

Основы ОС Linux/UNIX для пользователей

Copyright © 2010 Садов О.Л. Данное руководство может свободно использоваться и распространяться на условиях, оговоренных в Open Publication License, v1.0, доступной по следующему ресурсу <<http://www.opencontent.org/openpub/>>

Содержание

I	Начальные сведения	7
1	Процедура регистрации в системе	9
1.1	Вход в систему	9
1.2	Смена пароля	9
1.3	Выход из системы	10
2	Командные интерпретаторы и переменные среды	11
2.1	Понятие «командного интерпретатора»	11
2.2	Переменные среды	12
2.3	Специальные символы Shell	15
2.3.1	Обработка потоков информации (программные конвейеры, перенаправление ввода/вывода)	15
2.3.2	Метасимволы shell	15
2.3.3	Другие спецсимволы shell	16
2.4	Командные последовательности, упрощающие работу с командной строкой	16
2.4.1	alias	17
2.4.2	history	18
2.4.3	Jobs	18
2.5	Стартовые файлы оболочки	19
II	Основные команды и программы ОС UNIX/Linux	20
3	Поиск справочной информации	22
3.1	Системное руководство man	22
3.1.1	Команда man	22
3.1.2	Справочная система whatis	23
3.1.3	Команда argopos	24
3.1.4	Справочная система info	24
4	Информационные команды	26
4.1	Команда uname	26
4.2	Команда date	26
4.3	Команда cal	27
4.4	Команда id	27
4.5	Команда logname	27
5	Работа с другими пользователями	29
5.1	Команда who	29
5.2	Команда write	29
6	Работа с терминальными линиями	30
6.1	Команда tty	30
6.2	Команда stty	30
7	Работа с файлами и каталогами	32
7.1	Работа с файловой системой	32

7.1.1	Команда ls	32
7.1.2	Команда pwd	33
7.1.3	Команда cd	33
7.1.4	Команда cp	33
7.1.5	Команда ln	34
7.1.6	Команда mv	35
7.1.7	Команда rm	35
7.1.8	Команда mkdir	36
7.1.9	Команда rmdir	36
7.2	Изменение атрибутов файлов	37
7.2.1	Команда chmod	37
7.2.2	Команда chown	38
7.3	Просмотр содержимого файлов	38
7.3.1	Команда cat	38
7.3.2	Команда more	39
7.3.3	Команда less	40
7.3.4	Команда pg	40
7.3.5	Команда head	41
7.3.6	Команда tail	41
7.3.7	Команда tee	42
7.3.8	Команда split	42
7.3.9	Команда wc	42
7.3.10	Команда od	43
7.4	Поиск информации в файлах	44
7.4.1	Команды grep, egrep	44
7.5	Команды архивации, компрессии и декомпрессии	44
7.5.1	Команда tar	44
7.5.2	Команда cpio	46
7.5.3	Команды gzip и gunzip	46
7.6	Подсчет контрольных сумм	47
7.6.1	Команда sum	47
7.6.2	Команда md5sum	47
7.7	Команды доступа к принтеру	48
7.7.1	Команда lpr	48
7.7.2	Команда lpq	48
7.7.3	Команда lprm	48
7.7.4	Команда lp	48
7.7.5	Команда cancel	49
7.7.6	Команда lpstat	49
7.8	Поиск и проверка состояния файлов	50
7.8.1	Команда find	50
7.8.2	Команды which и type	51
7.8.3	Команда file	51
7.8.4	Команда test	52
8	Управление задачами	53
8.1	Команда ps	53
8.2	Команда jobs	54
8.3	Команда bg	55
8.4	Команда fg	55
8.5	Команда kill	55
9	Отложенной исполнение команд	56
9.1	Команда sleep	56

9.2	Команды at и batch	56
9.3	Команда atq	57
9.4	Команда atrm	57
9.5	Команда crontab	57
10	Текстовые редакторы	59
10.1	Команда ed	59
10.2	Команда vi	60
10.3	Команда emacs	61
10.4	Команда joe	62
11	Обработка текстовой информации	64
11.1	Команда sed	64
11.2	Команда awk	64
11.3	Команда cmp	66
11.4	Команда diff	66
11.5	Команда fold	66
11.6	Команда sort	67
11.7	Команда uniq	67
11.8	Команда tr	68
11.9	Команда join	68
11.10	Команда paste	69
12	Программирование в среде командных интерпретаторов	70
12.1	Запуск командных файлов	70
12.2	Позиционные параметры	71
12.3	Основные конструкции и операторы командного языка	71
12.3.1	Условный интерпретатор if	71
12.3.2	Оператор переключения	71
12.3.3	Оператор цикла while	72
12.3.4	Оператор цикла for	72
12.3.5	Оператор цикла until	72
12.3.6	Вычисление выражений	72
12.3.7	Определение подпрограмм	73
12.4	Встроенные функции	73
12.4.1	basename — выделение компонентов имени файла	73
12.4.2	dirname — выделение имени каталога	73
12.4.3	echo — вывести на печать	73
12.4.4	eval — выполнить команду	73
12.4.5	exec — запустить программу на исполнение	73
12.4.6	espr — вычисление арифметических выражений	73
12.4.7	read — прочесть строку	74
12.4.8	readonly — защита переменных от изменения	74
12.4.9	shift — сдвиг позиционных параметров	74
III	Сетевое окружение ОС Linux/UNIX	75
13	Утилиты из набора SSH (Secure Shell)	77
13.1	Вход на удаленный хост — ssh	77
13.2	Обмен данными с удаленным хостом — scp	77
14	Традиционные сетевые утилиты	79
14.1	Вход на удаленный хост — telnet	79

14.2	Обмен файлами — ftp	80
14.3	Отправка и чтение электронной почты — mail	81
15	Средства получения информации из сети Интернет	82
15.1	Текстовый веб-браузер — lynx	82
15.2	wget	83
15.3	lftp	83
15.4	rsync	84
IV	Графическая среда ОС Linux/UNIX	86
16	Основные понятия X-Window	88
16.1	Устройства ввода	88
16.2	Понятие DISPLAY	88
16.3	Шрифты X-Window	88
16.4	Цвета X-Window	89
16.5	Ресурсы X-Window	89
16.6	Стандартные опции Xt	90
17	Базовые приложения X-Window	91
17.1	Xserver	91
17.2	Window Manager	92
17.3	Эмулятор терминала — xterm	92
17.4	Уничтожение окон программ-клиентов — xkill	93
17.5	Получение информации об окне — xwininfo	93
17.6	Получение списка клиентов — xlsclients	94
17.7	Получение информации о клиенте — xdpwininfo	95
17.8	Список шрифтов — xlsfonts	97
17.9	Утилита выбора шрифтов — xfontsel	98
17.10	Средство просмотра шрифтов — xfd	98
17.11	Получение списка шрифтов — showrgb	98
17.12	Список текущих ресурсов клиента — appres	99
17.13	Манипуляции с ресурсами — xrdb	100
17.14	Редактор ресурсов — editres	100
17.15	Установка свойств дисплея и устройств ввода — xset	101
17.16	Установка свойств корневого окна — xsetroot	102
17.17	Отслеживание событий — xev	102
17.18	Установка раскладок клавиатуры и мыши — xmodmap	102
17.19	Установка параметров клавиатуры — setxkbmap	103
17.20	Обновление экрана — xrefresh	104
	Предметный указатель	105

Часть I

Начальные сведения

История Unix-систем начинается в 1970 г., когда Кен Томпсон (Ken Thompson) и Денис Ритчи (Dennis Ritchie) работавшие в AT&T Bell Laboratories в свободное время создали многозадачную многопользовательскую операционную систему, написанную на языке высокого уровня C. Система распространялась в исходных кодах среди университетов за символическую плату, что послужило взрывному росту ее популярности в 80-е годы прошлого столетия. Практически все разработчики новых компьютерных систем, начиная с этого периода, использовали UNIX как базовую платформу для своих новых разработок.

Коммерциализация рынка UNIX-систем и переход на закрытую модель разработки и распространения привели к созданию альтернативного движения по разработке набора программ, аналогичных набору утилит, стандартно входящих в UNIX — GNU (самоссылающаяся аббревиатура "GNU is Not Unix") проект. В 1991 году финский студент Линус Торвальдс (Linus Torvalds) создал собственное ядро операционной системы, совместимое по программным интерфейсам с ОС UNIX, получившее название Linux. Ядро Linux в сочетании с набором утилит проекта GNU послужили основой для создания полноценной ОС, сравнимой по возможностям с коммерческими UNIX-системами, а подчас и превосходящей их.

Глава 1

Процедура регистрации в системе

1.1 Вход в систему

Вход в систему начинается с системного приглашения:

```
login:
```

В ответ на приглашение следует ввести идентификатор пользователя, зарегистрированного в системе, и нажать клавишу **[ENTER]**.

Затем система запрашивает пароль пользователя:

```
login: guest
```

```
Password:
```

Ввод пароля также необходимо завершить нажатием клавиши **[ENTER]**. Вводимые символы при вводе на экране не отображаются из соображений секретности. При совпадении идентификатора и пароля с зарегистрированными в системе¹, появляется приглашение командного интерпретатора в случае обычной терминальной сессии или открывается графическая сессия.

1.2 Смена пароля

Для смены своего системного пароля² достаточно набрать команду **passwd**, затем дважды ввести новый пароль³ в ответ на приглашение. Пароль при этом не отображается.

```
$ passwd
Changing password for user guest.
Changing password for guest
(current) UNIX password:
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

¹После инсталляции системы в ней присутствует по крайней мере один пользователь: **root**, являющийся суперпользователем с максимально возможными полномочиями. Его пароль устанавливается при инсталляции.

²После первого входа в систему надо сменить пароль, выданный вам системным администратором, на новый, кроме того, пароль полезно менять не реже 1 раза в 2 месяца.

³Пароль должен состоять, по крайней мере, из 6 символов, в которые полезно включать буквы, набранные в разных регистрах, цифры и не специальные символы.

При несоответствии двух вариантов пароля друг другу система выдает сообщение об ошибке, и пароль не изменяется.

1.3 Выход из системы

Завершение пользовательского сеанса осуществляется при помощи команды **exit**, или **logout**, или же просто одновременным нажатием клавиш **[Ctrl]-[D]** (если эта возможность специально не отключена) при работе в терминальной сессии или средствами графической оболочки.

Глава 2

Командные интерпретаторы и переменные среды

2.1 Понятие «командного интерпретатора»

С самого момента создания ОС UNIX разработчики поместили пользовательский интерфейс, назвав его **shell** (оболочка), в пространство пользовательских процессов и снабдили его компактным, но мощным набором средств для общения с ядром и утилитами ОС. Несмотря на наличие большого количества мощных графических интерфейсов, появившихся в UNIX, командная строка является важнейшим средством для общения с операционной системой.

Все команды, набираемые в строке, могут быть использованы в командных файлах, исполняемых интерпретатором **shell**, и наоборот. Действия, выполняемые в командном интерпретаторе, в дальнейшем могут быть окружены графической оболочкой, и детали их исполнения, таким образом, окажутся скрыты от конечного пользователя.

При каждом входе в систему пользователь попадает в среду так называемого домашнего интерпретатора пользователя, выполняющего настроечные действия для пользовательской сессии и в дальнейшем осуществляющего интерактивное общение с пользователем. Выход из пользовательской сессии завершает работу интерпретатора и порожденных из него процессов. Любому пользователю может быть назначен любой из существующих в системе интерпретаторов или же интерпретатор собственной разработки. На данный момент существует целый набор командных интерпретаторов, способных быть пользовательской оболочкой и средством исполнения командных файлов:

- **sh** — Bourne-Shell, исторический и концептуальный предок всех остальных командных интерпретаторов, разработанный Стивеном Борном (Stephen Bourne) в AT&T Bell Labs.
- **csh** — C-Shell, интерпретатор, разработанный в университете Беркли Биллом Джоном (Bill Joy) для системы 3BSD с C-подобным синтаксисом управляющих операторов. Обладает расширенными средствами для работы в интерактивном режиме, средствами управления заданиями, но работа с командными файлами не очень корректна.
- **ksh** — Korn-Shell, интерпретатор, разработанный Дэвидом Корном (David Korn) и стандартно поставляемый с SYSV. Программно совместим с Bourne-Shell, включает в себя средства редактирования командных строк. Набор средств, предоставляемый Korn-Shell, зафиксирован в качестве стандарта командного языка в POSIX P1003.2.

Кроме вышеперечисленных оболочек, стандартно поставляемых с каждой коммерческой системой, существует некоторое количество интерпретаторов, свободно распространяемых в исходных кодах:

- **bash** — Bourne-Again-Shell, в достаточной степени совместимый с Bourne-Shell, включающий в себя как средства интерактивной работы, предложенные в C-Shell, так редактирование командных строк.
- **tcsh** — Tenex-C-Shell, дальнейшее развитие C-Shell с расширенным интерактивным интерфейсом и несколько улучшенным программным.
- **zsh** — Z-Shell, включает в себя все наработки Bourne-Again-Shell и Tenex-C-Shell, а также некоторые существенные их расширения (впрочем, не так популярен, как последние).
- **pdksh** — Public-Domain-Korn-Shell, свободно распространяемый аналог Korn-Shell с некоторыми дополнениями.

Для облегчения работы пользователей, не привыкших к работе с командной строкой, существует ряд свободно распространяемых интерфейсов, например, Midnight Commander (mc), напоминающий Norton Commander, или файловые менеджеры графических интерфейсов, напоминающие Explorer MS Windows.

2.2 Переменные среды

Операционная система поддерживает специальный вид ресурсов, называемых *переменные среды* (*environment variables*). Эти переменные представляют собой пару **ИМЯ - ЗНАЧЕНИЕ**. Имя может начинаться с буквы и состоять из букв, цифр и символов подчеркивания.

Для подстановки значения переменной в командную строку перед именем переменной ставится знак \$:

```
$ echo $USER
guest
```

В случае, если переменная не установлена, возвращается пустая строка.

Для установки значения переменной используется оператор присваивания (в случае Bourne-подобных оболочек):

```
$ TEST=test
```

или встроенный оператор **set** (в случае C-подобных):

```
$ set TEST=test
```

Команда **set** без аргументов выводит список значений всех переменных, установленных в среде:

```
$ set
COLUMNS=197
CVS_RSH=ssh
DIRSTACK=()
EUID=1000
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/home/guest/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/guest
HOSTNAME=myhost
HOSTTYPE=i686
IFS=$' \t\n'
INPUTRC=/etc/inputrc
```

```

KDEDIR=/usr
KDEDIRS=/home/guest/.local/
KDE_IS_PRELINKED=1
KDE_NO_IPV6=1
LANG=ru_RU.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=65
LOGNAME=guest
....

```

Переменные могут носить характер локальный для данного процесса или глобальный для сессии. Задать локальные значения для переменных можно, предварив ими вызов команд:

```

$ TEST=test1 sh -c 'echo $TEST'
test1

```

Оценить содержимое набора переменных для сессии можно, вызвав встроенную команду интерпретатора **env**, в случае Bourne-подобных интерпретаторов (sh, ksh, bash, zsh, pdksh...), и **printenv** в случае использования интерпретаторов клона C-Shell (csh, tcsh...):

```

$ env
HOSTNAME=myhost
TERM=xterm
SHELL=/bin/bash
HISTSIZE=1000
KDE_NO_IPV6=1
SSH_CLIENT=172.16.0.9 50487 22
QTDIR=/usr/lib/qt-3.3
QTINC=/usr/lib/qt-3.3/include
SSH_TTY=/dev/pts/6
USER=guest
MOZILLA_CERTIFICATE_FOLDER=/home/guest/.evolution/
KDEDIR=/usr
MAIL=/var/spool/mail/guest
PATH=/usr/games:/usr/local/bin:/bin:/usr/bin:/home/guest/bin
INPUTRC=/etc/inputrc
PWD=/home/guest
KDE_IS_PRELINKED=1
LANG=ru_RU.UTF-8
KDEDIRS=/home/guest/.local/
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
SHLVL=1
HOME=/home/guest
LOGNAME=guest
QTLIB=/usr/lib/qt-3.3/lib
CVS_RSH=ssh
SSH_CONNECTION=172.16.0.9 50487 172.16.2.9 22
LESSOPEN=|/usr/bin/lesspipe.sh %s
G_BROKEN_FILENAMES=1
_=/bin/env

```

Наборы команд **Shell** могут компоноваться в командные файлы, называемые скриптами, где в первой строке в специального вида комментарии указывается командный интерпретатор для исполнения этого набора. Например, создадим в текстовом редакторе файл с названием **test**, следующего содержания:

```
#!/bin/sh
```

```
echo Переменная TEST:
echo $TEST
```

Данная программа будет выводить на стандартный вывод текстовое сообщение «Переменная TEST:» и значение переменной TEST, если оно задано. Запустить его из командной строки можно, передав его в качестве параметра командному интерпретатору:

```
$ sh test
Переменная TEST:
```

Придать переменной глобальный характер можно при помощи оператора **export** (Bourne) или **setenv** (C-Shell):

```
$ export TEST=test1
$ sh test
Переменная TEST:
test1
```

Задать локальные значения переменных для выполнения данной программы можно, предварив ими вызов команд:

```
$ TEST=test2 sh test
Переменная TEST:
test2
```

Удаление переменных среды производится при помощи оператора **unset**:

```
$ unset TEST
```

Наиболее важные переменные:

- SHELL — домашняя оболочка.
- PATH — список каталогов, просматриваемых при поиске исполняемых файлов.
- MANPATH — список каталогов, просматриваемых при поиске файлов системного руководства **man** (см. [Разд. 3.1.1](#)).
- IFS — разделители полей.
- LPDEST — принтер, используемый по умолчанию, если данная переменная не установлена, используются установки системы.
- EDITOR — редактор, используемый по умолчанию.
- VISUAL — режим редактирования командной строки.
- PS1¹ или **prompt**² — первичное приглашение **shell**, выдаваемое в поток стандартного вывода в интерактивном режиме.
- PS2 — вторичное приглашение, выдаваемое в поток стандартного вывода в интерактивном режиме при вводе символа перевода строки в незавершенной команде.
- TERM — тип используемого терминала.
- PAGER — команда, используемая **man** для просмотра страниц руководства.
- TZ — часовой пояс.
- LINES — количество строк, помещающихся на экране.

¹Клон Bourne-Shell. В Bourne-Shell по умолчанию обычно \$ или #.

²Клон C-Shell. В C-Shell по умолчанию обычно %.

- COLUMNS — количество символов, помещающихся в столбце.
- HOME — домашний каталог, используемый, в частности, командой **cd** (см. Разд. 7.1.3).
- LOGNAME — ваше входное имя.

2.3 Специальные символы Shell

2.3.1 Обработка потоков информации (программные конвейеры, перенаправление ввода/вывода)

Стандартом для UNIX-команд является чтение информации из потока стандартного ввода (по умолчанию — клавиатура текущего терминала), запись — в стандартный вывод (экран терминала) и перенаправление ошибок — в стандартный поток ошибок (также экран терминала), если в параметрах команды не указано чего-либо иного. Эти умолчания могут быть переназначены средствами **shell**'а.

Таблица 2.1 Способы перенаправления

Команда	Назначение
< файл	Перенаправить стандартный ввод из <i>файла</i>
> файл	Перенаправить стандартный вывод в <i>файл</i>
<< метка_конца	Далее следует «встроенный документ», оканчивающийся <i>меткой_конца</i>
>> файл	Вывод будет дописываться в конец файла
поток<файл	Перенаправление ввода <i>потока</i>
поток>файл	Перенаправление вывода потока
поток>&поток файл	Слияние потоков
поток>>&поток файл	Слияние потоков и дописывание в конец <i>файла</i>
 программа	«Конвейер» — перенаправление стандартного вывода исполняемой команды на стандартный ввод <i>программы</i>

2.3.2 Метасимволы shell

Набор метасимволов, используемый **shell**, имеет для UNIX универсальное значение — с некоторыми расширениями он также используется текстовыми редакторами, программами поиска и т.д. Этот расширенный набор поддерживается свободно распространяемой библиотекой **regex**, которая встраивается во многие приложения GNU-проекта.

Таблица 2.2 Метасимволы shell

Метасимвол	Назначение
*	Любое количество символов (возможно 0)
?	Один символ
+	Один символ и более
[набор_символов]	Любой из <i>набора</i>
[!набор_символов]	Ни один из <i>набора</i>

Таблица 2.3 Способы перенаправления

Спецсимвол	Назначение
;	Ограничитель команды
&	Запуск команды в фоновом режиме
команда1&&команда2	В случае удачного завершения <i>команды1</i> выполнить <i>команду2</i>
команда1 команда2	В случае неудачного завершения <i>команды1</i> выполнить <i>команду2</i>
'строка'	Подставить строку, не делая подстановку переменных
"строка"	Подставить строку, делая подстановку переменных
'команда'	Подставить стандартный вывод команды в командную строку
\	Экранирующий символ или символ продолжения команды на следующей строке

2.3.3 Другие спецсимволы shell

2.4 Командные последовательности, упрощающие работу с командной строкой

Простейшие средства манипуляции командной строкой предоставляются на уровне драйвера терминала. Они могут быть переустановлены при помощи команды **stty**. В Табл. 2.4 показаны комбинации клавиш для наиболее необходимых операций:

Таблица 2.4 Комбинации клавиш для наиболее необходимых операций

Операция	Назначение	Комбинация клавиш
erase	Стереть символ	обычно [Ctrl]-[H] , или [Ctrl]-[?] , или просто [Backspace] или [Delete]
werase	Стереть слово	обычно [Ctrl]-[W]
kill	Стереть строку	обычно [Ctrl]-[U]
rprnt	Перевывести текущую строку	обычно [Ctrl]-[R]
intr	Послать запущенному процессу сигнал завершения	обычно [Ctrl]-[C] или [Delete]
quit	Прервать текущий процесс с образованием core dump'a памяти	обычно [Ctrl]-[\]
stop	Остановить вывод текущего процесса на терминал	обычно [Ctrl]-[S]
start	Продолжить остановленный вывод процесса	обычно [Ctrl]-[Q]
eof	Символ окончания файла	обычно [Ctrl]-[D]
susp	Остановить текущий процесс	обычно [Ctrl]-[Z]

Командный интерпретатор Bourne-Shell послужил отличным примером разумного разграничения встроенных интерпретаторов и внешних утилит, но, к сожалению, не обладал средствами, облегчающими пользователю интерактивную работу в командной строке. Начиная с C-Shell, такие улучшения постепенно стали вводиться. Были введены механизмы **alias**, **history** и **job-control**.

Начиная с интерпретатора Korn-Shell, был введен механизм редактирования командных строк путем эмуляции командных последовательностей двух наиболее распространенных в мире UNIX'a текстовых редакторов: **vi** и **emacs**. Установка в эти моды происходит при вызове **shell'a** с опцией **-o vi** или **-o emacs** или при установке переменной среды **VISUAL**.

Основные командные последовательности **vi**-моды:

- **[ESC]** — вход и выход из командного режима
- **[j]** — предыдущая команда в **history**
- **[k]** — последующая команда в **history**
- **[^]** — переход в начало строки
- **[\$]** — переход в конец строки
- **[l]** — передвинуться на символ вперед
- **[h]** — передвинуться на символ назад
- **[x]** — удалить один символ
- **[dw]** — удалить одно слово
- **[D]** — удалить все до конца строки

Основные командные последовательности **emacs**-моды:

- **[Ctrl]-[P]** — предыдущая команда в **history**
- **[Ctrl]-[N]** — последующая команда в **history**
- **[Ctrl]-[A]** — переход в конец строки
- **[Ctrl]-[E]** — переход в конец строки
- **[Ctrl]-[F]** — передвинуться на символ вперед
- **[Ctrl]-[B]** — передвинуться на символ назад
- **[Ctrl]-[D]** — удалить один символ
- **[Meta]-[D]** — удалить одно слово
- **[Ctrl]-[K]** — удалить все до конца строки

Во всех свободно распространяемых интерпретаторах, а также в версиях Korn-Shell некоторых коммерческих производителей для передвижения по командной строке можно использовать клавиши управления курсором.

Начиная с Korn-Shell, в интерпретаторе стала поддерживаться возможность «дописывания» имен в командной строке. Клавиатурной комбинацией для «дописывания» является **[ESC]-[ESC]**, в некоторых **shell**'ах — **[Tab]**. Первый аргумент командной строки является исполняемой командой, и поэтому его поиск производится согласно переменной `$PATH`. Остальные ищутся по файловой структуре при задании пути. Некоторые интерпретаторы (Tenex-C-Shell, Z-Shell) могут помочь и в наборе аргументов команд.

2.4.1 alias

Механизм псевдосимволов **alias** позволяет назначать более удобные заменители для наиболее необходимых команд. C-Shell:

```
% alias la ls -a
% alias ll ls -al
% alias bye exit
```

Клон Bourne-Shell:

```
% alias la='ls -a'
% alias ll='ls -al'
% alias bye='exit'
```

Команда **alias** без аргументов выдает список назначенных псевдонимов:

```
% alias
bue  exit
la   (ls -a)
ll   (ls -al)
```

Отменить назначение псевдонима можно при помощи команды **unalias**:

```
$ unalias la
$ alias
ll='ls -al'
bye='exit'
```

2.4.2 history

Использовать ранее введенные команды можно при помощи **history**:

```
% history
1  alias la ls -a
2  alias ll ls -al
3  alias bye exit
4  alias
5  unalias la
6  alias
```

В клоне C-Shell, а также в Bourne-Again-Shell, Z-Shell набранную ранее команду можно вызвать опять, набрав **!номер команды в history**:

```
% !4
alias
bye  exit
la   (ls -a)
ll   (ls -al)
```

2.4.3 Jobs

Начиная с интерпретатора Korn-Shell, был введен механизм управления задачами. Каждая задача может быть либо сразу запущена интерпретатором в фоновом режиме, либо остановлена комбинацией **[Ctrl]-[Z]**, а впоследствии переведена в фоновый (**bg 'номер задачи'**) или интерактивный (**fg 'номер задачи'**, или **%'номер задачи'**) режим. Текущее состояние задач можно оценить при помощи команды **jobs**. При помощи встроенного оператора **kill** процессу может быть послан сигнал (например, для его завершения).

```
$ xterm &
[1] 27259
$ xeyes

[2]+  Stopped                  xeyes
$ jobs
```

```
[1]-  Running          xterm &
[2]+  Stopped         xeyes
$ bg
[2]+  xeyes &
$ jobs
[1]-  Running          xterm &
[2]+  Running         xeyes &
$ fg
xeyes

$ jobs
[1]+  Running          xterm &
```

2.5 Стартовые файлы оболочки

В качестве стартовых файлов, запускаемых на исполнение при входе пользователя в систему, используются файлы:

- Bourne: `.profile`
- Z-Shell: `.zprofile`
- C-Shell: `.login`

Для инициализации **shell**'ов, порожденных вторично, используются:

- csh: `.cshrc`
- bash: `.bashrc`
- tcsh: `.tcshrc`
- zsh: `.zshrc`

При выходе из сессии вызывается на исполнение файл `.logout`, а в Z-Shell — `.zlogout`.

Часть II

Основные команды и программы ОС UNIX/Linux

Все команды, набираемые в командной строке или исполняемые в командном файле, являются либо командами, встроенными в интерпретатор, либо внешними исполняемыми файлами. Набор встроенных команд достаточно невелик, что определяется основной концепцией UNIX — система должна состоять из небольших программ, выполняющих достаточно простые четко определенные функции, связывающиеся между собой по стандартному интерфейсу. Существует два основных набора внешних утилит, являющихся на сегодняшний день стандартом для UNIX-подобных систем:

- **SYSV Utilities** — утилиты, разработанные в рамках проекта SYSTEM V и соответствующие стандарту POSIX P1003.2, установленному группой X/OPEN. Стандарт для всех коммерческих производителей систем UNIX.
- **GNU Utilities** — ставший стандартом de facto набор команд, разрабатываемый FSF (Free Software Foundation) и доступный бесплатно в исходных кодах. Это набор, в достаточной степени соответствующий POSIX, но с некоторыми расширениями. Достаточно легко собирается на любой UNIX-подобной системе.

Глава 3

Поиск справочной информации

Система UNIX с самого начала ее использования снабжалась богатым комплектом документации. Некоторое количество информации часто находится в каталогах `/usr/doc` или `/usr/local/doc` в виде текстовых файлов. Традиционной для UNIX'а с момента ее создания является команда **man**, созданная еще в эпоху телетайп-терминалов и отлично работающая до сих пор на всех видах оборудования, но имеющая определенные ограничения: невозможность использования графических иллюстраций и гипертекстовых ссылок. В рамках GNU-проекта была создана система **info**, также работающая на всех видах алфавитно-цифровых терминалов, но с поддержкой гипертекста. Для всех GNU-утилит прилагаются соответствующие справочные файлы как в формате **man**, так и в формате **info**. Практически каждым коммерческим производителем UNIX-систем была создана собственная система помощи, включавшая как поддержку гипертекстов, так и графику, и работающая под управлением системы X Window. С появлением HTML справочная информация стала предоставляться в этом формате прямо в составе системы или на WWW-серверах компаний производителей.

3.1 Системное руководство man

3.1.1 Команда man

Синтаксис:

	SYSV	man [-t] [-s <i>i</i>] <i>имя</i>
	GNU, BSD	man [-t] [-<i>i</i>] <i>имя</i>
	SYSV, GNU, BSD	man [-k]

Описание:

Команда **man** выводит страницу руководства для указанного имени на стандартный вывод или при помощи `pager'a`, установленного для данной сессии для постраничной выдачи информации.

Каждая страница руководства имеет стандартную форму со следующими разделами:

- NAME — название и назначение
- SYNOPSIS — синтаксис
- DESCRIPTIONS — описание
- FILE — используемые файлы
- SEE ALSO — смежные разделы

- DIAGNOSTIC — диагностика ошибок
- BUGS — замеченные ошибки

В UNIX-системах наблюдается две основные схемы разбиения страниц руководства на разделы:

Таблица 3.1 Схемы разбиения страниц руководства на разделы

Раздел	BSD	SYSV
Команды пользователя	1	1
Системные вызовы	2	2
Библиотечные функции	3	3
Спец. файлы и внешние устройства	4	7
Форматы файлов	5	4
Игры и демонстрации	6	6 или 1 или отсутствуют
Разное (наборы символов, типы файловых систем и т.д.)	7	7
Команды для системного администрирования	8	1m
Команды поддержки	8	8
Драйверы устройств	4	7 или 9

Опции:

-k	Работает аналогично команде apropos
-t	Осуществляет вывод информации в формате PostScript
-s <i>i</i> , <i>i</i>	Номер <i>i</i> указывает секцию руководства, в которой надо производить поиск

Примеры: Получение справки о системной команде **man**.

```
$ man man
man(1) man(1)
```

NAME

man - format and display the on-line manual pages

SYNOPSIS

```
man [-acdfFhkKtwW] [--path] [-m system] [-p string]
[-C config_file] [-M pathlist] [-P pager] [-B
browser] [-H htmlpager] [-S section_list] [section] name ...
```

DESCRIPTION

man formats and displays the on-line manual pages. If you specify section, man only looks in that section of the manual. name is normally the name of the manual page, which is typically the name of a command, function, or file. However, if name contains a slash (/) then man interprets it as a file specification, so that you can do man ./foo.5 or even man /cd/foo/bar.1.gz.

See below for a description of where man looks for the manual page files.

3.1.2 Справочная система **whatis**

Синтаксис:

whatis *имя*...

Описание:

Выводит краткое описание указанной страницы руководства.

Пример:

```
$ whatis mount
mount          (2) - mount and unmount filesystems
mount          (8) - mount a file system
```

Примечание:

Требует наличия специальной базы данных, предварительно созданной командой **catman** или **makewhatis**.

3.1.3 Команда *apropos*

Синтаксис:

apropos слово

Описание:

Ищет страницы описаний по ключевому слову.

Примеры:

```
$ apropos printf
format        (n) - Format a string in the style of sprintf
printf        (1) - format and print data
printf        (3) - formatted output conversion
printf        (3p) - print formatted output
```

Примечание:

Требует наличия специальной базы данных, предварительно созданной командой **catman** или **makewhatis**.

3.1.4 Справочная система *info*

Синтаксис:

info [-directory *каталог* . . .] [-file *файл*]

Описание:

Просмотр справочной информации в формате *texinfo*. Путь поиска файлов *info* помещается в переменную \$INFOPATH.

Опции:

-directory <i>каталог</i>	Список <i>каталогов</i> для поиска
-file <i>файл</i>	Читать из <i>файла</i>

Управляющие последовательности:

h	Просмотр руководства
?	Получение подсказки
n	Следующий раздел
p	Предыдущий раздел
u	Подняться на уровень
m	Выбор пункта меню
f	Перейти по ссылке
SPACE	Пролистывание вниз
DEL	Пролистывание вверх
q	Выход

Примеры:

```
$ info --directory /usr/share/info/
File: dir      Node: Top      This is the top of the INFO tree
```

This (the Directory node) gives a menu of major topics.
 Typing "q" exits, "?" lists all Info commands, "d" returns here,
 "h" gives a primer for first-timers,
 "mEmacs<Return>" visits the Emacs topic, etc.

In Emacs, you can click mouse button 2 on a menu item or cross reference to select it.

* Menu:

```
Texinfo documentation system
* Pinfo: (pinfo).          curses based lynx-style info browser.
* Texinfo: (texinfo).      The GNU documentation format.
* info standalone: (info-stnd)  Read Info documents without Emacs.
* infokey: (info-stnd)Invoking infokey.  Compile Info customizations.
* install-info: (texinfo)Invoking install-info. Update info/dir entries.
* makeinfo: (texinfo)Invoking makeinfo.  Translate Texinfo source.
* texi2dvi: (texinfo)Format with texi2dvi.  Print Texinfo documents.
* texi2pdf: (texinfo)PDF Output.          PDF output for Texinfo.
* texindex: (texinfo)Format with tex/texindex.  Sort Texinfo index files.
```

Miscellaneous

```
* As: (as).                The GNU assembler.
* Bfd: (bfd).              The Binary File Descriptor library.
* Binutils: (binutils).    The GNU binary utilities.
```

Глава 4

Информационные команды

4.1 Команда uname

Синтаксис:

uname [-amnrsv]

Описание:

Вывод информации о данной операционной системе. При отсутствии аргументов выводится лишь ее имя.

Опции:

-a	Выводится вся возможная информация
-m	Тип компьютера, на котором работает система
-n	Узел сети
-r	Номер редакции
-s	Название операционной системы
-v	Номер версии операционной системы

Примеры:

ОС Linux:

```
uname -a
```

```
Linux felix 2.6.18-164.15.1.el5PAE #1 SMP Tue Mar 16 19:14:29 EDT 2010  
i686 i686 i386 GNU/Linux
```

4.2 Команда date

Синтаксис:

date [~~ММДДЧЧММ~~][ГГ]

Описание:

Вывод и установки текущей даты и времени.

Формат даты по умолчанию:

<i>ММ</i>	Месяц
<i>ДД</i>	День
<i>чч</i>	Часы
<i>мм</i>	Минуты
<i>ГГ</i>	Год

4.3 Команда *cal*

Синтаксис:

***cal* [-y] [[*месяц*] *год*]**

Описание:

Просмотр календаря.

Опции:

-y	Показать календарь на текущий год
----	-----------------------------------

Примеры:

```

        Июль 2010
Вс Пн Вт Ср Чт Пт Сб
          1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

```

4.4 Команда *id*

Синтаксис:

***id* [*пользователь*]**

Описание:

Вывод идентификатора пользователя и группы. Если пользователь не указан, выдается информация о пользователе, вызвавшем команду.

Примеры:

Получить идентификаторы пользователя *root*:

```

$ id root
uid=0(root) gid=0(root) группы=0(root),1(bin),2(daemon),3(sys),4(adm),
6(disk),10(wheel),102(pkcs11) context=user_u:system_r:unconfined_t

```

4.5 Команда `logname`

Синтаксис:

`logname`

Описание:

Выдает регистрационное имя пользователя.

Примеры:

```
$ logname  
guest
```

Глава 5

Работа с другими пользователями

5.1 Команда `who`

Синтаксис:

who

Описание:

Вывести список пользователей, работающих в системе.

Примеры:

```
$ who
guest      :0                2010-07-01 18:45
guest      pts/1                2010-07-01 18:45 (:0.0)
guest      pts/2                2010-07-01 18:46 (:0.0)
guest      pts/3                2010-07-01 18:47 (:0.0)
guest      pts/4                2010-07-01 19:04 (:0.0)
```

5.2 Команда `write`

Синтаксис:

write *пользователь* [*терминал*]

Описание:

Посылка сообщения другому *пользователю*¹. В команде можно указать определенный терминал, на котором работает пользователь.

¹Разрешить или запретить доступ к своему терминалу можно командой **mesg**.

Глава 6

Работа с терминальными линиями

6.1 Команда `tty`

Синтаксис:

`tty [-s]`

Описание:

Получение имени спец. файла, соответствующего текущему терминалу.

Опции:

<code>-s</code>	Имя терминала не выводится. Если стандартный ввод происходит с терминала, возвращается 0
-----------------	--

Примеры:

```
$ tty  
/dev/ttyr3
```

6.2 Команда `stty`

Синтаксис:

`stty [-a] [установки] [< терминал]`

Описание:

Установка свойств терминальной линии.

Опции:

<code>-a</code>	Вывести все характеристики линии
<code>sane</code>	Установка значений терминальной линии по умолчанию

Примеры:

```
$ stty -a  
speed 38400 baud; rows 45; columns 113; line = 0;  
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = M-^?;
```

```
eol2 = M-^?; swtch = M-^?; start = ^Q;  
stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W; lnext = ^V; flush = ^O;  
min = 1; time = 0;  
-parenb -parodd cs8 hupcl -cstopb cread -clocal -crtscts -cdtrdsr  
-ignbrk brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon  
-ixoff -iuclc ixany imaxbel iutf8  
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0  
ff0 isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop  
-echoprt echoctl echoke
```

Глава 7

Работа с файлами и каталогами

7.1 Работа с файловой системой

7.1.1 Команда ls

Синтаксис:

ls [-acltuFR] [файлы_или_каталоги]

Описание:

При указании в аргументах *файла* выводится информация о нем согласно указанным опциям, в случае *каталога* — то же для файлов каталога. Если аргументов нет, — выдается листинг текущего каталога.

Опции:

-a	Выводить информацию обо всех файлах в каталоге (по системному соглашению файлы, имя которых начинается с точки, являются скрытыми, и командой ls не показываются)
-c	Использовать время последней модификации i-node, связанного с файлом (создание файла, изменение прав, и т.д.), а не время модификации
-i	Выдать номер i-node, связанного с файлом
-l	Выдавать информацию о файлах в длинном формате: тип файла (обычный файл -, каталог <i>d</i> , сокет <i>s</i> , символьная ссылка <i>L</i> , символьное устройство <i>c</i> , блочное устройство <i>b</i> , FIFO <i>p</i>), права доступа (чтение — <i>r</i> , запись — <i>w</i> , исполнение — <i>x</i> ; первые 3 символа относятся к владельцу, следующие 3 — к членам группы, владеющей файлом, и последние 3 — ко всем остальным), владелец, группа, размер в байтах, дата модификации, имя файла
-t	Отсортировать листинг по времени
-u	Использовать время последнего доступа к файлу, а не время модификации
-F	После каждого имени выдавать значок, характеризующий тип последнего: каталог (/), исполняемый файл (*), FIFO (), символьная ссылка (@), сокеты семейства AF_UNIX (=)
-R	Выдавать листинг рекурсивно по подкаталогам

Примеры:

Получение полного листинга текущего каталога, включая скрытые файлы.

```
$ ls -la
итого 224
drwx----- 16 guest guest 4096 Мар 21 18:16 .
drwxr-xr-x  3 root root 4096 Мар 21 17:53 ..
-rw-----  1 guest guest   21 Мар 21 17:54 .bash_history
-rw-r--r--  1 guest guest   33 Дек 16 22:42 .bash_logout
-rw-r--r--  1 guest guest  176 Дек 16 22:42 .bash_profile
-rw-r--r--  1 guest guest  124 Мар 16 22:42 .bashrc
drwxr-xr-x  2 guest test  4096 Мар 21 17:53 Desktop
```

7.1.2 Команда pwd

Синтаксис:

pwd

Описание:

Выводит имя текущего каталога.

Примеры:

Определяем текущий каталог.

```
$ pwd
/home/guest
```

7.1.3 Команда cd

Синтаксис:

cd [каталог]

Описание:

Сменить текущий *каталог* на указанный в аргументе. При отсутствии аргументов происходит переход в домашний каталог пользователя \$HOME.

Примеры:

Переход из текущего каталога в каталог /usr/share/doc/.

```
$ pwd
/home/guest
$cd /usr/share/doc/
$pwd
/usr/share/doc/
```

7.1.4 Команда cp

Синтаксис:

cp [-i] файл ... файл_или_каталог

cp -r [-i] каталог ... каталог

Описание:

Копирует *файлы* или *каталог*, указанный в первых параметрах, в файл или каталог, указанный в последнем.

Опции:

-i	Интерактивно — запрашивается подтверждение на перезапись существующих файлов
-r	Рекурсивное копирование каталога

Примеры:

Рекурсивное копирование каталога **a** в **b**:

```
$ cp -r a b
```

Интерактивное копирование файлов из каталога **a** в **b**:

```
$ cp -i a/* b
cp: overwrite b/1 (yes/no)? y
cp: overwrite b/2 (yes/no)? n
cp: overwrite b/2 (yes/no)? y
```

7.1.5 Команда ln

Синтаксис:

ln [-fs] *файл_или_каталог* ... [*ссылка_или_каталог*]

Описание:

Устанавливает ссылку в файловой системе (символьную или жесткую). Если второй операнд является уже существующим каталогом, то ссылки создаются внутри него. В случае, если второй аргумент отсутствует, ссылка создается в текущем каталоге с именем источника.

Опции:

-f	Устанавливать ссылку вместо существующего файла
-s	Символьная ссылка

Примеры:

Сделать символьную ссылку **b** на **a**:

```
$ ln -s a b
$ ls -l
итого 8
-rw-rw-r-- 1 guest guest 0 Map 21 18:57 a
lrwxrwxrwx 1 guest guest 1 Map 21 18:57 b -> a
```

Сделать жесткую ссылку между **a** и **c**:

```
$ ln a c
$ ls -l
итого 12
-rw-rw-r-- 2 guest guest 0 Map 21 18:57 a
lrwxrwxrwx 1 guest guest 1 Map 21 18:57 b -> a
-rw-rw-r-- 2 guest guest 0 Map 21 18:57 c
$ ls -il
итого 12
```

```
4247456 -rw-rw-r-- 2 guest guest 0 Map 21 18:57 a
4247457 lrwxrwxrwx 1 guest guest 1 Map 21 18:57 b -> a
4247456 -rw-rw-r-- 2 guest guest 0 Map 21 18:57 c
```

7.1.6 Команда mv

Синтаксис:

mv [-i] *файл_или_каталог* *файл_или_каталог*

Описание:

Переименовать *файл* или *каталог*, указанный в первом аргументе, в *файл* или *каталог*, указанный во втором.

Опции:

-i	Интерактивно — запрашивать разрешение на перезапись уже существующих объектов
----	---

Примеры:

Переименовать b в d:

```
$mv b d
$ ls -l
итого 12
-rw-rw-r-- 2 guest guest 0 Map 21 18:57 a
-rw-rw-r-- 2 guest guest 0 Map 21 18:57 c
lrwxrwxrwx 1 guest guest 1 Map 21 18:57 d -> a
```

Попытка переименования файла самого в себя:

```
$ mv d c
mv: `d' и `c' - один и тот же файл
```

7.1.7 Команда rm

Синтаксис:

rm [-f] [-i] *файл...*

rm -r [-f] [-i] *каталог...*

Описание:

Удалить файлы или каталоги. Если права доступа не позволяют этого сделать немедленно, то последние выводятся в восьмеричной форме и требуется подтверждение операции.

Опции:

-f	Не спрашивать подтверждений, удалить все, что возможно
-i	Запрашивать подтверждение на каждый удаляемый файл
-r	Выполнить удаление рекурсивно, включая каталоги

Примеры:

Рекурсивное удаление каталога `.mozilla/firefox/gb16cbr1.default/Cache` в принудительном режиме:

```
$ rm -rf .mozilla/firefox/gb16cbr1.default/Cache
```

7.1.8 Команда `mkdir`

Синтаксис:

`mkdir [-p] каталог`

Описание:

Создать *каталог*¹.

Опции:

<code>-p</code>	Создаются также и все несуществующие к этому моменту родительские каталоги указанного места
-----------------	---

Примеры:

Создать каталог `/tmp/a/b/c`:

```
$ mkdir -p /tmp/a/b/c
```

```
$ ls -R /tmp
```

```
/tmp:
```

```
a          gconfd-guest
```

```
/tmp/a:
```

```
b
```

```
/tmp/a/b:
```

```
c
```

```
/tmp/a/b/c:
```

7.1.9 Команда `rmdir`

Синтаксис:

`rmdir [-p] каталог`

Описание:

Удалить пустой *каталог*.

Опции:

<code>-p</code>	Удалить также и все пустые родительские каталоги
-----------------	--

¹Для создания каталога необходимо иметь права на запись в родительский каталог.

Примеры:

Удалить каталог /tmp/a/b/c:

```
$ ls -R /tmp
/tmp:
a          gconfd-guest

/tmp/a:
b

/tmp/a/b:
c

/tmp/a/b/c:

$ rmdir -p /tmp/a/b/c
$ ls -R /tmp
/tmp:
gconfd-guest
```

7.2 Изменение атрибутов файлов

7.2.1 Команда chmod

Синтаксис:

chmod [-R] ### файл_или_каталог

chmod [-R] [ugoa][+ -=][rwxst] файл_или_каталог

Описание:

Сменить права доступа *файлам* или *каталогам*, владельцем которых являетесь вы. Мода доступа представляет собой права владельца, группы и всех остальных на чтение, запись и исполнение.

Опции:

-R	Выполнить операцию рекурсивно
###	Трехзначное восьмеричное число, определяющее права доступа
u	Для владельца
g	Для группы
o	Для остальных
a	Для всех
+	Добавить права
-	Отменить права
=	Установить права
r	Чтение
w	Запись или создание/удаление файлов в каталоге
x	Исполнение или возможность перейти в каталог
s	Установка бита <i>set-uid</i> , <i>set-gid</i>
t	Файлы из каталога может удалять только их владелец

Примеры:

Разрешить для всего каталога `b` запись группе и запретить чтение и переход в этот каталог для всех остальных.

