ще ви е необходимо да използвате различни кодове за обикновената и PDF-версията на документа, може някъде в самото начало на документа да направите следното:

```
\newif\ifPDF
\ifx\pdfoutput\undefined\PDFfalse
\else\ifnum\pdfoutput >0\PDFtrue
     \else\PDFfalse
     \fi
\fi
```

Тук се определя специална команда, позволяваща лесно да се пише условен код:

```
\ifPDF
  \usepackage[T1]{fontenc}
  \usepackage{aeguill}
  \usepackage[pdftex]{graphicx,color}
  \usepackage[pdftex]{hyperref}
\else
  \usepackage[T1]{fontenc}
  \usepackage[dvips]{graphicx}
  \usepackage[dvips]{hyperref}
\fi
```

В горния пример пакетът hyperref е включен дори и в не-PDF-версията. Това е направено, за да работи командата `\href` във всеки случай и нейното използване да не се отхвърля в условните оператори.

Да отбележим, че в съвременните дистрибутиви на TEX (например, в TEXLive), изборът на pdftex или dvips при извикване на graphicx и color става автоматично в съответствие с настройките на конфигурационните файлове graphics.cfg и color.cfg.

4.9 Създаване на презентации с помощта на pdfscreen

Автор: Daniel Flipo <Daniel.Flipo@univ-lille1.fr>

Вие можете да представите резултатите от вашата научна работа на дъска, проектор, или направо от вашия ноутбук с помощта на презентационен софтуер.

pdfLATEX в комбинация с пакета pdfscreen ви позволява да създавате презентации във формат PDF, толкова красиви и живи, както и в *PowerPoint*, но много по-мобилни, тъй като Acrobat Reader съществува на много повече системи.

Класът pdfscreen използва пакетите graphicx, color и hyperref с опции, настроени за екранна презентация.

За създаване на такъв тип документ обикновено се използва класът article. Фигура 4.2 показва примерен входен файл. Отначало се зарежда пакета pdfscreen със съответните опции:

screen за екранна презентация. За хартиена версия използвайте опцията print.

panelright разполага навигационния панел в дясната част на екрана. Ако панелът трябва да е вляво, използвайте опцията panelleft. Ако той не ви е нужен въобще, използвайте nopanel.

french или всеки друг поддържан език ще представи текста на навигационните бутони на съответния език. Тази опция е независима от установените опции на пакета babel. Ако вашият език не се поддържа от pdfscreen, вие все пак можете да локализирате бутоните на панела с помощта на файла pdfscreen.cfg, вж. например pdfscreen.cfg.specimen.

chocolate Цветова схема на навигационния панел. Други възможности са gray, orange, palegreen, bluelace и blue, използван по подразбиране.

След това се настройва формата на изобразяване. Тъй като презентацията винаги се представя на пълен екран, това може да се използва за настройка на размера на шрифта:

\panelwidth определя ширината на навигационния панел.

\screensize{*ширина*}{*височина*} определя ширината и височината на екрана, включително навигационния панел.

\marginsize{*ляво*}{*дясно*}{*горе*}{*долу*} определя полетата на документа. В дадения пример документът не се центрира, затова номерата на разделите остават в лявото поле.

```
\documentclass[pdftex,12pt]{article}
%%% misc extensions %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\usepackage[latin1]{inputenc}
\usepackage[english]{babel}
\usepackage[T1]{fontenc}
\usepackage{aeguill}
%%% pdfscreen %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\usepackage[screen,panelleft,chocolate]{pdfscreen}
% Screen Format
\panelwidth=25mm
%%         height width
\screensize{150mm}{200mm}
%%         left right top  bottom
\marginsize{42mm}{8mm}{10mm}{10mm}
% Color or image for background
\overlayempty
\definecolor{mybg}{rgb}{1,0.9,0.7}
\backgroundcolor{mybg}
% Logo
\emblema{MyLogo}
%%% For PPower4 (post-processor) %%%%%%%%%%%%%%%%%%%%%%%%
\usepackage{pause}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\begin{document}
\begin{slide}
\begin{itemize}
\item Good News\dots \pause \item Bad News
\end{itemize}
\end{slide}
\end{document}
```

Фигура 4.2: Пример за входен файл на pdfscreen

Може да се използва фоново изображение във всеки от поддържаните от pdfTeX формати с използване на командата

`\overlay{`*изображение*`}`

или, ако предпочитате обикновен фон, може да зададете неговия цвят с командата

`\background{`*цвят*`}`

Накрая, ако искате да поставите в навигационния панел логото на вашата организация, използвайте командата

`\emblema{`*logo*`}`

Ако вярвате в убедителността на последователното представяне на отделните точки във вашето изложение, може да използвате пакета pause. Той предлага командата `\pause`. Тя може да се поставя на онези места в текста, където искате Acrobat да задържи за момент изображението на документа. Пакетът pause е част от системата `PPower4` (P^4: *PDF Presentation Post-Processor*), която обработва документа, генериран с pdfTeX и го кара да пее, да танцува и да събира дребни. Командният ред е подобен на този:

```
ppower4 xy.pdf xyz.pdf
```

За контролиране на това, какво да се изобразява на всеки отделен слайд, използвайте средата `\begin{slide}` ... `\end{slide}`. Съдържанието на всеки слайд се изобразява на своята страницата вертикално центрирано.

След компилиране на горния пример, се получава грешка:

```
! pdfTeX warning (dest): name{contents} has been
  referenced but does not exist, replaced by a fixed one
```

Причина тук е в това, че върху навигационния панел има бутон, водещ към съдържанието, а нашият пример не съдържа команда `\tableofcontents`.

Ако искате съдържанието да се изобразява направо в навигационния панел, може да използвате опция `paneltoc` при извикване на pdfscreen. Разбира се, това ще дава удовлетворителен резултат само ако съдържанието включва малко на брой кратки заглавия. Можете да задавате кратки заглавия на вашите раздели в квадратни скоби.

Това кратко въведение се плъзга само по повърхността на възможностите на pdfscreen и `PPower4`. Всеки от тях включва собствено подробно ръководство.

5 Генерация на математически графики

Повечето хора използват LaTeX за отпечатване на текстове. Макар и да не съдържа структурно ориентиран и удобен за авторите достъп, LaTeX също предлага, макар и ограничена, възможност за генерация на графика по текстово описание. Освен това са създадени доста разширения на LaTeX за преодоляване на тези ограничния. В тази глава ще научите малко за това.

5.1 Обзор

Средата `picture` позволява да се програмират картинки направо в средата на LaTeX. Подробно описание се привежда в *LaTeX Manual* [1]. От една страна, има твърде строги ограничения, например, както наклоните на отсечки, така и радиусите на кръговете са тясно ограничени по отношение на възможни стойности. От друга страна, средата `picture` на LaTeX 2_ε въвежда командата `\qbezier`, където «`q`» означава «квадратичен». Много от често използваните криви, като окръжности, елипси или дъги, могат да се апроксимират задоволително с квадратични криви на Безие, дори за това да се изискват известни математически усилия. Освен това, ако се използва език за програмиране като Java, за генериране на блоковете на командата `\qbezier` във входния LaTeX-файл, средата `picture` става твърде мощна.

Въпреки че програмирането на картинки направо в LaTeX е строго ограничено и често твърде уморително, има основания да се прави това. Получените по такъв начин документи стават малки по обем и не е необходимо да се прилагат графични файлове към документа.

Пакети, като epic и eepic (описани, например в *The LaTeX Companion* [3]) или pstricks помагат да се елиминират ограниченията, които пречат в оригиналната среда `picture`, и силно разширяват графичните възможности на LaTeX.

Докато първите два пакета просто подобряват средата `picture`, пакетът pstricks има собствена среда за рисуване, `pspicture`. Силата на pstricks произтича от това, че той разширява възможностите на PostScript. Освен този пакет, съществуват и много други, написани за конкретни цели. Един от тях е пакетът XY-pic, описан в края на тази глава. Множеството такива пакети е детайлно описано в *The LaTeX Graphics Companion* [4] (не бъркайте с *The LaTeX Companion* [3]).

Вероятно най-мощният графичен инструмент, свързан с LaTeX, е METAPOST, програма-близнак на създадената от Donald E. Knuth програма `METAFONT`. METAPOST използва много мощен и математически строг език `METAFONT`. За разлика от `METAFONT`, генериращ растери, METAPOST генерира файлове Encapsulated PostScript, които могат да се импортират в LaTeX. За начално запознаване вижте *Ръководство за потребителя на MetaPost* [15], или ръководството *Графика в LaTeX 2ε* [17].

Много подробно работата с графики (и шрифтове) в LaTeX и TeX е описана в *TeX Unbound* [16].

5.2 Среда picture

Автор: Urs Oswald <osurs@bluewin.ch>

5.2.1 Основни команди

Средата `picture` се създава с една от двете команди:

$\boxed{\texttt{\textbackslash begin\{picture\}}(x,y)\ldots\texttt{\textbackslash end\{picture\}}}$

или

$\boxed{\texttt{\textbackslash begin\{picture\}}(x,y)(x_0,y_0)\ldots\texttt{\textbackslash end\{picture\}}}$

Числата x, y, x_0, y_0 се задават в размерност `\unitlength`, която може да се изменя във всеки момент (но не вътре в средата `picture`) с команди, като

$\boxed{\texttt{\textbackslash setlength\{\textbackslash unitlength\}\{1.2cm\}}}$

Стойността на `\unitlength` по премълчаване е 1pt. Първата двойка числа, (x, y), определя резервираното за картинката правоъгълно пространство в документа. Незадължителната втора двойка числа, (x_0, y_0), задава произволни координати на долния ляв ъгъл на резервирания правоъгълник.

