Содержание

4 Таблицы и рисунки

4.1 Плавающие объекты

Большинство публикаций в наши дни содержат множество иллюстраций и таблиц. Эти элементы нуждаются в специальном обращении с ними, так как они не могут быть разбиты между страницами. Одним из выходов было бы начинать новую страницу каждый раз, когда встречается иллюстрация или таблица, слишком большая, чтобы поместиться на текущей странице. Этот подход привел бы к тому, что страницы оставались бы частично пустыми, что смотрится очень плохо.

Для решения этой проблемы любая иллюстрация или таблица, не умещающаяся на текущей странице, может «плавать», перемещаясь на следующую страницу в процессе заполнения текстом текущей. LaTeX предлагает для плавающих объектов два окружения, одно для таблиц и одно для иллюстраций. Чтобы полностью использовать их преимущества, важно примерно представлять, как LaTeX обрабатывает плавающие объекты. Иначе они могут стать источником разочарования из-за того, что LaTeX помещает их не туда, куда вы хотите.

Давайте вначале рассмотрим команды, предоставляемые LaTeX для плавающих объектов.

Любой материал, включенный в окружения `figure` или `table`, трактуется как плавающий. Оба окружения имеют необязательный параметр

```
\begin{figure}[спецификация размещения] или
\begin{table}[спецификация размещения]
```

называемый *спецификацией размещения*. Этот параметр используется для указания LaTeX, куда можно перемещать плавающий объект. *Спецификация размещения* конструируется путем собирания в строчку *ключей размещения плавающего объекта*. См. таблицу 1.

Например, таблицу можно начать следующей строкой:

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

Спецификация размещения `[!hbp]` позволяет LaTeX разместить таблицу прямо по месту (`h`), или внизу той же страницы (`b`), или на выделенной странице (`p`), и все это — даже если это будет смотреться не так уж хорошо (`!`). Если никакой спецификации размещения не задано, стандартные классы предполагают `[tbp]`.

LaTeX размещает каждый встреченный плавающий объект в соответствии с заданной автором спецификацией. Если объект нельзя поместить на текущей странице, он откладывается, помещаясь в очередь иллюстраций или в очередь таблиц.[1] Когда начинается новая страница, LaTeX проверяет, можно ли заполнить специальную страницу плавающими объектами из очередей. Если нет, то

[1] Эти очереди подчиняются дисциплине *fifo*: ‘первым вошел — первым вышел’.

Таблица 1: Ключи размещения плавающего объекта

Ключ	Разрешает помещать объект …
`h`	*здесь же*, в том самом месте текста, где он появился. Обычно используется для маленьких объектов.
`t`	*наверху* страницы
`b`	*внизу* страницы
`p`	на *специальной странице*, содержащей только плавающие объекты.
`!`	не рассматривать большинство внутренних параметров,[a] которые могут предотвратить размещение этого объекта.

[a] Таких, как максимальное число плавающих объектов, разрешенных на одной странице

первый объект из каждой очереди считается только что встретившимся в тексте: LaTeX снова пытается разместить их в соответствии с их спецификациями (за исключением ‘h’, что уже невозможно). Новые встреченные в тексте плавающие объекты помещаются в соответствующие очереди. LaTeX сохраняет порядок, в котором встретились плавающие объекты соответствующего типа. Поэтому иллюстрация, которую не удается разместить, отталкивает все дальнейшие иллюстрации к концу документа. Следовательно:

> Если LaTeX не размещает плавающие объекты, как вы этого ожидаете, то часто это только один объект устроил затор в одной из очередей.

Хотя и возможно задавать в LaTeX конкретную спецификацию размещения плавающего объекта, это может вызвать проблемы. Если объект не помещается в указанном месте, он «застревает», блокируя последующие плавающие объекты. В частности, никогда не используйте ключ `[h]`; это настолько плохо, что в современных версиях LaTeX, он автоматически заменяется `[ht]`.

После объяснения этих механизмов остается еще несколько замечания про окружения `table` и `figure`. Командой

`\caption{`*текст заголовка*`}`

вы можете задать заголовок для объекта. Увеличивающийся номер и строка «Рисунок» или «Таблица» добавляются LaTeX.