```
$ ls -la b
итого 28
drwxrwxr-x 2 guest guest 4096 Map 21 19:40 .
drwx----- 5 guest guest 4096 Map 21 19:39 ..
-rw-rw-r-- 1 guest guest 0 Map 21 19:40 1
-rw-rw-r-- 1 guest guest 0 Map 21 19:40 2
-rw-rw-r-- 1 guest guest 0 Map 21 19:40 3
$ chmod -R g+w,o-rx b
$ ls -la b
итого 28
drwxrwx--- 2 guest guest 4096 Map 21 19:40 .
drwx----- 5 guest guest 4096 Map 21 19:39 ..
-rw-rw---- 1 guest guest 0 Map 21 19:40 1
-rw-rw---- 1 guest guest 0 Map 21 19:40 2
-rw-rw---- 1 guest guest 0 Map 21 19:40 3
```

7.2.2 Команда `chown`

Синтаксис:

`chown [-R] владелец[:группа] файл_или_каталог...`

Описание:

Сменить владельца (и группу, если указана) файла или каталога.

Опции:

<code>-R</code>	Выполнить операцию рекурсивно
-----------------	-------------------------------

Примеры:

Сменить у файла `test` владельца на `oracle`, а группу на `dba`:

```
$ ls -l test
-rw-rw-r-- 1 guest guest 124 Map 21 19:50 test
$ chown oracle:dba test
$ ls -l test
-rw-rw-r-- 1 oracle dba 124 Map 21 19:50 test
```

7.3 Просмотр содержимого файлов

7.3.1 Команда `cat`

Синтаксис:

`cat [-n] [файл...]`

Описание:

Вывести на стандартный вывод содержимое *файлов* в порядке их следования, или на стандартный ввод, если файлы не указаны. Ввод завершается символом *End of File* (**[Ctrl]-[D]**), который в выходной поток не помещается.

-n	Выводить порядковый номер строки
----	----------------------------------

Примеры:

Заполнить файл `test`:

```
$ cat > test
one
two
three
four
five
six
seven
eight
nine
ten
```

Вывести содержимое файла `test`:

```
$ cat test
one
two
three
four
five
six
seven
eight
nine
ten
```

7.3.2 Команда `more`

Синтаксис:

`more [+номер_строки] [+образец_поиска] [файл...]`

Описание:

Постраничный просмотр *файлов* или стандартного ввода.

Некоторые управляющие последовательности:

SPACE	Переместиться на одну страницу вниз
ENTER	Переместиться на одну строку вниз
d	Переместиться на полстраницы вниз
b	Переместиться на одну страницу вверх
/образец/	Поиск по заданному образцу вперед
?образец?	Поиск по заданному образцу назад
h	Помощь
q	Выход

Опции:

+номер_строки	Спозиционироваться на указанную строку
+/образец_поиска	Спозиционироваться на строку, соответствующую образцу

7.3.3 Команда less

Синтаксис:

GNU: `less [+номер_строки] [+/образец_поиска] [файл. . .]`

Описание:

Постраничный просмотр *файлов* или стандартного ввода.

Управляющие последовательности:

стрелка_вверх	Переместиться на одну строку вверх
стрелка_вниз	Переместиться на одну строку вниз
PageUp	Переместиться на одну страницу вверх
PageDown	Переместиться на одну страницу вниз
SPACE	Переместиться на одну страницу вниз
ENTER	Переместиться на одну строку вниз
d	Переместиться на полстраницы вниз
b	Переместиться на одну страницу вверх
/образец/	Поиск по заданному образцу вперед
?образец?	Поиск по заданному образцу назад
h	Помощь
q	Выход

Опции:

+номер_строки	Спозиционироваться на указанную строку
+/образец_поиска	Спозиционироваться на строку, соответствующую образцу

7.3.4 Команда rg

rg `[+номер_строки] [+/образец_поиска] [файл. . .]`

Описание:

Постраничный просмотр *файлов* или стандартного ввода.

Управляющие последовательности:

l	Переместиться на одну строку вниз
ENTER	Переместиться на одну страницу вниз
f	Переместиться на одну страницу вверх
/образец/	Поиск по заданному образцу вперед
?образец?	Поиск по заданному образцу назад
h	Помощь
q	Выход

Опции:

+номер_строки	Спозиционироваться на указанную строку
+/образец_поиска	Спозиционироваться на строку, соответствующую образцу

7.3.5 Команда head

Синтаксис:

head [-строк] [файл...]

Описание:

Вывести начало файла или входного потока.

-строк	Количество строк
---------------	------------------

Примеры:

Вывести первые 5 строк файла test:

```
$ head -5 test
one
two
three
four
five
```

7.3.6 Команда tail

Синтаксис:

tail [-строк] [файл...]

Описание:

Вывести конец файла или входного потока.

Опции:

-строк	Количество строк
---------------	------------------

Примеры:

Вывести последние 5 строк файла `test`:

```
$ head -5 test
six
seven
eight
nine
ten
```

7.3.7 Команда `tee`

Синтаксис:

`tee [файл...]`

Описание:

Разделить входной поток. В результате выполнения команды он копируется в стандартный выходной поток и в указанные файлы.

7.3.8 Команда `split`

Синтаксис:

`split [-l строк] [-b байтов[km]] [файл] [выходной_префикс]`

Описание:

Разбить *файл* или поток на файлы указанного размера в строках и байтах. В результате операции создается набор файлов. Файлы получают имена, начинающиеся с указанного *выходного_префикса* (по умолчанию — `'x'`) и заканчивающиеся набором букв в соответствующем лексикографическом порядке.

Опции:

<code>-l <i>строк</i></code>	В каждом выходном файле должно оказаться не более указанного количества <i>строк</i>
<code>-b <i>байтов</i>[<i>km</i>]</code>	В каждом выходном файле должно оказаться не более указанного количества <i>байтов</i> . Дополнительные спецификаторы обозначают: <i>k</i> — килобайты, <i>m</i> — мегабайты

7.3.9 Команда `wc`

Синтаксис:

`wc [-clw] [файл...]`

Описание:

Подсчет строк, слов и байтов.

Опции:

<code>-c</code>	Подсчет байтов
<code>-l</code>	Подсчет строк
<code>-w</code>	Подсчет слов

Примеры:

Подсчитать количество строк, слов и байтов в файле `test`:

```
$ wc test
    10      10      49 test
```

Подсчитать количество байтов в файле `test`:

```
$ wc -c test
    49 test
```

Подсчитать количество строк в файле `test`:

```
wc -l test
    10 test
```

Подсчитать количество слов в файле `test`:

```
wc -w test
    10 test
```

7.3.10 Команда `od`

Синтаксис:

`od [-bcdox] [файл]`

Описание:

Вывод содержимого файла в указанном формате.

Опции:

<code>-b</code>	Побайтовый вывод в восьмеричном виде
<code>-c</code>	Побайтовый вывод в виде символа
<code>-d</code>	Пословный вывод как беззнаковое десятичное
<code>-o</code>	Пословный вывод как беззнаковое восьмеричное
<code>-x</code>	Пословный вывод как беззнаковое шестнадцатеричное

Примеры:

Просмотреть содержимое жесткого диска `/dev/hda` побайтно в восьмеричной форме и символьной:

```
$ od -bc /dev/hda | less
```

7.4 Поиск информации в файлах

7.4.1 Команды `grep`, `egrep`

Синтаксис:

`grep [-cinv] образец [файл...]`

`egrep [-cinv] образец [файл...]`

Описание:

Поиск строк по заданным образцам. В команде **egrep** в качестве образца могут использоваться регулярные выражения.

Опции:

<code>-c</code>	Вывести только общее число строк
<code>-i</code>	Не учитывать регистр букв
<code>-n</code>	Перед каждой строкой выводить ее номер
<code>-v</code>	Учитывать только строки, не соответствующие образцу

Примеры:

Найти строки, содержащие сочетание букв `ne`:

```
$ grep -n ne test
1:one
9:nine
```

Найти строки, где букве `o` предшествует хотя бы один символ:

```
$ egrep '.+o' test
two
four
```

Найти строки, где стоящей в конце строки букве `o` предшествует хотя бы один символ:

```
$ grep '.*o$' test
two
```

7.5 Команды архивации, компрессии и декомпрессии

7.5.1 Команда `tar`

Синтаксис:

SYSV: `tar [-c|r|t|u|x[wv]][f файл] файл_или_каталог...`

GNU: `tar [-c|r|t|u|x[wvz]][f файл] файл_или_каталог...`

Описание: Архивация на внешних устройствах или в файлах.

Обязательные опции:

c	Создать
r	Дописать файлы в конец архива
t	Вывести листинг архива
u	Добавить файл, если его в архиве нет
x	Извлечь указанные файлы из архива (по умолчанию — все)

Необязательные опции:

f <i>файл</i>	Работать с файлом <i>файл</i> , вместо устройства, установленного по умолчанию. Символ '-' в качестве имени файла означает стандартный вывод или ввод, в зависимости от контекста. В GNU-версии может использоваться в сети. Полное имя архива в этом случае — <i>пользователь@хост:/файл</i> , но необходимо иметь права доступа на архивный хост по <i>r-службам</i>
w	Запрашивать разрешение
v	Отображать больше информации (verbose)
z	Использовать компрессию gzip
j	Использовать компрессию bzip2

Примеры:

Создать архив `test.tar` из каталогов `a` и `b`:

```
$ tar cvf test.tar a b
a/
a/c
a/a
b/
b/2
b/1
b/3
```

Оценить содержимое `test.tar`:

```
$ tar tvf test.tar
drwxrwxr-x test/test      0 2010-07-09 14:43:24 a/
lrwxrwxrwx test/test      0 2010-07-09 14:43:24 a/c -> a
-rw-rw-r-- test/test      0 2010-07-09 14:43:17 a/a
drwxrwxr-x test/test      0 2010-07-09 14:43:42 b/
-rw-rw-r-- test/test      0 2010-07-09 14:43:40 b/2
-rw-rw-r-- test/test      0 2010-07-09 14:43:36 b/1
-rw-rw-r-- test/test      0 2010-07-09 14:43:42 b/3
```

Извлечь файл `a/a` из архива `test.tar`:

```
$ tar xvf test.tar a/a
a/a
```

7.5.2 Команда `cpio`

Синтаксис:

`cpio -i[cdkmrtuv] [-H формат]`

`cpio -o[acvAL] [-O файл] [-H формат]`

`cpio -p[lmuvL] каталог`

Описание:

Архивация на внешних устройствах или в файлах

Опции:

Обязательные:

-i	Copу-In mode. Читает список файлов из стандартного ввода и выводит созданный архив на стандартный вывод
-o	Copу-Out mode. Читает архив из стандартного ввода и распаковывает содержащиеся в нем файлы
-p	Copу-Pass mode. Совмещает в себе Copу-In и Copу-Out, используется для копирования набора файлов в <i>каталог</i>

Опции:

-a	Переустановить <i>access time</i> так, чтобы они не выглядели только что прочитанными
-c	Использовать старый переносимый SVR4 ASCII формат
-d	Создавать каталоги в случае необходимости
-k	Пропускать сбойные заголовки при ошибках ввода/вывода
-l	Предпочтительное создание ссылки, а не копирование
-m	Восстановление <i>modification time</i> при извлечении файлов
-r	Переименовывать файлы интерактивно
-t	Вывести содержимое архива. Файлы не извлекаются
-u	Безусловное копирование (по умолчанию новые файлы не замещаются старыми)
-v	Выдавать больше информации
-A	Добавить файлы в архив, требует опции -O. Работает только с файлами, но не с устройствами
-L	Следовать символьным ссылкам
-O <i>файл</i>	Имя <i>файла</i> архива. В GNU-версии может использоваться в сети. Полное имя архива в этом случае — <i>пользователь@хост:/файл</i> , но необходимо иметь права доступа на архивный хост по <i>r-службам</i>
-H <i>формат</i>	Использовать архив указанного <i>формата</i> . О типах форматов, поддерживаемых данной версией cpio , надо справиться в документации.

7.5.3 Команды `gzip` и `gunzip`

Синтаксис:

GNU: `gzip [-cdv] [файл . . .]`

GNU: `gunzip [-cv] [файл . . .]`

Описание:

Сжатие и разжатие *файлов*. Стандарт для *GNU-проекта*. Сжимает по алгоритму LZ77. Сейчас разработаны и другие программы, предназначенные для той же цели, например, **bzip2**, **bunzip2**, осуществляющие более эффективное сжатие по алгоритму Burrows-Wheeler'a и имеющие аналогичные опции.

Опции:

-c	Операции производить с потоками стандартного ввода и вывода
-d	Разжатие
-v	Выдавать больше информации

Примечание:

В комплекте с **GNUzip** поставляются также утилиты **zcat**, **zmore**, **zless**, работающие как и соответствующие им обычные UNIX-команды, а также **gzexe**, осуществляющая компрессию исполняемых файлов.

7.6 Подсчет контрольных сумм

7.6.1 Команда sum

Синтаксис:

sum *файл*

Описание:

Вывод контрольной суммы и размера файла.

Примеры:

Получить контрольную сумму файла *adv-user-guide.pdf*:

```
$ sum adv-user-guide.pdf
24892  291
```

7.6.2 Команда md5sum

Синтаксис:

GNU: **md5sum** *файл*

Описание:

Вывод контрольной суммы MD5.

Примеры:

Получить контрольную сумму файла *adv-user-guide.pdf*:

```
$ md5sum adv-user-guide.pdf
2ce4d2c7d1d6720d2072cb239f4acf9d  adv-user-guide.pdf
```

7.7 Команды доступа к принтеру

7.7.1 Команда lpr

Синтаксис:

BSD: lpr [-P *принтер*] [-m] [-# *экземпляров*] [*файл*]

Описание:

Поставить файл в очередь на печать

[-P <i>принтер</i>]	Послать на <i>принтер</i> вместо используемого по умолчанию
[-m]	Послать уведомление по <i>e-mail</i> после окончания печати
[-# <i>экземпляров</i>]	Количество <i>экземпляров</i>

7.7.2 Команда lprq

Синтаксис:

BSD: lprq [-l] [-P *принтер*] [*job номер...*] [*пользователь...*]

Описание:

Просмотреть очередь на печать.

Опции:

-l	Выдать информацию в расширенном формате
-P <i>принтер</i>	Для указанного <i>принтера</i>
<i>job принтер</i>	По заданию <i>номер</i>
<i>пользователь</i>	Для <i>пользователя</i>

7.7.3 Команда lprm

Синтаксис:

BSD: lprm [-P *принтер*] [-] [*job номер...*] [*пользователь...*]

Описание:

Удалить запрос из очереди.

Опции:

>-P <i>принтер</i>	Для указанного <i>принтера</i>
-	Удалить все задания данного пользователя
<i>job принтер</i>	Удалить задание <i>номер</i>
<i>пользователь</i>	Для <i>пользователя</i>

7.7.4 Команда lpr

Синтаксис:

SYSV: lpr [-d *принтер*] [-m] [-n *экземпляров*] [*файл...*]

Описание:

Поставить файл в очередь на печать.

Опции:

-d	Послать на <i>принтер</i> вместо используемого по умолчанию
-m	Послать уведомление по <i>e-mail</i> после окончания печати
-n <i>экземпляров</i>	Количество <i>экземпляров</i>

7.7.5 Команда `cancel`

Синтаксис:

SYSV: `cancel [задание] [принтер]`

SYSV: `cancel -u пользователь ... [принтер]`

Описание:

Отмена ранее посланного командой `lp` задания.

Первая форма удаляет задание по его уникальному идентификатору, а вторая — по идентификатору пользователя, пославшего задание на печать.

Опции:

<i>принтер</i>	Удалить задание из очереди <i>принтера</i>
----------------	--

7.7.6 Команда `lpstat`

Синтаксис:

SYSV: `lpstat [-s] [-t] [-p [принтер] [-l]] [-u пользователь...]`

Описание:

Вывод информации о состоянии системы печати.

Опции:

-d	Показать принтер, установленный по умолчанию
-s	Статус
-t	Показать всю информацию, выдаваемую по запросу <code>-s</code> , плюс занятость принтеров
-p <i>принтер</i>	Для <i>принтера</i>
-l	Длинный список
-u <i>пользователь...</i>	Запросы <i>пользователя</i>

7.8 Поиск и проверка состояния файлов

7.8.1 Команда find

Синтаксис:

find каталог... выражение

Описание:

Поиск файлов² в указанном списке *каталогов* согласно *выражению*.

Опции:

Выражение может быть составлено из следующих операторов:

-name <i>файл</i>	Поиск по имени, заданному образцом (возможно использование метасимволов)
-perm <i>права_доступа</i>	Поиск по правам доступа в виде 3-хзначного восьмеричного числа (9 младших бит моды). Если первым символом прав является '-', используются младшие 11 бит
-type [b c d p f]	Истинно, если тип файла — специальный блок-ориентированный, специальный байт-ориентированный каталог, FIFO-канал или обычный файл, соответственно
-links <i>число_связей</i>	Истинно, если файл имеет указанное <i>число_связей</i>
-user <i>пользователь</i>	Владельцем файла является <i>пользователь</i>
-group <i>группа</i>	Файл принадлежит <i>группе</i>
-size <i>размер</i> [c]	Размер файла равен числу блоков (по 512 байт), указанному в аргументе <i>размер</i> . Если после числа стоит c, — размер указан в байтах
-atime <i>дней</i>	К файлу в последние <i>дней</i> осуществлялся доступ. Сама команда find тоже переустанавливает <i>atime</i>
-mtime <i>дней</i>	Файл модифицировался в течение <i>дней</i>
-ctime <i>дней</i>	У файла в течение <i>дней</i> производилась смена атрибутов
-exec <i>команда</i>	Истинно, если при выполнении <i>команды</i> был возвращен код 0. Аргумент {} заменяется именем текущего файла
-ok <i>команда</i>	Аналогично -exec, но перед исполнением <i>команды</i> требуется подтверждение
-print	Всегда истинно. Печатает имя файла на стандартный вывод
-newer <i>файл</i>	Истинно, если текущий файл модифицировался позже, чем <i>файл</i>
(<i>выражение</i>)	Истинно, если все атомарные выражения в <i>выражении</i> истинны

Выражения, записанные подряд, комбинируются по **И**. Атомарные выражения могут комбинироваться с помощью следующих логических операторов:

!	Унарное отрицание
-o	Выражения комбинируются по ИЛИ

²В наборе *GNU Utilites* существуют средства, облегчающие эту задачу — **locate** и **updatedb**

7.8.2 Команды `which` и `type`

Синтаксис:

which *команда*

type *команда*

Описание:

Выводят полный путь до *команды*, если она находится в пути поиска `$PATH`, причем **which** — исполняемый файл, а **type** — встроенная команда некоторых интерпретаторов.

Примеры:

Определить местоположение команды **which**:

```
$ which which
/usr/bin/which
$ which type
which: no type in (/usr/lib/qt-3.3/bin:/usr/kerberos/bin:
/usr/local/bin:/bin:/usr/bin:/home/test/bin)
$ type which
which is hashed (/usr/bin/which)
$ type type
type is a shell builtin
```

7.8.3 Команда `file`

Синтаксис:

file *файл...*

Описание:

Выводит типы указанных *файлов*. Типы файлов определяются согласно описанию в файле `magic`, обычно находящемся в `/etc` или `/usr/lib`.

Примеры:

```
$ file *
5:                ASCII text
a:                directory
b:                directory
birds.avi:        RIFF (little-endian) data, AVI, 576 x 320, 23.98 fps,
video: XviD, audio: MPEG-1 Layer 3 (stereo, 48000 Hz)
Booklet.doc:      Microsoft Office Document
Booklet.pdf:      PDF document, version 1.4
Desktop:          directory
index.html:       HTML document text
linux-ink_logo.gif: GIF image data, version 89a, 200 x 47
Mail:             directory
Nau.png:          PNG image data, 270 x 55, 8-bit/color RGBA
rn.sh:            Bourne shell script text executable
test:             ASCII text
test.tar:         POSIX tar archive
text.txt:         ASCII text
true:             ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV),
```

for GNU/Linux 2.6.9, dynamically linked (uses shared libs),
for GNU/Linux 2.6.9, stripped

7.8.4 Команда test

Синтаксис:

test выражение

[*выражение*]

Описание:

Вычисление условного выражения.

Операнды выражения:

-r <i>файл</i>	<i>Файл</i> существует и доступен для чтения
-w <i>файл</i>	<i>Файл</i> существует и доступен для записи
-x <i>файл</i>	<i>Файл</i> существует и доступен для исполнения
-f <i>файл</i>	<i>Файл</i> существует и является обычным файлом
-d <i>файл</i>	<i>Файл</i> существует и является каталогом
-c <i>файл</i>	<i>Файл</i> существует и является байт-ориентированным устройством
-b <i>файл</i>	<i>Файл</i> существует и является блочно-ориентированным устройством
-p <i>файл</i>	<i>Файл</i> существует и является FIFO-файлом
-u <i>файл</i>	<i>Файл</i> существует и у него установлен атрибут <i>set-user-id</i>
-g <i>файл</i>	<i>Файл</i> существует и у него установлен атрибут <i>set-group-id</i>
-s <i>файл</i>	<i>Файл</i> существует и его размер > 0
-t [<i>файловый дескриптор</i>]	<i>Файловый дескриптор</i> (по умолчанию 1) — терминал
<i>файл1</i> -nt <i>файл2</i>	Время модификации <i>файла1</i> больше времени модификации <i>файла1</i>
<i>файл1</i> -ot <i>файл2</i>	Время модификации <i>файла1</i> меньше времени модификации <i>файла1</i>
<i>файл1</i> -ef <i>файл2</i>	<i>Файл1</i> и <i>файл2</i> — один и тот же файл
-n <i>строка</i>	Длина <i>строки</i> не 0
<i>строка1</i> = <i>строка2</i>	Равенство строк
<i>строка1</i> != <i>строка2</i>	Неравенство строк
<i>строка</i> ~ <i>образец</i>	<i>Строка</i> соответствует <i>образцу</i>
<i>строка</i> !~ <i>образец</i>	<i>Строка</i> не соответствует <i>образцу</i>
<i>строка1</i> < <i>строка2</i>	<i>Строка1</i> лексикографически меньше <i>строки2</i>
<i>строка1</i> > <i>строка2</i>	<i>Строка1</i> лексикографически больше <i>строки2</i>
<i>выражение1</i> - <i>авыражение2</i>	Объединить <i>выражения</i> по И
<i>выражение1</i> -o <i>выражение2</i>	Объединить <i>выражения</i> по ИЛИ
! <i>выражение</i> - <i>авыражение2</i>	Унарное отрицание <i>выражения</i>

Глава 8

Управление задачами

8.1 Команда ps

Синтаксис:

<i>SYSV:</i>	ps [-efl]
<i>BSD:</i>	ps [-][alx]

Описание:

Вывод информации о состоянии процессов.

Опции:

В *SYSV* системах:

-e	Выдать информацию и о процессах других пользователей
-f	Выдать полную информацию о процессах
-l	Выдать информацию о процессах в расширенном виде

В *BSD* системах:

a	Выдать информацию и о процессах других пользователей
l	Выдать информацию о процессах в длинном формате
x	Выдать в том числе и информацию о процессах, не подсоединенных к терминалам

Поля в выходном листинге:

USER (BSD), UID (System V)	Пользователь — хозяин процесса
PID	Идентификатор процесса
%CPU	Доля использования ресурсов процессора
%MEM	Доля использования оперативной памяти
SZ	Используемая виртуальная память в килобайтах (BSD) или страницах (System V)
RSS	Используемая реальная память (в тех же единицах, что и SZ)
TT, TTY	Терминал, связанный с процессом
STAT (BSD), S (System V)	Статус текущего процесса: <ul style="list-style-type: none"> • R — работает • S — спит • I — простаивает (BSD), промежуточное состояние (System V) • T — остановлен • Z — зомби • D (BSD) — ожидание диска • P (BSD) — ожидание страницы • X (System V) — ожидание добавочной памяти • W (BSD) — выгружено в свопинг • N (BSD) — Niced: приоритет понижен • > (BSD) — Niced: приоритет искусственно повышен
TIME	Полное время работы на CPU
COMMAND	Командная строка программы
STIME (System V)	Время старта процесса
F	Флаги, связанные с процессом
PPID	PID процесса-родителя
NI	Nice процесса
C (System V), CP (BSD)	Используемость процессора, используется при вычислении приоритета (PRI)
PRI	Текущий приоритет
WCHAN	Ожидание события

8.2 Команда jobs

Синтаксис:

jobs [-l]

Описание:

Вывести список задач в данной сессии **shell**.

Опции:

-l	Вывести также PID процесса, соответствующего задаче
----	---

8.3 Команда **bg**

Синтаксис:

bg [%*задача*]

Описание:

Перевод *задачи shell* в фоновый режим. Если *задача* не указана, имеется в виду текущая.

8.4 Команда **fg**

Синтаксис:

fg [%*задача*]

Описание:

Перевод *задачи shell* в активный режим. Если *задача* не указана, имеется в виду текущая.

8.5 Команда **kill**

Синтаксис:

kill [-*сигнал*] *PID* . . .

kill [-*сигнал*] %*задача* . . .

Описание:

Посылка *сигнала* процессу с идентификационным *PID* (получаемый командой **ps** или номером *задачи shell*).

Глава 9

Отложенной исполнение команд

9.1 Команда sleep

Синтаксис:

sleep *время*

Описание:

Приостановить интерпретацию команд текущего **shell**'а на время, указанное в секундах.

9.2 Команды at и batch

Синтаксис:

at [-f *файл...*] [-m] *время...*

batch [-m] *время...*

Описание:

Запуск заданий в определенное время. Если время **batch** не указано, задание выполняется немедленно. По умолчанию задания читаются со стандартного ввода.

Опции:

-f <i>файл</i>	Файл с выполняемым shell -скриптом
-m	Послать пользователю, выполняющему команды, <i>e-mail</i> с содержимым потоков стандартного вывода и вывода ошибок запущавшихся команд по завершению их работы
<i>время...</i>	Время запуска в формате <i>чч:мм</i> или <i>midnight</i> (00:00), <i>noon</i> (12:00), <i>now</i> (выполнить немедленно). Если указанное время при наборе запроса уже прошло, задание назначается на следующие сутки. В запросе может быть указана и дата в форматах <i>месяц день [год]</i> , <i>ММДДГГ</i> , <i>ММ/ДД/ГГ</i> , <i>ММ:ДД:ГГ</i> или <i>today</i> (сегодня), <i>tomorrow</i> (завтра). Возможно указание смещения в формате <i>метка_времени + количество_единицы_времени</i> , где <i>единица_времени</i> — <i>minutes</i> (минуты), <i>hours</i> (часы), <i>days</i> (дни), <i>weeks</i> (недели)

Примеры:

Произвести сортировку файла `file` завтра в 07:30.