Повечето команди за рисуване имат една от двете форми:

$\boxed{\texttt{\textbackslash put}(x,y)\texttt{\{}\textit{обект}\texttt{\}}}$

или

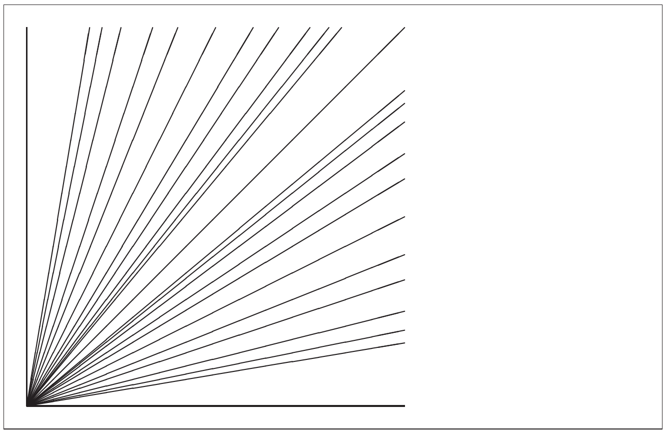
$\boxed{\texttt{\textbackslash multiput}(x,y)(\Delta x,\Delta y)\texttt{\{}n\texttt{\}\{}\textit{обект}\texttt{\}}}$

Кривите на Безие са изключение. Те се рисуват с командата

$\boxed{\texttt{\textbackslash qbezier}(x_1,y_1)(x_2,y_2)(x_3,y_3)}$

5.2.2 Отсечки

```
\setlength{\unitlength}{5cm}
\begin{picture}(1,1)
  \put(0,0){\line(0,1){1}}
  \put(0,0){\line(1,0){1}}
  \put(0,0){\line(1,1){1}}
  \put(0,0){\line(1,2){.5}}
  \put(0,0){\line(1,3){.3333}}
  \put(0,0){\line(1,4){.25}}
  \put(0,0){\line(1,5){.2}}
  \put(0,0){\line(1,6){.1667}}
  \put(0,0){\line(2,1){1}}
  \put(0,0){\line(2,3){.6667}}
  \put(0,0){\line(2,5){.4}}
  \put(0,0){\line(3,1){1}}
  \put(0,0){\line(3,2){1}}
  \put(0,0){\line(3,4){.75}}
  \put(0,0){\line(3,5){.6}}
  \put(0,0){\line(4,1){1}}
  \put(0,0){\line(4,3){1}}
  \put(0,0){\line(4,5){.8}}
  \put(0,0){\line(5,1){1}}
  \put(0,0){\line(5,2){1}}
  \put(0,0){\line(5,3){1}}
  \put(0,0){\line(5,4){1}}
  \put(0,0){\line(5,6){.8333}}
  \put(0,0){\line(6,1){1}}
  \put(0,0){\line(6,5){1}}
\end{picture}
```



Отсечки се рисуват с командата

$\boxed{\texttt{\textbackslash put(}x,y\texttt{)\{\textbackslash line(}x_1,y_1\texttt{)\{}length\texttt{\}\}}}$

Командата `\line` има два аргумента:

1. вектор на направление,
2. дължина.

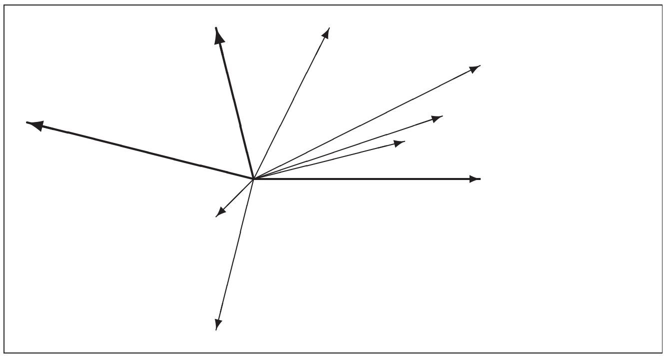
Компонентите на вектора на направление са ограничени; те могат да бъдат измежду целите числа

$$-6,\ -5\ \ldots,\ 5,\ 6,$$

и трябва да бъдат взаимно прости (да нямат общ делител, освен 1). Фигурата показва всичките 25 възможни стойности на наклона в първи квадрант. Дължината се изразява в единици `\unitlength`. Аргументът дължина е вертикалната координата в случай на вертикална отсечка и хоризонталната – във всички останали случаи.

5.2.3 Вектори

```
\setlength{\unitlength}{1mm}
\begin{picture}(60,40)
  \put(30,20){\vector(1,0){30}}
  \put(30,20){\vector(4,1){20}}
  \put(30,20){\vector(3,1){25}}
  \put(30,20){\vector(2,1){30}}
  \put(30,20){\vector(1,2){10}}
  \thicklines
  \put(30,20){\vector(-4,1){30}}
  \put(30,20){\vector(-1,4){5}}
  \thinlines
  \put(30,20){\vector(-1,-1){5}}
  \put(30,20){\vector(-1,-4){5}}
\end{picture}
```



Вектор се рисува с командата

$\boxed{\texttt{\textbackslash put}(x,y)\texttt{\{\textbackslash vector}(x_1,y_1)\texttt{\{}length\texttt{\}\}}}$

За векторите, стойностите на вектора на направление са още по-ограничени, отколкото за отсечките, а именно – те могат да бъдат измежду целите числа

$$-4, -3, \ldots, 3, 4.$$

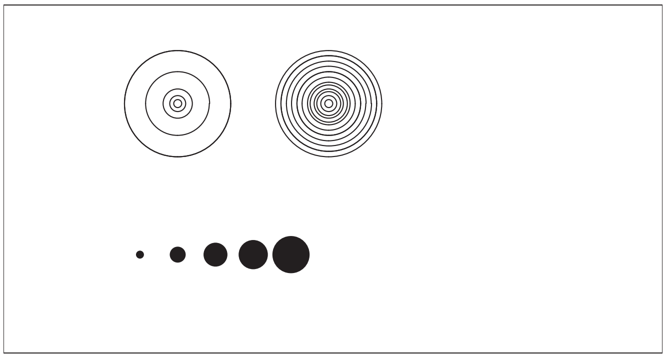
Компонентите също трябва да бъдат взаимно прости (да нямат общ делител, освен 1). Забележете ефекта на командата `\thicklines` на двата вектора, сочещи горния ляв ъгъл.

5.2.4 Окръжности

```
\setlength{\unitlength}{1mm}
\begin{picture}(60, 40)
  \put(20,30){\circle{1}}
  \put(20,30){\circle{2}}
  \put(20,30){\circle{4}}
  \put(20,30){\circle{8}}
  \put(20,30){\circle{16}}
  \put(20,30){\circle{32}}

  \put(40,30){\circle{1}}
  \put(40,30){\circle{2}}
  \put(40,30){\circle{3}}
  \put(40,30){\circle{4}}
  \put(40,30){\circle{5}}
  \put(40,30){\circle{6}}
  \put(40,30){\circle{7}}
  \put(40,30){\circle{8}}
  \put(40,30){\circle{9}}
  \put(40,30){\circle{10}}
  \put(40,30){\circle{11}}
  \put(40,30){\circle{12}}
  \put(40,30){\circle{13}}
  \put(40,30){\circle{14}}

  \put(15,10){\circle*{1}}
  \put(20,10){\circle*{2}}
  \put(25,10){\circle*{3}}
  \put(30,10){\circle*{4}}
  \put(35,10){\circle*{5}}
\end{picture}
```



Командата

`\put(`x, y`){\circle{`*диаметър*`}}`

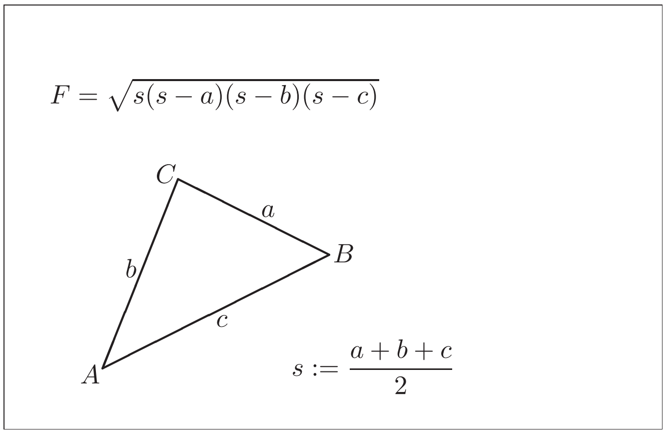
рисува окръжност с център в точката (x, y) и диаметър (не радиус!) *диаметър*. Средата `picture` допуска диаметри до около 14 мм, и дори в тези граници не са допустими всички диаметри. Командата `\circle*` рисува кръгове (запълнени окръжности).

Както и в случая на отсечки, можете да се обръщате към допълнителни пакети, такива, като eepic или pstricks. Подробно описание на тези пакети е дадено в *The LaTeX Graphics Companion* [4].

Съществува също и изход в рамките на средата `picture`. Ако не се боите да направите необходимите изчисления (или ги възлагате на програма), то може да изобразявате произволни окръжности и елипси с помощта на криви на Безие. Примери и изходни текстове на Java са дадени в *Графика в LaTeX 2ε* [17].

5.2.5 Текст и формули

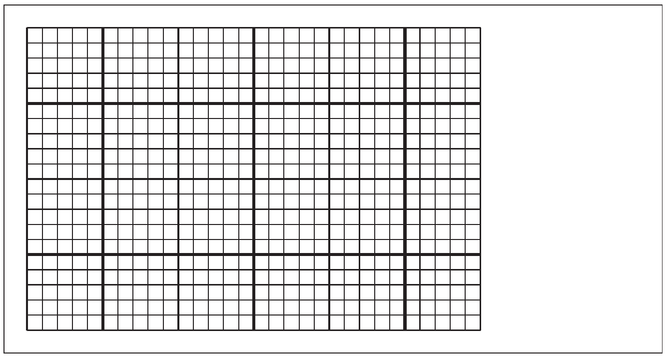
```
\setlength{\unitlength}{1cm}
\begin{picture}(6,5)
  \thicklines
  \put(1,0.5){\line(2,1){3}}
  \put(4,2){\line(-2,1){2}}
  \put(2,3){\line(-2,-5){1}}
  \put(0.7,0.3){$A$}
  \put(4.05,1.9){$B$}
  \put(1.7,2.95){$C$}
  \put(3.1,2.5){$a$}
  \put(1.3,1.7){$b$}
  \put(2.5,1.05){$c$}
  \put(0.3,4){$F=
    \sqrt{s(s-a)(s-b)(s-c)}$}
  \put(3.5,0.4){$\displaystyle
    s:=\frac{a+b+c}{2}$}
\end{picture}
```



Както показва този пример, текст и формули могат да се разполагат в средата `picture` по обичайния начин — с командата `\put`.