Две команды

`\listoffigures` и `\listoftables`

работают аналогично команде `\tableofcontents`, печатая список иллюстраций или таблиц, соответственно. В этих списках заголовки повторяются цели-

ком. Если вы используете длинные заголовки, то вы должны предоставить их краткий вариант для включения в списки. Это делается помещением краткого варианта в квадратные скобки после команды `\caption`.

```
\caption[Короткий]{Дддддллллллиииинннннныыыыыыыыыыыыыййй}
```

При помощи `\label` и `\ref` можно делать ссылки из вашего текста на плавающий объект.

Следующий пример рисует квадрат и вставляет его в документ. Подобную технику можно использовать, чтобы оставить в документе место под изображения, которые вы вставите позже.

```
1  Рисунок~\ref{white} является примером Поп-Арта.
2  \begin{figure}[!hbp]
3  \makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
4  \caption{Пять на пять сантиметров} \label{white}
5  \end{figure}
```

В этом примере LaTeX будет *очень сильно* (`!`) стараться разместить иллюстрацию прямо *по месту* (`h`).[2] Если это невозможно, он попытается разместить ее *внизу страницы* (`b`). Если ему не удастся поместить иллюстрацию на текущей странице, он выяснит, можно ли создать страницу плавающих объектов, содержащую эту иллюстрацию и, возможно, некоторые таблицы из очереди таблиц. Если для отдельной страницы материала еще не накопилось, LaTeX начинает новую страницу и снова рассматривает иллюстрацию, как если бы она только что появилась в тексте.

В определенных случаях может быть необходимо использовать команду

`\clearpage` или даже `\cleardoublepage`

Она указывает LaTeX немедленно разместить все плавающие объекты, остававшиеся в очередях, и затем начать новую страницу. `\cleardoublepage`, помимо этого, начинает новую правостороннюю страницу.

Позже вы узнаете, как включать в ваши документы LaTeX рисунки в формате PostScript, JPG, PNG и TIFF.

4.2 Окружения

Для верстки специальных видов текста LaTeX определяет множество окружений для разных типов форматирования:

`\begin{`*название*`}` *текст* `\end{`*название*`}`

где *название* определяет окружение. Окружения можно вызывать внутри окру-

[2] Предполагая, что очередь иллюстраций пуста

жений, соблюдая порядок вызова и возврата:

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

В следующих разделах рассказывается обо всех важных окружениях.

4.2.1 Список, перечисление и описание

Окружение `itemize` подходит для простых списков, окружение `enumerate` — для нумерованных списков, а окружение `description` — для описаний.

```
\flushleft
\begin{enumerate}
\item Вы можете как угодно
смешивать окружения списков:
\begin{itemize}
\item Но это может смотреться
глупо.
\item[-] С минусом.
\end{itemize}
\item Поэтому помните:
\begin{description}
\item[Глупые] вещи не станут
умнее от помещения в список.
\item[Умные] вещи, однако,
вполне можно представить
списком.
\end{description}
\end{enumerate}
```

1. Вы можете как угодно смешивать окружения списков:
   - Но это может смотреться глупо.
   - С минусом.
2. Поэтому помните:

   Глупые вещи не станут умнее от помещения в список.

   Умные вещи, однако, вполне можно представить списком.

4.2.2 Выравнивание влево, вправо и по центру

Окружения `flushleft` и `flushright` форматируют абзацы, выровненные влево или вправо. Окружение `center` дает центрированный текст. Если вы не используете `\\` для указания разрывов строк, LaTeX определит их автоматически.

```
\begin{flushleft}
Этот текст\\ выровнен влево.
\LaTeX{} не старается сделать
все строки одинаковой длины.
\end{flushleft}
```

Этот текст
выровнен влево. LaTeX не старается сделать все строки одинаковой длины.

```
\begin{flushright}
Этот текст\\ выровнен вправо.
\LaTeX{} не старается сделать
все строки одинаковой длины.
\end{flushright}
```

Этот текст
выровнен вправо. LaTeX не старается сделать все строки одинаковой длины.

```
\begin{center}
В центре\\Земли
\end{center}
```

В центре
Земли

4.2.3 Цитаты и стихи

Окружение `quote` полезно для цитат, важных фраз и примеров.

```
Типографское правило для
длины строки:
\begin{quote}
Обычно строки должны
содержать не больше
66~символов.
\end{quote}

Поэтому \LaTeX{} делает такими
широкими поля страниц. Поэтому
в газетах часто применяют набор
в несколько колонок.
```

Типографское правило для длины строки:

> Обычно строки должны содержать не больше 66 символов.