5.2.6 Команди `\multiput` и `\linethickness`

```
\setlength{\unitlength}{2mm}
\begin{picture}(30,20)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){31}%
    {\line(0,1){20}}
  \multiput(0,0)(0,1){21}%
    {\line(1,0){30}}
  \linethickness{0.15mm}
  \multiput(0,0)(5,0){7}%
    {\line(0,1){20}}
  \multiput(0,0)(0,5){5}%
    {\line(1,0){30}}
  \linethickness{0.3mm}
  \multiput(5,0)(10,0){3}%
    {\line(0,1){20}}
  \multiput(0,5)(0,10){2}%
    {\line(1,0){30}}
\end{picture}
```



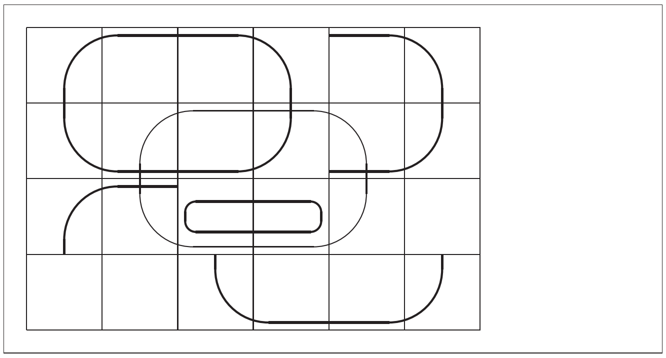
Командата

$\boxed{\texttt{\textbackslash multiput}(x,y)(\Delta x,\Delta y)\{n\}\{\textit{обект}\}}$

има 4 аргумента: начална точка, вектор на прехода от един обект към следващия, брой обекти и самия обект за рисуване. Командата `\linethickness` се прилага към хоризонтални и вертикални отсечки, но никога – към наклонени отсечки или окръжности. Тя, обаче, се прилага също и към квадратични криви на Безие!

5.2.7 Овали. Команди `\thinlines` и `\thicklines`

```
\setlength{\unitlength}{1cm}
\begin{picture}(6,4)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){7}%
    {\line(0,1){4}}
  \multiput(0,0)(0,1){5}%
    {\line(1,0){6}}
  \thicklines
  \put(2,3){\oval(3,1.8)}
  \thinlines
  \put(3,2){\oval(3,1.8)}
  \thicklines
  \put(2,1){\oval(3,1.8)[tl]}
  \put(4,1){\oval(3,1.8)[b]}
  \put(4,3){\oval(3,1.8)[r]}
  \put(3,1.5){\oval(1.8,0.4)}
\end{picture}
```



Командата

`\put(`x,y`){\oval(`w,h`)}`

или

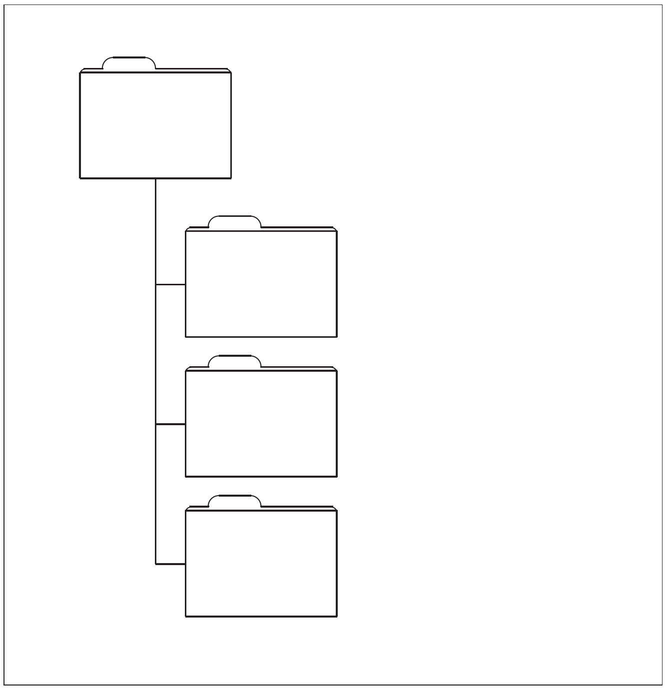
`\put(`x,y`){\oval(`w,h`)[`*позиция*`]}`

произвежда овал с център в (x, y), имащ дължина w и височина h. Незадължителният аргумент *позиция* може да приема стойности `b`, `t`, `l` и `r` (долу/горе/ляво/дясно), които могат да се комбинират, както показва дадения пример.

Дебелината на линията може да се контролира с два вида команди: `\linethickness{`*дължина*`}` от една страна, и `\thinlines` и `\thicklines` – от друга. Докато `\linethickness{`*дължина*`}` се прилага само към хоризонтални и вертикални линии (и квадратични криви на Безие), `\thinlines` и `\thicklines` се прилагат към наклонени отсечки, окръжности и овали.

5.2.8 Многократно използване на блокове с картинки

```
\setlength{\unitlength}{0.5mm}
\begin{picture}(120,168)
\newsavebox{\foldera}% обявяване
\savebox{\foldera}
  (40,32)[bl]{% определяне
  \multiput(0,0)(0,28){2}
    {\line(1,0){40}}
  \multiput(0,0)(40,0){2}
    {\line(0,1){28}}
  \put(1,28){\oval(2,2)[tl]}
  \put(1,29){\line(1,0){5}}
  \put(9,29){\oval(6,6)[tl]}
  \put(9,32){\line(1,0){8}}
  \put(17,29){\oval(6,6)[tr]}
  \put(20,29){\line(1,0){19}}
  \put(39,28){\oval(2,2)[tr]}
}
\newsavebox{\folderb}% обявяване
\savebox{\folderb}
  (40,32)[l]{%       определяне
  \put(0,14){\line(1,0){8}}
  \put(8,0){\usebox{\foldera}}
} \put(34,26){\line(0,1){102}}
\put(14,128){\usebox{\foldera}}
\multiput(34,86)(0,-37){3}
  {\usebox{\folderb}}
\end{picture}
```



Блок с картинка може да бъде *обявен* с командата

`\newsavebox{`*име*`}`

а след това *определен* с командата

`\savebox{`*име*`}(`*ширина,височина*`)[`*позиция*`]{`*съдържание*`}`

и накрая, многократно *нарисуван* с командата

`\put(`x, y`)\usebox{`*име*`}`

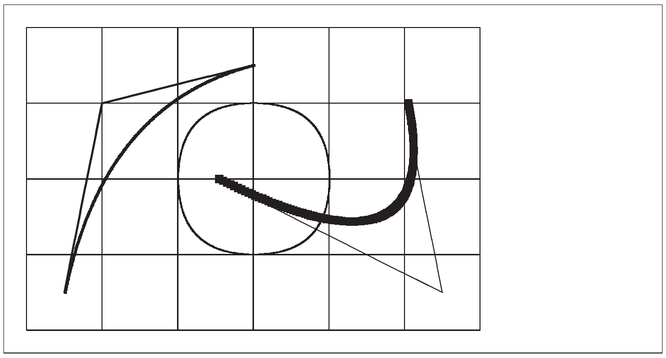
Незадължителният аргумент *позиция* определя на точката на закрепване на блока. В дадения пример той е установен на `bl`, което поставя точката на закрепване в долния ляв ъгъл на блока. Други варианти са стойностите – `t` (горе) и `r` (дясно).

Аргументът *име* всъщност става команда на LaTeX (от тук и обратната наклонена черта пред него в разглеждания пример). Блоковете могат да бъдат вложени. В този пример, `\foldera` се използва вътре в дефиницията на `\folderb`.

Командата `\oval` тук е необходимо да се използва, защото командата `\line` не работи, ако дължината на отсечката е по-малка примерно от 3 мм.

5.2.9 Квадратични криви на Безие

```
\setlength{\unitlength}{1cm}
\begin{picture}(6,4)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){7}
    {\line(0,1){4}}
  \multiput(0,0)(0,1){5}
    {\line(1,0){6}}
  \thicklines
  \put(0.5,0.5){\line(1,5){0.5}}
  \put(1,3){\line(4,1){2}}
  \qbezier(0.5,0.5)(1,3)(3,3.5)
  \thinlines
  \put(2.5,2){\line(2,-1){3}}
  \put(5.5,0.5){\line(-1,5){0.5}}
  \linethickness{1mm}
  \qbezier(2.5,2)(5.5,0.5)(5,3)
  \thinlines
  \qbezier(4,2)(4,3)(3,3)
  \qbezier(3,3)(2,3)(2,2)
  \qbezier(2,2)(2,1)(3,1)
  \qbezier(3,1)(4,1)(4,2)
\end{picture}
```



Както показва този пример, разбиването на окръжност на четири квадратични криви на Безие не дава удовлетворителен резултат. Изискват се минимум осем. Фигурата отново показва влиянието на командата `\linethickness` на хоризонталните и вертикални линии, както и на командите `\thinlines` и `\thicklines` — на наклонените отсечки. Тя също показва, че и двете команди влияят на квадратичните криви на Безие, и че всяка следваща команда отменя предишните.

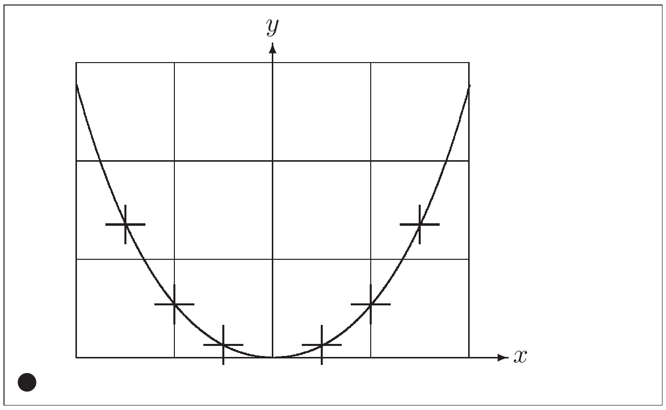
Нека $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$ задават крайните точки, а m_1, m_2 – съответните наклони на допирателните към квадратичната крива на Безие. Тогава междинната управляваща точка $S = (x, y)$ се задава с уравнението

$$\left\{ \begin{array}{lcl} x & = & \dfrac{m_2x_2 - m_1x_1 - (y_2 - y_1}{m_2 - m_1}, \\ y & = & y_i + m_i(x - x_i) \qquad (i = 1, 2). \end{array} \right. \tag{5.1}$$

В *Графика в LaTeX 2ε* [17] е дадена Java-програма, която генерира необходимите `\qbezier` командни редове.

5.2.10 Верижна линия

```
\setlength{\unitlength}{1.3cm}
\begin{picture}(4.3,3.6)(-2.5,-0.25)
  \put(-2,0){\vector(1,0){4.4}}
  \put(2.45,-.05){$x$}
  \put(0,0){\vector(0,1){3.2}}
  \put(0,3.35){\makebox(0,0){$y$}}
  \qbezier(0.0,0.0)(1.2384,0.0)
    (2.0,2.7622)
  \qbezier(0.0,0.0)(-1.2384,0.0)
    (-2.0,2.7622)
  \linethickness{.075mm}
  \multiput(-2,0)(1,0){5}
    {\line(0,1){3}}
  \multiput(-2,0)(0,1){4}
    {\line(1,0){4}}
  \linethickness{.2mm}
  \put( .3,.12763){\line(1,0){.4}}
  \put(.5,-.07237){\line(0,1){.4}}
  \put(-.7,.12763){\line(1,0){.4}}
  \put(-.5,-.07237){\line(0,1){.4}}
  \put(.8,.54308){\line(1,0){.4}}
  \put(1,.34308){\line(0,1){.4}}
  \put(-1.2,.54308){\line(1,0){.4}}
  \put(-1,.34308){\line(0,1){.4}}
  \put(1.3,1.35241){\line(1,0){.4}}
  \put(1.5,1.15241){\line(0,1){.4}}
  \put(-1.7,1.35241){\line(1,0){.4}}
  \put(-1.5,1.15241){\line(0,1){.4}}
   \put(-2.5,-0.25){\circle*{0.2}}
\end{picture}
```



На тази фигура всяка симетрична половина на верижната линия $y = \cosh x - 1$ се апроксимирана с квадратична крива на Безие. Дясната половина на кривата завършва в точката (2, 2.7622), наклона в която има стойност $m = 3.6269$. Като използваме отново уравнението (5.1), можем да изчислим междинните управляващи точки. Те се оказват (1.2384, 0) и (−1.2384, 0). С кръстчета са означени точките на *истинската* верижна линия. Грешката е едва забележима, тя е по-малка от един процент.

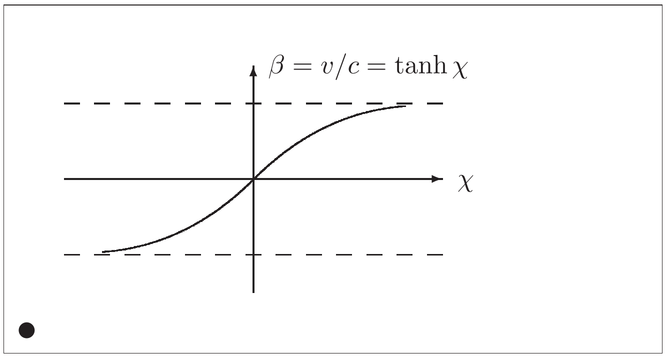
Този пример показва използването на незадължителния аргумент на командата `\begin{picture}`. Картинката е определена в удобни «математически» координати, след което командата

```
\begin{picture}(4.3,3.6)(-2.5,-0.25)
```

присвоява на долния ляв ъгъл (отбелязан с черна точка) координати (−2.5, −0.25).

5.2.11 Скорост в специалната теория на относителността

```
\setlength{\unitlength}{1cm}
\begin{picture}(6,4)(-3,-2)
  \put(-2.5,0){\vector(1,0){5}}
  \put(2.7,-0.1){$\chi$}
  \put(0,-1.5){\vector(0,1){3}}
  \multiput(-2.5,1)(0.4,0){13}
    {\line(1,0){0.2}}
  \multiput(-2.5,-1)(0.4,0){13}
    {\line(1,0){0.2}}
  \put(0.2,1.4)
    {$\beta=v/c=\tanh\chi$}
  \qbezier(0,0)(0.8853,0.8853)
    (2,0.9640)
  \qbezier(0,0)(-0.8853,-0.8853)
    (-2,-0.9640)
  \put(-3,-2){\circle*{0.2}}
\end{picture}
```



Управляващите точки на двете криви на Безие са изчислени по формулите (5.1). Положителният клон се определя от $P_1 = (0, 0)$, $m_1 = 1$ и $P_2 = (2, \tanh 2)$, $m_2 = 1/\cosh^2 2$. Картинката отново се определя в математически удобни координати, а долния ляв ъгъл получава математически координати $(-3, -2)$ (черната точка).

5.3 XY-pic

Автор: Alberto Manuel Brandão Simões <albie@alfarrabio.di.uminho.pt>

xy е специален пакет за рисуване на диаграми. За неговото използване просто добавете към преамбюла на документа следния ред:

```
\usepackage[опции]{xy}
```

където *опции* е списък от функции на XY-pic, които искате да заредите. Тези опции, на първо място, са полезни при тестване на пакета. Препоръчвам ви да използвате опцията `all`, инструктираща LaTeX да зареди всички команди на XY.

Диаграмите на XY-pic се рисуват в матричен вид, като всеки елемент на диаграмата се поставя в определена клетка на матрица:

```
\begin{displaymath}
\xymatrix{A & B \\
          C & D }
\end{displaymath}
```

$$\begin{matrix} A & B \\ C & D \end{matrix}$$

Командата `\xymatrix` трябва да се използва в математически режим. Тук ние сме задали два реда и два стълба. За да направим от тази матрица диаграма, добавяме стрелки-вектори с командата `\ar`.

```
\begin{displaymath}
\xymatrix{ A \ar[r] & B \ar[d] \\
           D \ar[u] & C \ar[l] }
\end{displaymath}
```

$$
\begin{tikzcd}
A \arrow[r] & B \arrow[d] \\
D \arrow[u] & C \arrow[l]
\end{tikzcd}
$$

Командата за рисуване на вектора се поставя в клетката, от която излиза вектора. Аргумент е направлението, в което е насочен векторът (`up`, `down`, `right` и `left`).

```
\begin{displaymath}
\xymatrix{
  A \ar[d] \ar[dr] \ar[r] & B \\
  D                       & C }
\end{displaymath}
```

$$
\begin{tikzcd}
A \arrow[d] \arrow[dr] \arrow[r] & B \\
D & C
\end{tikzcd}
$$

За рисуване на диагонали, укажете повече от едно направления. Можете също така да повторите знак за направление за рисуване на по-дълги вектори.

```
\begin{displaymath}
\xymatrix{
  A \ar[d] \ar[dr] \ar[drr] & & \\
  B                        & C & D }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[d] \arrow[dr] \arrow[drr] & & \\ B & C & D \end{tikzcd}$$

Могат да се рисуват още по-интересни диаграми, добавяйки към векторите етикети. За това се използват обикновените оператори за долни и горни индекси.

```
\begin{displaymath}
\xymatrix{
  A \ar[r]^f \ar[d]_g &
           B \ar[d]^{g’} \\
  D \ar[r]_{f’}       & C }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ D \arrow[r, "f'"'] & C \end{tikzcd}$$

Както е показано по-горе, тези оператори се използват както в математически режим. Единствената разлика е в това, че горният индекс означава «над стрелката на вектора», а долният – «под стрелката». Има още и трети оператор, вертикална черта: `|`. Той поставя текста *в* стрелката.