Поэтому LaTeX делает такими широкими поля страниц. Поэтому в газетах часто применяют набор в несколько колонок.

Существуют еще два похожих окружения: `quotation` и `verse`. Окружение `quotation` полезно для более длинных цитат, охватывающих несколько абзацев, потому что оно начинает абзацы с красной строки. Окружение `verse` используют для стихов, где важны разрывы строк. Строки разделяются при помощи `\\` в конце строки и пустой строки после каждой строфы.

```
Я знаю только одно английское
стихотворение наизусть: про
Шалтая-Болтая:
\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}
```

Я знаю только одно английское стихотворение наизусть: про Шалтая-Болтая:

> 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.

4.2.4 Буквальное воспроизведение

Текст, заключенные между `\begin{verbatim}` и `\end{verbatim}` будет напрямую напечатан, как набранный на пишущей машинке, со всеми пробелами и возвратами каретки, без выполнения каких бы то ни было команд LaTeX.

Внутри абзаца аналогичную функцию выполняет команда

`\verb+`*текст*`+`

Здесь «+» — это только пример символа-ограничителя. Вы можете использовать любой символ, кроме букв, «*» или пробела. Многие примеры на LaTeX в этом буклете набраны этой командой.

```
Команда \verb|\ldots| \ldots

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

Команда `\ldots` …

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

Окружение `verbatim*` отличается от `verbatim` тем, что пробел печатается как символ ␣.

```
\begin{verbatim*}
вариант   окружения
verbatim         со
звездочкой выделяет
пробелы   в   тексте
\end{verbatim*}
```

```
вариант␣␣␣окружения
verbatim␣␣␣␣␣␣␣␣␣со
звездочкой␣выделяет
пробелы␣␣␣в␣␣тексте
```

Команду `\verb` тоже можно использовать аналогичным образом со звездочкой:

```
\verb*|вот   так :-) |
```

```
вот␣␣␣так␣:-)␣
```

Окружение `verbatim` и команду `\verb` нельзя использовать внутри параметров других команд.

4.2.5 Таблицы

Окружение `tabular` используют для верстки таблиц, возможно, с горизонтальными и вертикальными линиями. LaTeX автоматически определяет ширину столбцов.

Аргумент *спецификация* команды

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

определяет формат таблицы. Используйте `l` для столбца текста, выровненного влево, `r` для текста, выровненного вправо и `c` для центрированного текста, `p{`*ширина*`}` для столбца, содержащего выровненный текст с переносом строк, и `|` для вертикальной линии.

Позиция определяет вертикальное положение всего табличного окружения: `t`, `b` и `c` означают выравнивание по верхнему краю, нижнему краю или по центру окружения.

Внутри окружения `tabular` знак «`&`» переходит к следующему столбцу, команда `\\` начинает новую строку, а `\hline` вставляет горизонтальную линию. Вы можете добавлять неполные линии при помощи команды `\cline{j-i}`, где `j` и `i` — номера столбцов, над которыми должна проходить линия.

```
\begin{tabular}{|r|l|}
\hline
7C0 & шестнадцатеричное \\
3700 & восьмеричное \\ \cline{2-2}
11111000000 & двоичное \\
\hline \hline
1984 & десятичное \\
\hline
\end{tabular}
```

7C0	шестнадцатеричное
3700	восьмеричное
11111000000	двоичное
1984	десятичное

```
\begin{tabular}{|p{4.7cm}|}
\hline
Добро пожаловать в абзац в
рамочке. Надеемся, вам всем
тут понравится.\\
\hline
\end{tabular}
```

Добро пожаловать в абзац в рамочке. Надеемся, вам всем тут понравится.

Разделитель столбцов можно задать конструкцией `@{...}`. Эта команда удаляет пробел между столбцами и заменяет его на то, что включено в фигурные скобки. Одно из частых использований этой команды показано ниже, при рассказе о проблеме выравнивания по десятичной точке. Другое возможно использование — для подавления ведущего пробела в таблице при помощи `@{}`:

```
\begin{tabular}{@{} l @{}}
\hline
нет ведущего пробела\\
\hline
\end{tabular}
```

нет ведущего пробела