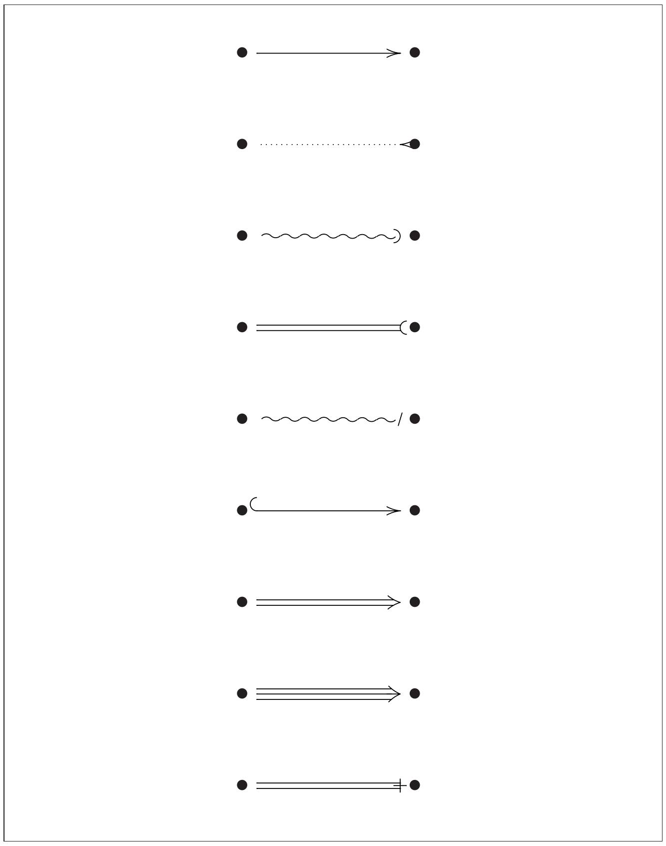
```
\begin{displaymath}
\xymatrix{
  A \ar[r]|f \ar[d]|g &
           B \ar[d]|{g’} \\
  D \ar[r]|{f’}       & C }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[r, "f" description] \arrow[d, "g" description] & B \arrow[d, "g'" description] \\ D \arrow[r, "f'" description] & C \end{tikzcd}$$

За да нарисувате стрелка с интервал в нея, използвайте командата `\ar[...]|\hole`.

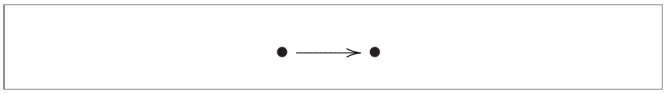
В някои случаи е важно да се различават няколко видове стрелки. Това може да се направи чрез поставяне на етикети върху тях или чрез промяна на техния вид:

```
\shorthandoff{"}
\begin{displaymath}
\xymatrix{
 \bullet\ar@{->}[rr] && \bullet\\
 \bullet\ar@{.<}[rr] && \bullet\\
 \bullet\ar@{~)}[rr] && \bullet\\
 \bullet\ar@{=(}[rr] && \bullet\\
 \bullet\ar@{~/}[rr] && \bullet\\
 \bullet\ar@{^{(}->}[rr]&& \bullet\\
 \bullet\ar@2{->}[rr] && \bullet\\
 \bullet\ar@3{->}[rr] && \bullet\\
 \bullet\ar@{=+}[rr] && \bullet
}
\end{displaymath}
\shorthandon{"}
```

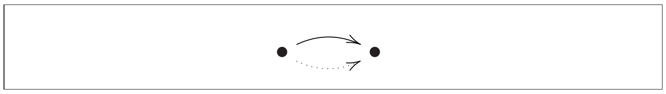


Забележете разликата между следните две диаграми:

```
\begin{displaymath}
\xymatrix{
 \bullet \ar[r]
        \ar@{.>}[r] &
            \bullet }
\end{displaymath}
```



```
\begin{displaymath}
\xymatrix{
 \bullet \ar@/^/[r]
        \ar@/_/@{.>}[r] &
                \bullet }
\end{displaymath}
```



Модификаторите между наклонените черти определят, как ще се рисуват кривите. XY-pic предлага много способи за въздействие на върху начина на рисуване на кривите; подробности вижте в документацията на XY-pic.

6 Настройка на LaTeX

Документите, произведени с използване на изучените до сега команди, изглеждат напълно приемливо за широката аудитория. Те не изглеждат модно, подчинявайки се на всички правила на добрия тон в набора, затова се четат леко и са приятни за гледане.

Обаче има ситуации, в които LaTeX не предоставя команда или среда, удовлетворяващи вашите нужди, или произвеждания от някаква съществуваща команда печат не отговаря на вашите изисквания.

В тази глава ще се опитам да дам съвети за това как да учим LaTeX на нови трикове, и как да се направи неговото извеждане различно от това, което се прави по подразбиране.

6.1 Нови команди, среди и пакети

Може би сте забелязали, че всички команди, които въвеждам в тази книга се отпечатват в рамка и че се появяват в указателя в края на книгата. Вместо директно да се използват необходимите за това команди на LaTeX, аз съм създал пакет, в който се дефинират нови команди и среди за тази цел. Сега мога просто да пиша:

```
\begin{lscommand}
\ci{dum}
\end{lscommand}
```

$\boxed{\texttt{\textbackslash dum}}$

В този пример се използва както нова среда, наречена `lscommand`, която отговаря за рисуване на рамка около командата, така и нова команда, наречена `\ci`, която отпечатва името на командата и праща съответния елемент в указателя. Можете да проверите това, като потърсите командата `\dum` в указателя накрая на книгата, където ще намерите запис за `\dum`, насочващ към всяка страница, на която е била спомената командата `\dum`.

Ако някога реша, че не ми харесват повече команди, отпечатани в рамка, просто ще изменя дефиницията на средата `lscommand`, за да създам ново извеждане. Това е много по-лесно, отколкото да се обходи целия документ, за да се издирят всички места, където са използвани общи команди на LaTeX и да се рисуват рамки около думи.

6.1.1 Нови команди

За да добавите ваши собствени команди, използвайте командата

`\newcommand{`*име*`}[`*брой-арг*`]{`*определение*`}`

Обикновено тази команда изисква два аргумента: *име* на командата, която създавате и *определение* на командата. Аргументът *брой-арг* в квадратните скоби е незадължителен и определя броя на аргументите, които изисква новата команда (възможно е до 9). Ако *брой-арг* липсва, по подразбиране е 0, т.е. не се допускат аргументи.

Следващите два примера трябва да ви помогнат да получите понятие. Първият пример определя нова команда, наречена `\tnss`. Това е съкращение на “The Not So Short Introduction to LaTeX 2ε”. Такава команда е удобна, ако многократно трябва да пишете името на тази книга.

```
\newcommand{\tnss}{The not
   so Short Introduction to
   \LaTeXe}
% в тялото на документа:
``\tnss'' \ldots{} ``\tnss''
```

“The not so Short Introduction to LaTeX 2ε” … “The not so Short Introduction to LaTeX 2ε”

Следващият пример показва, как да се определи нова команда, която допуска един аргумент. Етикетът `#1` се заменя със зададения аргумент. Ако искате да използвате повече от един аргумент, използвайте `#2`, и така нататък.

```
\newcommand{\txsit}[1]
    {Това е \emph{#1} кратко
      въведение в \LaTeXe}
% в тялото на документа
\begin{itemize}
\item \txsit{не много}
\item \txsit{много}
\end{itemize}
```

- Това е *не много* кратко въведение в LaTeX 2ε
- Това е *много* кратко въведение в LaTeX 2ε

LaTeX няма да ви позволи да създадете нова команда, която би изменила вече съществуваща. Но има специална команда, в случай, че вие категорично искате да измените съществуваща команда: `\renewcommand`. Тя има същия синтаксис, както и командата `\newcommand`.

В някои случаи може също да се използва командата `\providecommand`. Тя работи така, както `\newcommand`, но, ако командата вече е дефинирана, LaTeX 2ε мълчаливо я игнорира.

Съществуват някои особености, свързани с интервалите след команди на LaTeX. Подробности вижте на страница 5.

6.1.2 Нови среди

Аналогично на командата `\newcommand`, съществува команда за създаване на ваша собствена среда. Командата `\newenvironment` има следния синтаксис:

`\newenvironment{`*име*`}[`*брой-арг*`]{`*преди*`}{`*след*`}`

Подобно на командата `\newcommand`, `\newenvironment` може да има незадължителен аргумент. Материалът, определен в аргумента *преди*, се обработва преди обработката на текста вътре в средата. Материалът, включен в аргумента *след*, се обработва, когато се срещне командата `\end{`*име*`}`. Следният пример илюстрира използването на командата `\newenvironment`.

```
\newenvironment{king}
 {\rule{1ex}{1ex}%
      \hspace{\stretch{1}}}
 {\hspace{\stretch{1}}%
      \rule{1ex}{1ex}}

\begin{king}
Моите смирени почитания\ldots
\end{king}
```

■ Моите смирени почитания… ■

Аргументът *брой-арг* се използва по същия начин, както в командата `\newcommand`. LaTeX контролира дали не определяте вече съществуваща среда. Ако искате все пак да смените съществуваща среда, можете да използвате командата `\renewenvironment`. Тя има същия синтаксис като `\newenvironment`.

Командите, използвани в този пример, ще бъдат разяснени по-късно: описание на командата `\rule` има на стр. 115, командата `\stretch` е описана на стр. 108, а описанието на командата `\hspace` се намира на стр. 108.

6.1.3 Допълнително пространство

Когато създавате нова среда, можете лесно да получите вмъкнато в нея допълнително пространство, което в края на краищата може да има фатален ефект. Например, в случая се създава среда за заглавие, която забранява неговата собствена команда за нов ред, както и командата за нов ред на следващия параграф. Командата `\ignorespaces`, поставена в началния блок на средата, ще игнорира цялото пространство след изпълнение на началния блок. Крайният блок е малко по-сложен, тъй като в края на средата се среща специално действие. С командата `\ignorespacesafterend` LaTeX ще даде като резултат `\ignorespaces`, след изпълнението на специалния край.

```
\newenvironment{simple}%
 {\noindent}%
 {\par\noindent}

\begin{simple}
See the space\\to the left.
\end{simple}
Same\\here.
```

 See the space
to the left.

 Same
here.

```
\newenvironment{correct}%
 {\noindent\ignorespaces}%
 {\par\noindent\ignorespacesafterend}

\begin{correct}
No space\\to the left.
\end{correct}
Same\\here.
```

No space
to the left.

Same
here.

6.1.4 Командни редове на LATEX

Ако работите в система Unix, например OS, вие можете да използвате Makefiles за изграждане на вашите LATEX проекти. В тази връзка, може би ви интересува как се правят различни версии на един и същ документ посредством извикване на LATEX с commandline параметри. Ако добавите следната структура към вашия документ:

```
\usepackage{ifthen}
\ifthenelse{\equal{\blackandwhite}{true}}{
  % режим "черно и бяло"; прав\'и следното..
}{
  % "цветен" режим; прав\'и нещо друго..
}
```

Сега можете да извикате LATEX така:

```
latex '\newcommand{\blackandwhite}{true}\input{test.tex}'
```

Първо командата `\blackandwhite` е определена и след това се чете конкретен файл с командата input. С установяването след `\blackandwhite` на стойност false, ще бъде произведена цветна версия на документа.

6.1.5 Ваш собствен пакет

Ако определяте много нови среди и команди, преамбюлът на вашия документ ще стане твърде дълъг. В тази ситуация е разумно да се създаде LATEX-пакет, съдържащ дефинициите на всички ваши команди и среди. След това можете да използвате командата `\usepackage` за да стане пакетът достъпен в документа.

```
% Пакет за демонстрация от Tobias Oetiker.
\ProvidesPackage{demopack}
\newcommand{\tnss}{Не много кратко въведение в \LaTeXe}
\newcommand{\txsit}[1]{\emph{#1} кратко
                        въведение в \LaTeXe}
\newenvironment{king}{\begin{quote}}{\end{quote}}
```

Фигура 6.1: Примерен пакет

Създаването на пакета основно се състои в пренасяне на съдържанието на вашия преамбюл в отделен файл с име, завършващо на `.sty`. Има само една една специална команда,

`\ProvidesPackage{`*име на пакета*`}`

която трябва да се използва в самото начало на файла с вашия пакет. `\ProvidesPackage` указва на LaTeX името на пакета, което ще му позволи да издаде смислено съобщение за грешка, ако се опитвате да включите пакета два пъти. Фигура 6.1 показва малък примерен пакет, съдържащ дефинираните в горните примери команди.

6.2 Шрифтове и техните размери

6.2.1 Команди за смяна на шрифта

LaTeX избира подходящия вид и размер на шрифта, основавайки се на логическата структура на документа (раздели, препратки, ...). В някои случаи може да е желателно да се смени шрифта ръчно. За да направите това, можете да използвате командите, изброени в Таблица 6.1 и Таблица 6.2. Действителният размер на всеки шрифт се определя от дизайна и зависи от класа на документа и неговите опции. Таблица 6.3 показва абсолютните точкови размери за онези команди, които се използват като инструменти в стандартните класове документи.

```
{\small малък,
\textbf{плътен},
\Large голям
\textit{курсив}.}
```

малък, **плътен**, ГОЛЯМ *курсив*.

Важна особеност на LaTeX 2_ε е това, че атрибутите на шрифта са независими. Това означава, че можете да давате команди за смяна на размера или дори на шрифта, съхранявайки при това установените по-рано атрибути за наклон или наситеност.

В *математически режим* можете да използвате *команди* за смяна на шрифта, за да излезете временно от *математически режим* и да влезете в нормален текст.

Ако искате да превключите на друг шрифт за математическо отпечатване, вие се нуждаете от друг, специален набор команди; вж. Таблица 6.4.

Таблица 6.1: Шрифтове

`\textrm{...}`	прав шрифт	`\textsf{...}`	sans serif
`\texttt{...}`	`пишеща машина`		
`\textmd{...}`	нормален	`\textbf{...}`	**плътен**
`\textup{...}`	прав шрифт	`\textit{...}`	*курсив*
`\textsl{...}`	*наклонен шрифт*	`\textsc{...}`	МАЛКИ ГЛАВНИ
`\emph{...}`	*отделéн шрифт*	`\textnormal{...}`	нормален

Таблица 6.2: Размер на шрифта

`\tiny`	мъничък	`\Large`	по-голям
`\scriptsize`	много малък	`\LARGE`	най-голям
`\footnotesize`	твърде малък	`\huge`	едър
`\small`	малък	`\Huge`	грамаден
`\normalsize`	нормален		
`\large`	голям		

Таблица 6.3: Абсолютни размери на шрифтовете в стандартните класове

Размер	*10pt (по умълчаване)*	*опция 11pt*	*опция 12pt*
`\tiny`	5pt	6pt	6pt
`\scriptsize`	7pt	8pt	8pt
`\footnotesize`	8pt	9pt	10pt
`\small`	9pt	10pt	11pt
`\normalsize`	10pt	11pt	12pt
`\large`	12pt	12pt	14pt
`\Large`	14pt	14pt	17pt
`\LARGE`	17pt	17pt	20pt
`\huge`	20pt	20pt	25pt
`\Huge`	25pt	25pt	25pt

Във връзка с командите за размера на шрифта, забележителна роля играят фигурните скоби. Те се използват за построяване на *групи*. Групите ограничават областта на действие на повечето команди на LaTeX.

```
Той  харесва {\LARGE големи и
{\small малки} букви}.
```

Той харесва голeми и малки букви.

Командите за смяна размера на шрифта, променят също и разстоянието между редовете, но само ако параграфа завършва в областта на действие на командата за смяна размера на шрифта. Затова затваряща фигурна скоба `}` не трябва да се слага прекалено рано. Забележете положението на командата `\par` в следващите два примера [1]:

```
{\Large Не четете това! То не
е вярно. Вярвайте ми!\par}
```

Не четете това! То не е вярно. Вярвайте ми!

```
{\Large Това също е лъжа.
Но помнете, че аз съм лъжец.}\par
```

Това също е лъжа. Но помнете, че аз съм лъжец.

Ако искате да активирате команда за изменение на размера към целия параграф с текста или повече от това, то тогава по-добре да използвате синтаксис на среда за смяна размера на шрифта.

[1] Командата `\par` е еквивалентна на празен ред.

Таблица 6.4: Математически шрифтове

Команда	*Пример*	*Извеждане*
`\mathcal{...}`	`$\mathcal{B}=c$`	$\mathcal{B}=c$
`\mathrm{...}`	`$\mathrm{K}_2$`	K_2
`\mathbf{...}`	`$\sum x=\mathbf{v}$`	$\sum x=\mathbf{v}$
`\mathsf{...}`	`$\mathsf{G\times R}$`	$\mathsf{G\times R}$
`\mathtt{...}`	`$\mathtt{L}(b,c)$`	$\mathtt{L}(b,c)$
`\mathnormal{...}`	`$\mathnormal{R_{19}}\neq R_{19}$`	$\mathnormal{R_{19}}\neq R_{19}$
`\mathit{...}`	`$\mathit{ffi}\neq ffi$`	$\mathit{ffi}\neq ffi$

```
\begin{Large}
Това е лъжа. Но
кое в наши дни\ldots
\end{Large}
```

Това е лъжа. Но кое в наши дни...

Това ще ви спаси от преброяването на много фигурни скоби.

6.2.2 Опасност, Will Robinson, опасност

Както е отбелязано в началото на тази глава, опасно е да се претрупват вашите документи с явни команди, като описаните по-горе, защото това противоречи на основната идея на LaTeX: разделяне на логическата и визуалната структура на документа. Това означава, че ако използвате едни и същи команди за смяна на шрифта на няколко места в документа за отпечатване на специален вид информация, вие трябва да използвате `\newcommand`, за да дефинирате «логическа команда» за смяна размера на шрифта.

```
% в преамбюла или пакета
\newcommand{\oops}[1]{\textbf{#1}}
% в документа
Не \oops{влизайте} в тази стая.
В нея има \oops{машина} с
неизвестен произход и предназначение.
```

Не **влизайте** в тази стая. В нея има **машина** с неизвестен произход и предназначение.

Този подход има това преимущество, че ако решите по-късно да използвате друго визуално представяне на опасността, вместо `\textbf`, можете да го направите без необходимост да преровите целия документ, търсейки всички команди `\textbf` и определяйки дали всяко от тях е било използвано за посочване на опасност, или е нещо друго.

6.2.3 Съвет

За завършване на нашето пътешествие в света на шрифтовете и шрифтовите размери, позволете да ви дам един съвет:

Помнете! *Колкото* пОвече шрифтове И размери използвате **във** вашия документ, *толкова* по-леко ТОЙ СЕ ЧЕТЕ и толкова *по-красив* той ще бЪДе.

6.3 Интервали

6.3.1 Интервали между редовете

Ако искате да използвате по-големи интервали между редовете, можете да промените тяхната стойност с поставянето на командата

\linespread{*коефициент*}

в преамбюла на документа. Използвайте `\linespread{1.3}` за междуредов интервал «едно и половина», а за «двоен» интервал – `\linespread{1.6}`. Нормално редовете не са «дебели», така че по подразбиране коефициента на дебелина на редовете е равен на 1.

Да отбележим, че ефектът от командата `\linespread` е твърде драстичен и затова тя не е подходяща за публикации. Затова, ако имате сериозни основания за промяна на междуредовия интервал, по-добре използвайте следната команда:

```
\setlength{\baselineskip}{1.5\baselineskip}
```

```
{\setlength{\baselineskip}%
          {1.5\baselineskip}
Този параграф е отпечатан с интервал
1.5 пъти по-голям отколкото е бил
преди. Обърнете внимание на командата
\par{} в края на параграфа.\par}

Ясно е, защо е даден за пример този
параграф: той показва, че след
затваряне на фигурната скоба,
всичко се връща към нормалното
състояние.
```

Този параграф е отпечатан с интервал 1.5 пъти по-голям отколкото е бил преди. Обърнете внимание на командата

в края на параграфа.

Ясно е, защо е даден за пример този параграф: той показва, че след затваряне на фигурната скоба, всичко се връща към нормалното състояние.

6.3.2 Форматиране на параграфи

В LaTeX има два параметъра, които влияят на вида на параграфа. Поставяйки дефиниция от вида

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

в преамбюла на входния файл, вие можете да смените оформлението на параграфите. Тези две команди увеличават разстоянието между параграфите и установяват параграфния отстъп равен на нула.

Във втората команда, `plus` и `minus` указват на TeX, че може да увеличава и намалява интервала между параграфите с указаните величини, ако това е необходимо за правилното разполагане на параграфите на страницата.

В Европа, параграфите често се отделят с интервали и не се прави никакъв отстъп. Обаче имайте предвид, че това влияе и на съдържанието: неговите редове също стават по-разредени. За да се избегне това, тези команди могат да се пренесат от преамбюла на документа някъде след `\tableofcontents`, или да не се използват изобщо, защото вие ще откриете, че повечето професионални книги използват «отстъп», а не интервали, за отделяне на параграфите.

Ако искате да направите параграфен отстъп в параграф, който няма такъв, поставете командата[1]

```
\indent
```

в началото на параграфа. Разбира се, ефект от нея ще има само ако `\parindent` не е установен равен на нула.

За създаване на параграф без отстъп, може да използвате

```
\noindent
```

като първа команда в параграфа. Това е удобно, когато започвате документа с текст, а не с команди за секциониране.

6.3.3 Хоризонтални интервали

LaTeX определя автоматично интервалите между думите и изреченията. За да добавите хоризонтален интервал, използвайте

\hspace{*дължина*}

Ако такъв интервал трябва да бъде задържан, дори ако той попада в началото или края на реда, то използвайте `\hspace*` вместо `\hspace`. В най-простите случаи *дължина* е просто число плюс единица за измерване. Най-важните единици са изброени в Таблица 6.5.