```
\begin{tabular}{l}
\hline
ведущий пробел слева и справа\\
\hline
\end{tabular}
```

ведущий пробел слева и справа

Поскольку встроенный способ выровнять числовые столбцы по десятичной точке отсутствует,[3] мы можем «обмануть» TEX и добиться этого при по-

[3] Если на вашей системе установлен комплект ‘tools’, обратите внимание на пакет dcolumn.

мощи двух столбцов: выровненной вправо целой части и выровненной влево дробной. Команда `@{.}` в строке `\begin{tabular}` заменяет нормальный пробел между столбцами просто на «.», давая эффект одного столбца, выровненного по десятичной точке. Не забудьте заменить в ваших числах точку на разделитель столбцов (&)! Метку столбца можно поместить над нашим числовым «столбцом» командой `\multicolumn`:

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

Выражение с π	Значение
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}
```

Ene	
Mene	Muh!

Материал, набираемый в окружении `tabular`, всегда верстается на одной странице. Если вам нужно набирать длинные таблицы, используйте окружения supertabular, longtable или xtab.

4.3 Графика и цвет

Встраивание графических изображений, созданных другими приложениями, обеспечивают пакеты из коллекции graphics, написанной Дэвидом Карлайлом (Carlisle, David). Кроме импортирования рисунков эти пакеты также обеспечивают работу с цветом, позволяют производить вращение или изменять масштаб любого бокса, будь то рисунок, таблица или отдельная буква.

При загрузке любого графического пакета нужно выбрать драйвер устройства, которое предполагается использовать для печати документа или, в более широком плане, для вывода документа на выходное устройство. Драйвер указывается в необязательном аргументе декларации `\usepackage` в преамбуле

входного файла. Например,

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

загружает пакеты graphicx и color с драйвером dvips, а

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

предполагает использование драйвера pdftex. В данном контексте драйвером называется опция, определяющая набор инструкций для преобразования команд LaTeX'а в команды, которые управляют работой программы-драйвера конкретного выходного устройства.

Программа `dvips` долгое время имела слишком очевидные преимущества перед другими приложениями, что и обеспечило её преобладающее распространение. Она единственная поддерживала все графические возможности пакетов из коллекции `graphics`. Соответственно только драйвер `dvips` был реализован для всех типов компьютеров и для всех операционных систем. Поэтому сейчас новые драйверы обычно обладают всеми возможностями `dvips`, позволяя загружать графические пакеты с опцией `dvips`.

Небольшой диссонанс внесло изобретение переносимого формата документов Portable Document Format, или просто PDF. С появлением компилятора pdflatex стала реальностью прямая компиляция документов с разметкой LaTeX в формат PDF, но в таком случае при загрузке графических пакетов следует указывать опцию pdftex вместо dvips.

Кстати, а что произойдёт, если при загрузке графического пакета драйвер вообще не указан? Вопрос вполне разумный, ведь драйвер указывается в необязательном аргументе. Ответ содержится в конфигурационных файлах, где указан драйвер, используемый по умолчанию. Конфигурационные файлы графических пакетов graphics.cfg и color.cfg хранятся в одном из каталогов, где компилятор ищет необходимую ему информацию. Графические пакеты сконфигурированы так, что компилятор latex выбирает опцию dvips, а компилятор pdflatex выбирает опцию pdftex. Таким образом, лучше всего вообще не указывать, какой именно нужен драйвер.

Кроме того, если вы используете компилятор XeLaTeX можно вообще не указывать загрузку пакета `graphicx`, так как он загружается автоматически.

Импортирование изображений, созданных специализированными графическими программами, осуществляется с помощью команды `\includegraphics`. В простейшем варианте, чтобы включить в печатный документ рисунок, записанный в графическом файле `gr-file`, достаточно в нужное место входного файла вставить команду `\includegraphics{gr-file}`, указав в её аргументе имя файла.

В команде `\includegraphics` есть один обязательный параметр — вставляемая картинка.

Необязательные параметры передаются с помощью пар «ключ»=«значение», разделяемых запятой. За подобный способ объявления параметров отвечает пакет keyval.