```
Тук\hspace{1.5cm}интервалът
е 1,5см.
```

Тук   интервалът е 1,5см.

Командата

\stretch{n}

генерира специален «еластичен» интервал. Той се разтяга, докато се запълни цялото оставащо място на реда. Ако на един ред се използват две команди `\hspace{\stretch{n}}`, то те се разтягат пропорционално на своите коефициенти.

[1]За добавяне на отстъп към първия параграф след заглавие на раздел използвайте пакета indentfirst от комплекта 'tools'.

Таблица 6.5: Единици за размерност в TEX

`mm`	милиметър $\approx 1/25$ in	⊔
`cm`	сантиметър = 10 mm	⊔
`in`	inch = 25.4 mm	⊔
`pt`	точка $\approx 1/72$ in $\approx \frac{1}{3}$ mm	⊔
`em`	примерна ширина на буквата 'M' в текущия шрифт	⊔
`ex`	примерна височина на буквата 'x' в текущия шрифт	⊔

```
x\hspace{\stretch{1}}
x\hspace{\stretch{3}}x
```

x x x

При използване на хоризонтални интервали заедно с текст, може да има смисъл да се генерира интервал, който регулира своя размер според размера на текущия шрифт. Това може да се направи чрез относителните единици за размерност `em` и `ex`:

```
{\Large{}big\hspace{1em}y}\\
{\tiny{}tin\hspace{1ex}y}
```

big y
tin y

6.3.4 Вертикални интервали

Интервалите между параграфи, раздели, подраздели, ... се определят от LaTeX автоматично. Ако е необходимо, допълнителен вертикален интервал *между два параграфа* може да се добави с командата

`\vspace{`*длина*`}`

Тази команда трябва обикновено да се поставя между два празни реда. Ако това пространство трябва да се остави горе или долу на страницата, използвайте вариант на командата със звезда: `\vspace*`.

Командата `\stretch`, в комбинация с `\pagebreak`, може да се използва за отпечатване на текст на последния ред на страницата или за вертикално центриране на текст на страницата.

```
Някакъв текст\ldots

\vspace{\stretch{1}} Това ще се окаже на последния ред на
страницата.\pagebreak
```

Допълнителен интервал между два реда в *един* параграф или вътре в таблица се определя с командата

`\\[`*дължина*`]`

С помощта на `\bigskip` и `\smallskip` можете да оставите предварително определени вертикални интервали, без да се замисляте за конкретни числа.

6.4 Макет на страницата

LaTeX 2ε ви позволява да определите размер на хартията в командата `\documentclass`. Това автоматично определя съответните текстови полета. Но понякога предопределените стойности може да не ви устройват. Естествено, вие можете да ги измените. Фигура 6.2 показва всички параметри, които могат да бъдат променени. Тя е генерирана с помощта на пакета layout от комплекта 'tools'.[1]

Почакайте! . . . преди веднага да се хвърлите в безумието да правите «тази твърде тясна страница малко по-широка», нека малко да помислим. Подобно на други неща, има сериозни основания за това, макетът на страницата в LaTeX да е такъв, какъвто е.

Безусловно, ако сравните със страница, отпечатана с ново-инсталиран MS Word, то страниците на LaTeX изглеждат ужасно тесни. Обаче, погледнете вашата любима книга[2] и пребройте символите на един стандартен текстов ред. Ще откриете, че на всеки ред има не повече от 66 символа. Сега повторете това със страниците на LaTeX. Ще видите, че и тук също има около 66 символа на ред. Опитът показва, че четенето става толкова по-трудно, колкото повече символи има на един обикновен ред. Това е така, защото за очите е трудно да се преместват от края на един ред към началото на следващия. Именно затова вестниците често се отпечатват в няколко колони.

Така че, ако увеличите ширината на вашия текст, имайте предвид, че вие затруднявате живота на неговите читатели. И така, стига предупреждения, обещах да ви разкажа как да се направи това. . .

LaTeX предоставя две команди за изменение на тези параметри. Те обикновено се използват в преамбюла на документа.

Първата команда присвоява фиксирана стойност на всеки от параметрите:

`\setlength{`*параметър*`}{`*дължина*`}`

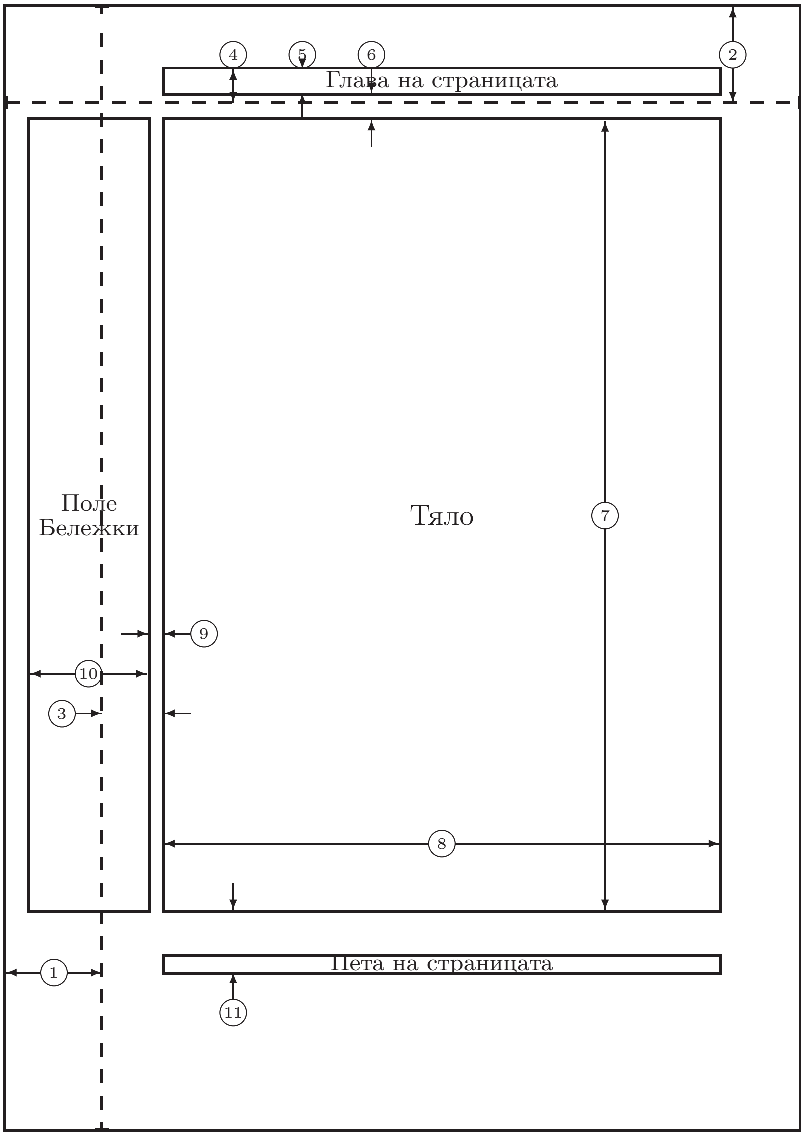
Втората команда добавя дължина към всеки от параметрите:

`\addtolength{`*параметър*`}{`*дължина*`}`

Втората команда всъщност е по-полезна, отколкото `\setlength`, защото ви позво-

[1] `CTAN:/tex-archive/macros/latex/required/tools`

[2] Има се предвид истинска печатна книга, издадена от уважавано издателство.



1	`1 инч + \hoffset`	2	`1 инч + \voffset`
3	`\oddsidemargin = -12pt` или `\evensidemargin`	4	`\topmargin = -25pt`
5	`\headheight = 18pt`	6	`\headsep = 20pt`
7	`\textheight = 595pt`	8	`\textwidth = 418pt`
9	`\marginparsep = 12pt`	10	`\marginparwidth = 89pt`
11	`\footskip = 47pt`		`\marginparpush = 6pt` (не е показано)
	`\hoffset = 0pt`		`\voffset = 0pt`
	`\paperwidth = 597pt`		`\paperheight = 845pt`

Фигура 6.2: Параметри на макета на страницата.

лява да правите настройка относно съществуващите установявания. За да добавя един сантиметър към общата ширина на текста, например, в преамбюла поставям следното:

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

В този контекст, може да видите пакета calc. Той позволява да се използват аритметични операции в аргумента на `\setlength` и на други места, където можете да въвеждате числени стойности в аргументите на функция.

6.5 Още за дължините

Винаги, когато това е възможно, въздържайте се от използване на абсолютни величини в LaTeX документите . Аз по-скоро се опитвам да основавам нещата на ширина или височина на други елементи на страницата. За ширина на фигура така може да служи `\textwidth`, за да запълва тя изцяло страницата.

Следните три команди позволяват да се определят ширината, височината и дълбочината на текстов стринг.

`\settoheight{`*променлива*`}{`*текст*`}`
`\settodepth{`*променлива*`}{`*текст*`}`
`\settowidth{`*променлива*`}{`*текст*`}`

Следващият пример показва възможното приложение на тези команди.

```
\flushleft
\newenvironment{vardesc}[1]{%
  \settowidth{\parindent}{#1:\ }
  \makebox[0pt][r]{#1:\ }}{}

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

\begin{vardesc}{където}$a$,
$b$ -- прилежащи към правия
ъгъл на правоъгълния триъгълник.

$c$ -- самотната хипотенуза
на този триъгълник.

$d$ -- въобще тук не участва.
Ето ви загадка\ldots
\end{vardesc}
```

$$a^2 + b^2 = c^2$$

където: a, b – прилежащи към правия ъгъл на правоъгълния триъгълник.

c – самотната хипотенуза на този триъгълник.

d – въобще тук не участва. Ето ви загадка…

6.6 Блокове

LaTeX изгражда страниците, придвижвайки блокове. Отначало всяка буква е малък блок, който се залепва към други букви, формирайки дума. Думата се залепва за други думи, но със специално еластично лепило, което може да се разтяга или свива така, че точно да запълни реда на страницата.

Признавам, че това е твърде опростена версия на това, което всъщност става, но идеята е в това, че TeX винаги работи с блокове и лепило. Не само буквата може да бъде блок. Вие можете да поставите в блок практически всичко, дори и други блокове. Всеки блок след това се обработва с LaTeX, както ако това е отделна буква.

В предните глави вие вече сте срещали блокове, макар че не съм споменавал за това. Примери могат да бъдат средите `tabular` или `\includegraphics`, и двете произвеждащи блок. Това означава, че вие лесно можете да подредите две таблици или фигури една до друга. Само трябва да се убедите, че тяхната обща ширина не превишава `\textwidth`.

Вие можете също да опаковате всеки параграф в блок, или с командата

`\parbox[`*позиция*`]{`*ширина*`}{`*текст*`}`

или в средата

`\begin{minipage}[`*позиция*`]{`*ширина*`}` текст `\end{minipage}`

Параметърът *позиция*, приемащ стойност една от буквите `c, t` или `b`, контролира вертикалното позициониране на блока по отношение на базовата линия на съседния текст. *Ширина* приема за аргумент дължината, определяща ширината на блока. Основната разлика между `\minipage` и `\parbox` е в това, че вътре в `\parbox` не могат да се използват всички команди и среди, докато вътре в `\minipage` може практически всичко.

Докато `\parbox` опакова целия параграф, разбивайки редовете, съществува клас блокови команди, обработващи само хоризонтално разположен материал. Една от тях ние вече знаем. Тя се нарича `\mbox` и просто опакова последователност от блокове в един блок, което може да се използва за предотвратяване разделянето от LaTeX на две свързани думи. Тъй като вие можете да поставяте едни блокове в други, тези опаковчици на хоризонтални блокове са изключително гъвкави.

`\makebox[`*ширина*`][`*позиция*`]{`*текст*`}`

Ширина определя ширината на получения блок така, както той се вижда отвън. [1] Освен изрази за дължина, вие може да използвате `\width`, `\height`, `\depth` и `\totalheight` в параметъра *ширина*. Те приемат стойности, равни на стойностите,

[1] Това означава, че тя може да бъде по-малка, отколкото материала вътре в блока. Вие дори можете да установите ширината на 0pt, така че текста вътре в блока да се отпечатва, без да въздейства на съседните блокове.

получени при измерването на отпечатвания *текст*.[1] Параметърът *позиция* приема еднобуквена стойност: **c**enter: центриран, flush**l**eft: подравнен вляво, flush**r**ight: подравнен вдясно или **s**pread: равномерно запълва блока с текст.

Командата `\framebox` работи точно така, както `\makebox`, но рисува рамка около текста.

Следващият пример показва някои възможности на използването на командите `\makebox` и `\framebox`.

```
\makebox[\textwidth]{%
    ц е н т р и р а н}\par
\makebox[\textwidth][s]{%
    р а з т е г н а т}\par
\framebox[1.1\width]{Сега
съм в рамка!} \par
\framebox[0.8\width][r]{О,
 аз съм твърде дебел} \par
\framebox[1cm][l]{Нищо,
аз също} Можете ли да прочетете това?
```

ц е н т р и р а н

р а з т е г н а т

Сега съм в рамка!

О, аз съм твърде дебел

Нищо, аз също Можете ли да прочетете това?

Сега, когато управляваме хоризонталата, очевидно следващата стъпка е вертикалата. Никакви проблеми! Командата

`\raisebox{`*преместване*`}[`*разтег-над-оснлин*`][`*разтег-под-оснлин*`]{`*текст*`}`

ви позволява да определите вертикалните характеристики на блока. В първите три параметъра могат да се използват `\width`, `\height`, `\depth` и `\totalwidth`, за да се въздейства на размера на блока, вътре в аргумента *текст*.

```
\raisebox{0pt}[0pt][0pt]{\Large%
\textbf{Aaaa\raisebox{-0.3ex}{a}%
\raisebox{-0.7ex}{aa}%
\raisebox{-1.2ex}{a}%
\raisebox{-2.2ex}{a}%
\raisebox{-4.5ex}{a}}}
викал той, но дори стоящият до
него не забелязал, че с него се
е случило нещо ужасно.
```

Aaaaaaaaa викал той, но дори стоящият до него не забелязал, че с него се е случило нещо ужасно.

[1] Съответно ширина, височина, дълбочина и обща височина (височина плюс дълбочина) на текста. — *Бел. ред.*

6.7 Линийки и подпори

Няколко страници по-назад сте забелязали командата

`\rule[`*преместване*`]{`*ширина*`}{`*височина*`}`

При обикновено използване тя генерира просто черен блок.

```
\rule{3mm}{.1pt}%
\rule[-1mm]{5mm}{1cm}%
\rule{3mm}{.1pt}%
\rule[1mm]{1cm}{5mm}%
\rule{3mm}{.1pt}
```

Това е полезно за рисуване на вертикални и хоризонтални линии. Например, линията на заглавната страница е нарисувана с командата `\rule`.

Специален случай е линийката, която няма ширина, но има определена височина. В професионалната текстообработка я наричат *подпора*. Тя се използва, за гарантиране на определена минимална височина на даден елемент на страницата. Бихте могли да я използвате в средата `tabular`, за да сте сигурни, че даден ред има определена минимална височина.

```
\begin{tabular}{|c|}
\hline
\rule{1pt}{4ex}Pitprop \ldots\\
\hline
\rule{0pt}{4ex}Strut\\
\hline
\end{tabular}
```

Pitprop …
Strut

Край.

Библиография

[1] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201-52983-1.

[2] Donald E. Knuth. *The TeXbook,* Volume A of *Computers and Typesetting*, Addison-Wesley, Reading, Massachusetts, second edition, 1984, ISBN 0-201-13448-9.

[3] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-54199-8.[1]

[4] Michel Goossens, Sebastian Rahtz and Frank Mittelbach. *The LaTeX Graphics Companion.* Addison-Wesley, Reading, Massachusetts, 1997, ISBN 0-201-85469-4.[2]

[5] Всеки дистрибутив на LaTeX трябва да съдържа така наречения *LaTeX Local Guide*, обясняващ особеностите на локалната система. Той трябва да се намира във файл, наречен `local.tex`. За съжаление някои лениви администратори не предоставят такъв файл. В такъв случай помолете за помощ местния LaTeX специалист.

[6] LaTeX3 Project Team. *LaTeX 2ε for authors.* Включен в дистрибутива на LaTeX 2ε като `usrguide.tex`.

[7] LaTeX3 Project Team. *LaTeX 2ε for Class and Package writers.* Включен в дистрибутива на LaTeX 2ε като `clsguide.tex`.

[8] LaTeX3 Project Team. *LaTeX 2ε Font selection.* Включен в дистрибутива на LaTeX 2ε като `fntguide.tex`.

[9] D. P. Carlisle. *Packages in the 'graphics' bundle.* Влиза в състава на комплекта 'graphics' като `grfguide.tex`, достъпен в дистрибутива на LaTeX.

[10] Rainer Schöpf, Bernd Raichle, Chris Rowley. *A New Implementation of LaTeX's verbatim Environments.* Влиза в състава на комплекта 'tools' като `verbatim.dtx`, достъпен в дистрибутива на LaTeX.

[11] Vladimir Volovich, Werner Lemberg and LaTeX3 Project Team. *Cyrillic languages support in LaTeX.* Включен в дистрибутива на LaTeX 2ε като `cyrguide.tex`.

[1] Издаден руски превод: М.Гуссенс, Ф.Миттелбах, А.Самарин. *Путеводитель по пакету LaTeX и эго расширению LaTeX 2ε.* Мир, 1999, ISBN 5-03-003325-4.

[2] Издаден руски превод: М.Гуссенс, С.Ратц и Ф.Миттелбах. *Путеводитель по пакету LaTeX и эго графическим расширениям.* Мир, 2002, ISBN 5-03-003388-2.

[12] Graham Williams. *The TeX Catalogue* пълен списък на множество пакети, свързани с TEX и LATEX Достъпен в Интернет на адрес `CTAN:/tex-archive/help/Catalogue/catalogue.html`

[13] Keith Reckdahl. *Using EPS Graphics in LATEX 2ε Documents* обяснява всичко, което бихте искали да знаете за EPS файловете и тяхното използване в LATEX-документите. Достъпен в Интернет на адрес `CTAN:/tex-archive/info/epslatex.ps`

[14] Kristoffer H. Rose. *XY-pic User's Guide.* Достъпен в Интернет на адреса на CTAN заедно с дистрибутива на XY-pic.

[15] John D. Hobby. *A User's Manual for MetaPost.* Достъпен на адрес `http://cm.bell-labs.com/who/hobby/`

[16] Alan Hoenig. *TEX Unbound.* Oxford University Press, 1998, ISBN 0-19-509685-1; 0-19-509686-X (pbk.)

[17] Urs Oswald. *Graphics in LATEX 2ε*, съдържа набор изходни текстове на Java за генерация на произволни окръжности и елипси от средата `picture`, и *MetaPost - A Tutorial.* И двата са достъпни на адрес `http://www.ursoswald.ch`

Азбучен указател