Некоторые из поддерживаемых пакетом параметров перечислены далее.

angle — поворачивает картинку на указанный угол в градусах.

origin — определяет координаты центра, вокруг которого вращается рисунок. Кроме непосредственно координат origin принимает и буквенные сокращения: l, b, r и t — соответствует центру вращения слева, снизу, справа и сверху. В этом случае выбирается середина указанной стороны. Возможны комбинации, задающие углы картинки: lt, rt, rb и lb. c — означает центр картинки.

width — ширина вставляемой картинки.

height — высота вставляемой картинки.

scale — масштабный коэффициент.

keepaspectratio — логический переключатель. Модифицирует параметры высоты и ширины картинки в сторону уменьшения с целью сохранения естественных пропорций картинки.

Демонстрацию возможностей применения команды `\includegraphics` можно посмотреть по ссылке.

4.4 Картинки «в оборку»

Маленькие иллюстративные рисунки удобно делать в оборку с текстом, то есть текст должен обтекать их. Такие картинки располагаются на внешней стороне страницы, то есть слева для чётных, а справа для нечётных страниц или в случае одностороннего режима печати.

Традиционно разбираются два пакета для создания подобных рисунков — это floatflt и wrapfig. Заметим, что floatflt более автоматизирован для размещения картинок, но он также чаще «ломается» при большом числе плавающих объектов. Возможны даже «потери» картинок. Эти пакеты определяют окружения floatingfigure и wrapfigure соответственно.

```
\begin{floatingfigure}[«размещение»]{«ширина»}
...
\end{floatingfigure}
```

Необязательный параметр «размещение» окружения floatingfigure позволяет изменить алгоритм размещения картинки: rflt — размещать справа, lflt — размещать слева, vflt — слева для чётных и справа для нечётных страниц (по умолчанию).

Окружение wrapfigure имеет следующую структуру:

```
\begin{wrapfigure}[«число строк в оборке»]
{«размещение»}{«ширина»}
...
\end{wrapfigure}
```

В отличие от floatingfigure окружение wrapfigure требует определить правила размещения картинки в обязательном порядке. Доступные варианты:

r — размещать справа,

l — размещать слева,

i — размещать с внешней стороны страницы,

o — размещать с внутренней стороны страницы.

Если вместо строчных букв передать заглавные, то включается запрет на сдвиг по вертикали — картинка должна быть размещена, начиная с той строки абзаца, в которой она была определена. Необязательный параметр «число строк в оборке» позволяет указать число строк текста, которые должны быть сбоку от картинки. При этом выносная фор- мула считается за три строки текста. Если параметр не определён, то число строк вычисляется автоматически, к сожалению, не всегда оптимально. Свою процедуру размещения картинки в оборку с текстом предлагает также и пакет nccfloats из коллекции ncctools, созданной А.И. Роженко:

```
\sidefig(«ширина картинки»)(«ширина текста»)
{\includegraphics{«картинка»}}{«текст»}
```

В этом случае предлагается передавать команде `\sidefig` и саму картинку и текст, помещаемый сбоку. Параметр «ширина текста» можно опустить. Тогда текст занимает всё оставшееся пространство. Подробности в `nccfloats.pdf`.

4.5 Настройка подписей плавающих объектов с помощью пакета caption

Для добавление подписи к рисунку используется команда `\caption`, которую можно применять только внутри плавающих объектов. В качестве обязательного параметра передаётся текст подписи.

Внутри стандартных классов LaTeX'а подписи не получили заслуженного внимания. Они набираются как обычный абзац и внешне не отличаются от основного текста.

При выводе подпись центрируется, если она достаточно мала. В противном случае подпись оформляется в виде абзаца. Текст подписи не должен содержать команд разрыва строки. Все хрупкие команды внутри подписи должны быть защищены с помощью команды `\protect`. `\caption` можно передать

также необязательный параметр, который должен представлять собой краткую версию подписи, появляющуюся в автоматически создаваемых списках.

Оформление подписи жёстко привязано к стилю документа, и изменить её без переопределения самой команды `\caption` не просто. Для модификации пара- метров следует воспользоваться пакетами `caption` или `ccaption`. Документация в `caption.pdf` и `ccaption.pdf` соответственно.

При включении русской локализации `\usepackage[russian]{babel}` перед подписью выводится слово «Рис.», за которым идёт автоматически вычисляемый порядковый номер картинки. В качестве разделителя между счётчиком и подписью по умолчанию используется двоеточие. Для замены двоеточия на точку в преамбуле достаточно набрать, например, следующее:

```
\usepackage{ccaption}
% заменяем для рисунков ’: ’ после номера рисунка на ’. ’
\captiondelim {. } % после точки стоит пробел !
```