```
$ at -m 07:30 tomorrow
at< sort > file < outfile
at< >EOT<
job 5 at 2010-07-11 07:30
$ batch -f daily-processing now +1 hour
```

9.3 Команда atq

Синтаксис:

atq

Описание:

Просмотреть очередь заданий.

Примеры:

```
$ atq
2      2010-07-11 07:30 a guest
```

9.4 Команда atrm

Синтаксис:

atrm задание [задание...]

Описание:

Удалить задание из очереди по номеру, полученному из **atq**.

Примеры:

```
$ atq
2      2010-07-11 07:30 a guest
$ atrm 2
$ atq
```

9.5 Команда crontab

Синтаксис:

crontab [-l|r|e]

Описание:

Периодический запуск команд. Запросы заполняются следующим образом:

минуты(0-59) часы(0-23) день(1-31) месяц(1-12) день_недели(1-7) команда

Опции:

-e	Отредактировать файл запросов
-l	Вывести список запросов
-r	Удалить файл запросов

Примеры:

Удалять файлы core каждый рабочий день в 03:15.

```
$ crontab -l
```

```
15 3 * * 1-5 find $HOME -name core 2>/dev/null | xargs rm -f
```

Глава 10

Текстовые редакторы

10.1 Команда ed

Синтаксис:

ed [-] [файл...]

Описание:

Вызвать строковый редактор для указанных *файлов*.

Опции:

-	Подавление вывода посторонней информации (число строк, диагностика)
---	---

Командные последовательности:

.	Текущая строка
\$	Последняя строка
<i>строка</i>	Номер <i>строки</i>
<i>/рег_выражение/</i>	Первая вперед по буферу строка, соответствующая <i>регулярному выражению</i>
<i>?рег_выражение?</i>	Первая назад по буферу строка, соответствующая <i>регулярному выражению</i>
<i>+строк -строк</i>	На указанное количество строк вперед или назад
<i>+ + -</i>	На 1 строку вперед или назад

Команды:

a	Добавить строку
c	Заменить строку
d	Удалить строку
e <i>файл</i>	Открыть <i>файл</i> для редактирования
i	Вставить перед указанной строкой
<i>s/рег_выражение/замена</i>[g]	Заменить участки, соответствующие <i>регулярному выражению</i> на <i>замену</i> . Участки <i>рег_выражения</i> , ограниченные (), вставляются при замене на место, указанное конструкцией <i>\номер</i>
q	Выйти
w [<i>файл</i>]	Сохранить в <i>файле</i>
!<i>команда</i>	Выполнить команду shell

10.2 Команда vi

Синтаксис:

vi [+***строка***] ***файл***...

vi -r *файл*

Описание:

Вызвать редактор для указанных *файлов*.

Вторая форма позволяет восстановить прерванную сессию.

Опции:

+<i>строка</i>	Спозиционироваться в <i>строку</i> файла
+	Спозиционироваться в конец файла

Командные последовательности:

l, h	Сместиться на символ влево, вправо
w, W	Сместиться на слово влево, вправо
k, j	Сместиться на символ вверх, вниз
CTRL-B, CTRL-F	Сместиться на экран вверх, вниз
a	Вставлять текст после курсора
A	Вставлять текст в конец строки
i	Вставлять текст перед курсором
I	Вставлять текст в начало строки
x, X	Удалить символ до или после курсора
dw, dW	Удалить слово или набор символов до пробела
dd	Удалить строку
u	Отменить предыдущее действие
J	Слить строки
/рег_выражение	Поиск вперед
?рег_выражение	Поиск назад
число	Повторить указанное <i>число</i> раз
P, p	Вставить содержимое буфера до или после курсора
CTRL/L	Перерисовать экран
:rфайл	Прочитать <i>файл</i>
:wфайл	Записать в <i>файл</i>
:wq	Выход с сохранением
:q!	Выход без сохранения

10.3 Команда emacs

Синтаксис:

emacs **[+[строка]]** *файл* . . .

Описание:

Вызвать редактор для указанных *файлов*.

Опции:

+строка	Спозиционироваться в <i>строку</i> файла
+	Спозиционироваться в конец файла

Командные последовательности:

C-b, C-f	Сместиться на символ влево, вправо
M-b, M-f	Сместиться на слово влево, вправо
C-p, C-n	Сместиться на символ вверх, вниз
C-v, M-v	Сместиться на экран вверх, вниз
DEL, C-d	Удалить символ перед, после курсора
M-d	Удалить слово
C-k	Удалить текст до конца строки
C-w	Удалить маркированный участок
M-w	Пометить маркированный участок
C-x C-u	Отменить предыдущее действие
C-s	Поиск вперед
C-r	Поиск назад
C-, C-SPC	Поставить метку
C-y	Вставить содержимое буфера
M-x recover-session	Восстановить после краха
C-l	Перерисовать экран
C-x C-f	Открыть файл
C-x C-s	Сохранить файл
C-x C-c	Выйти

10.4 Команда joe

Синтаксис:

joe [-asis] [-crlf] *файл* . . .

Описание:

Вызвать редактор для указанных *файлов*.

Опции:

-asis	Отображать символы > 128, не отсекая 8-ой бит
-crlf	Использовать в качестве конечных символов в строке CR/LF

Командные последовательности:

Ctrl-K H	Help
Ctrl-B, Ctrl-F	Сместиться на символ влево, вправо
Ctrl-P, Ctrl-N	Сместиться на символ вверх, вниз
Ctrl-Z, Ctrl-X	Сместиться на слово влево, вправо
Ctrl-U, Ctrl-V	Сместиться на экран вверх, вниз
Ctrl-A	Сместиться в начало строки
Ctrl-E	Сместиться в конец
Ctrl-K U	Сместиться в начало файла
Ctrl-K V	Сместиться в конец файла
Ctrl-K L	Сместиться на определенную строку
Ctrl-K F	Поиск
Ctrl-L	Повторить поиск
Ctrl-K B	Начало блока
Ctrl-K K	Конец блока
Ctrl-K M	Переместить блок
Ctrl-K C	Скопировать блок
Ctrl-K W	Скопировать блок в файл
Ctrl-K Y	Удалить блок
Ctrl-K /	Направить блок на фильтр
Ctrl-D	Удалить букву
Ctrl-Y	Удалить строку
Ctrl-W	Удалить слово справа
Ctrl-O	Удалить слово слева
Ctrl-J	Удалить остаток строки
Ctrl-_	Отменить совершенное действие
Ctrl-^	Повторить отмененное действие
Ctrl-K E	Открыть файл
Ctrl-K R	Вставить файл
Ctrl-K D	Сохранить файл
Ctrl-K X	Выход с сохранением
Ctrl-C	Выход без сохранения
Ctrl-K Z	Выйти в shell

Глава 11

Обработка текстовой информации

11.1 Команда sed

Синтаксис:

`sed [-n] [-e команды] [-a ком_файл]файл...`

Описание:

Вызвать редактор для указанных файлов.

Опции:

<code>-e<i>команды</i></code>	Исполняются <i>команды</i> sed-скрипта
<code>-f<i>файл_программы</i></code>	Запустить sed -программу из файла
<code>-n</code>	Выводит только строки, указанные командой p

Командные последовательности:

Синтаксис команд аналогичен **ed** (см. [Разд. 10.1](#)).

11.2 Команда awk

Синтаксис:

`awk [-Fразделитель] ['программа'] [файл...]`

`awk [-Fразделитель] [-f файл_программы] [файл...]`

Описание:

Интерпретатор языка обработки текстовых потоков.

Каждая строка входного потока разбивается на поля согласно установленным разделителям — по умолчанию пробелы и табуляции. Эти умолчания могут быть изменены установкой переменной среды `FS`. Поля представляются переменными `$1`, `$2`...; `$0` обозначает всю строку. При изменении любого поля меняется и вся строка.

Программы имеют следующий формат:

```
BEGIN { начальные_операторы } { селектор действие ... } END { конечные_операторы }
```

Опции:

-F <i>разделитель</i>	Установить <i>разделитель</i> полей
-f <i>файл_программы</i>	Запустить awk -программу из <i>файла</i>

Переменные:

FS	Разделитель входных полей
RS	Разделитель входных строк
FILENAME	Имя текущего входного файла
NF	Количество полей в текущей строке
NR	Номер строки
OFMT	Формат для вывода чисел. По умолчанию — %bg
OFS	Разделитель выходных полей
ORS	Разделитель выходных строк

Операторы:

if (выражение) оператор [else оператор]
while (выражение) оператор
for (выражение; выражение; выражение) оператор
for (выражение in массив) оператор
break
continue
([оператор...])
<i>переменная</i> = <i>выражение</i>
print [список_выражений] [> >> файл]
print [список_выражений] [процесс]
printf [список_выражений] [> >> файл]
printf [список_выражений] [процесс]
next
exit (выражение)

Функции:

length(())	Длина строки, по умолчанию \$0
split(строка, массив, разделители)	Разбить строки
substr(строка, начало, длина)	Выделить подстроку
index(строка, подстрока)	Найти позицию <i>подстроки</i> в <i>строке</i> . В случае неудачи возвращается 0
getline	Прочитать следующую входную строку. В случае конца файла возвращается 1, иначе — 0
sprintf(формат, выражение, ...)	Форматный вывод в строку
exp(выражение)	Экспонента
log(выражение)	Натуральный логарифм
sqrt(выражение)	Квадратный корень
int(выражение)	Округление до ближайшего целого

Примеры:

```
$ cat > test
123 456 789
abc def ghi
~!@ #$$ % ^&*
```

```
$ awk '/abc/' < test
abc def ghi
$ awk 'END{print NR}' < test
3
$ awk '{print $2 $3}' < test
456789
defghi
#$$^&*
```

11.3 Команда `str`

Синтаксис:

`str [-s] файл1 файл2`

Описание:

Производится сравнение двух бинарных файлов. При их различии возвращается код ошибки и выводится номер байта, где это произошло.

Опции:

<code>-s</code>	Не выводить текстового сообщения
-----------------	----------------------------------

11.4 Команда `diff`

Синтаксис:

`diff [-befr] файл_или_каталог1 файл_или_каталог2`

Описание:

Найти различия в текстовых файлах и вывести их на стандартный вывод.

Опции:

<code>-b</code>	Игнорирование лишних пробельных символов
<code>-e</code>	Создать последовательность ed -команд (см. Разд. 10.1) для преобразования <i>файла1</i> в <i>файл2</i>
<code>-f</code>	Создать последовательность ed -команд (см. Разд. 10.1) для преобразования <i>файла2</i> в <i>файл1</i>
<code>-r</code>	Если аргументы — каталоги, выполнять команду рекурсивно

11.5 Команда `fold`

Синтаксис:

`fold [-b] [-s] [-w ширина] [файл...]`

Описание:

Разбивка длинных строк на строки требуемой длины.

Опции:

<code>-b</code>	Учитывать символы возврата каретки
<code>-s</code>	Разбивка по ближайшему пробелу
<code>-wширина</code>	Установить <i>ширину</i> выходной строки (по умолчанию — 80)

11.6 Команда sort

Синтаксис:

sort [-cmnr] [файл...]

Описание:

Сортировка файлов.

Опции:

<code>-c</code>	Проверка, отсортирован ли файл; выдача кода завершения
<code>-m</code>	Слияние отсортированных файлов
<code>-u</code>	Вывод только одной строки из нескольких одинаковых
<code>-n</code>	Численный порядок сортировки
<code>-r</code>	Обратный порядок сортировки

Примеры:

```
$ cat > test
009
01
2
abc
xyz
$ sort test
009
01
2
abc
xyz
$ sort -n test
abc
xyz
01
2
009
```

11.7 Команда uniq

Синтаксис:

uniq [-cdu] [-полей] [+символов] [входной_файл [выходной_файл]]

Описание:

Поиск повторяющихся строк. По умолчанию применяются опции `-d` и `-u`.

Опции:

<code>-полей</code>	Проигнорировать указанное число начальных <i>полей</i>
<code>+символов</code>	Проигнорировать указанное число начальных <i>символов</i>
<code>-с</code>	Каждой группе повторяющихся строк предшествует число повторений
<code>-d</code>	Выводить только первые строки из набора повторяющихся строк
<code>-u</code>	Вывести только неповторяющиеся строки

11.8 Команда tr

Синтаксис:

`tr [-cдs] строка1 [строка2]`

`tr -d|s [-с] строка`

`tr -d|s [-с] строка1 строка2`

Описание:

Транслитерация входного потока.

В первом случае транслитерация производится путем совершения подстановки символов из *строки2* на место соответствующих символов из *строки1*. Если длина *строки1* больше, то в соответствие остающимся символам ставится последний символ *строки2*.

Во втором случае происходит удаление указанных символов из входного потока.

Опции:

<code>-с</code>	Дополнение до набора, указанного в <i>строке1</i>
<code>-d</code>	Удалить все символы, соответствующие заданным в <i>строке1</i>
<code>-s</code>	Заменить на выходе все кратные вхождения литер из <i>строки2</i> одним символом

11.9 Команда join

Синтаксис:

`join [-а номер_файла] [-е строка] [-j [номер_файла] поле] [-о список_полей] [-t символ_разделитель] файл1|- файл2`

Описание:

Объединение строк файлов при наличии общих полей.

Опции:

<code>-аномер_файла</code>	Дополнительно вывести непарные строки файла 1 или 2
<code>-естрока</code>	Пустые поля заменять <i>строкой</i>
<code>-j[номер_файла]поле</code>	Выполнить сравнение по <i>полю файла</i> или, если файл пропущен, — по каждому файлу
<code>-о список_полей</code>	Выходные строки составлять согласно спискам, состоящим из элементов вида <i>номер_файла.поле</i>
<code>-t символ_разделитель</code>	Задание разделителей полей

11.10 Команда paste

Синтаксис:

paste [-s] [-d список_разделителей] файл. . .

Описание:

Слияние строк файлов. Строки файлов сливаются, разделяясь табуляцией.

Опции:

<code>-s</code>	Последовательное слияние — все строки файла сливаются в одну, разделяясь табуляцией, в конце ставится перевод строки, операция повторяется со следующим файлом
<code>-d список_разделителей</code>	Вместо табуляции для разделения полей по очереди используются символы из <i>списка_разделителей</i>

Глава 12

Программирование в среде командных интерпретаторов

Shell обладает средствами для написания исполняемых программных файлов. Он включает все основные средства обычных языков программирования высокого уровня: переменные, условные операторы, операторы цикла, подпрограммы. Вместе с тем, поддержка метасимволов и возможность использования мощного набора программ, существующих в UNIX, позволяет выполнять разработку полноценных программных продуктов в более короткий срок, чем при помощи традиционных языков программирования.

12.1 Запуск командных файлов

Простейший метод запуска командных файлов, написанных на каком-либо **shell**'е, — это вызвать соответствующий **shell**, задав ему в качестве параметра ваш программный файл:

```
shell файл параметр...
```

Например:

```
$ cat > hello
echo Hello word!
$ sh hello
Hello word!
^D
```

Интерпретатору можно задавать ключи, помогающие при отладке скриптов:

-v	Выводить больше информации при исполнении скрипта
-x	Отображать ход исполнения скрипта пошагово

При наличии в файле первой строки, указывающей текущий интерпретатор (по умолчанию `/bin/sh`), и прав доступа на исполнение, файл можно запускать на исполнение как любую другую программу:

```
$ cat > hello
echo Hello word!
$ sh hello
Hello word!
^D
```

```
$ chmod +x hello
$ ./hello
Hello word!
```

Можно запустить командный файл без порождения нового процесса. Главным следствием этого является то, что все установки переменных, сделанные в скрипте, будут иметь силу в текущем **shell**'е:

```
$ echo $BYE

$ echo 'BYE="Good Bye!"' >> hello
$ . ./hello
Hello word!
$ echo $BYE
Good Bye!
```

12.2 Позиционные параметры

В **shell** применяется набор зарезервированных переменных:

Переменная	Bourne-Shell	C-Shell	Korn-Shell и выше
Количество аргументов	<code>\$#</code>	<code>\$#argv</code>	<code>\$#</code>
Все аргументы	<code>\$*</code>	<code>*,\$argv*,\$argv[*]</code>	<code>\$*</code>
n-ый аргумент	<code>\$n</code>	<code>\$n,argv[n]</code>	<code>\$n,\$[n]</code>
Имя программы	<code>\$0</code>	<code>\$0</code>	<code>\$0</code>
Последний аргумент		<code>\$argv[\$#argv]</code>	

12.3 Основные конструкции и операторы командного языка

12.3.1 Условный интерпретатор `if`

Bourne-Shell:

```
if выражение; then команды; [else команды]; fi
```

C-Shell:

```
if выражение; then команды; [else if выражение команды]; [else команды]; endif
```

Korn-Shell:

```
if выражение; then команды; [elif выражение команды]; [else команды]; fi
```

Если *выражение* истинно, то выполняются *команды* первого уровня. Если же это не так, то либо выполняются *команды* второго уровня, стоящие после оператора **else**, либо вычисляется следующее *выражение*.

12.3.2 Оператор переключения

Bourne-Shell:

```
case строка in образец) команды ;; *) команды ;; esac
```

C-Shell:

switch (строка) case образец: команды breaksw default: команды breaksw endsw

Оператор работает как переключатель между различными альтернативами, описываемыми *образцами*. Если ни один из образцов не соответствует указанной *строке*, выполняются *команды*, указанные в умолчании. В образцах могут использоваться метасимволы.

12.3.3 Оператор цикла while

Bourne-Shell:

while выражение; do команды; done

C-Shell:

while выражение команды; end

Цикл, выполняющий *команды*, пока *выражение* является истинным.

12.3.4 Оператор цикла for

Bourne-Shell:

for переменная in список; do команды; done

C-Shell:

foreach переменная (список) команды; end

Цикл, выполняющий *команды*, выбирая элементы из *списка* и присваивая *переменной* значение этого элемента.

12.3.5 Оператор цикла until

Bourne-Shell:

until выражение; do команды; done

Цикл, выполняющий *команды*, пока *выражение* является ложным.

12.3.6 Вычисление выражений

Код завершения любой программы в условных интерпретаторах истолковывается как истина, если он равен 0, в противном случае — ложь. Существуют команды **true** и **false**, единственная функция которых состоит в том, чтобы возвращать истину или ложь соответственно.

Оператор **test** или его аналог **[]** (см. [Разд. 7.8.4](#)) часто используется для выполнения различных проверок, связанных с файловой структурой.

12.3.7 Определение подпрограмм

[function] *имя_функции* () {*операторы*}

Параметры, передаваемые функции, передаются как позиционные параметры. Возврат из подпрограммы осуществляется по завершении набора *операторов* с результирующим кодом завершения или при помощи оператора **return**.

12.4 Встроенные функции

12.4.1 basename — выделение компонентов имени файла

basename *строка* [*суффикс*]

Из *строки* удаляется любой префикс, кончающийся на '/' и указанный *суффикс*.

12.4.2 dirname — выделение имени каталога

dirname [*строка*]

Выделение из *строки* имени каталога. При отсутствии аргументов возвращается символ текущего каталога (.).

12.4.3 echo — вывести на печать

echo [-n] [*аргумент...*]

В стандартный вывод помещаются *аргументы*, разделенные пробелами. Если не указана опция -n, завершается переводом строки.

12.4.4 eval — выполнить команду

eval [*оператор...*]

Происходит выполнение указанных *операторов* при выполнении всех указанных подстановок.

12.4.5 exec — запустить программу на исполнение

exec [*команда*] [*аргумент...*]

Происходит исполнение *команды* с *аргументами* без порождения нового процесса и возврата в текущий **shell**.

12.4.6 espr — вычисление арифметических выражений

espr *выражение*

Вычисляется арифметическое выражение, результаты записываются в стандартный вывод.

12.4.7 read — прочесть строку

read *переменная...*

Происходит чтение строки из стандартного ввода. Полученные слова последовательно присваиваются указанным переменным. Остаток строки присваивается последней переменной.

12.4.8 readonly — защита переменных от изменения

readonly [*переменная...*]

Запрещение записи в указанные командные *переменные*. При отсутствии аргументов выводится список защищенных переменных.

12.4.9 shift — сдвиг позиционных параметров

shift [*сдвиг*]

Сдвиг на указанное число параметров. По умолчанию — 1.

Часть III

Сетевое окружение ОС Linux/UNIX

Важнейшими задачами, решаемыми при удаленной работе по сети, со времен распространения протоколов для связи компьютеров по коммутируемым телефонным линиям UUCP и первых версий протокола TCP/IP, являются передача данных и удаленное исполнение команд. Наиболее распространенным средством для решения таких задач сейчас является набор утилит SSH (Secure Shell), осуществляющих передачу данных и исполнение программ по специальному зашифрованному протоколу. В некоторых случаях могут оказаться полезными и аналогичные средства более ранних поколений — telnet, ftp, rcp, rhex и пр.

Как правило, организация передачи данных при использовании таких средств гораздо проще, чем при использовании распространенных сетевых файловых систем типа NFS или SAMBA, а накладные расходы при передаче данных по сети — меньше.

Большую помощь в организации удаленной работы по сети оказывают специализированные программные компоненты, позволяющие поддерживать в рабочем состоянии рабочие сессии как в текстовом (напр. screen), так и в графическом режиме (например, VNC) даже при отключении сетевого соединения.

Глава 13

Утилиты из набора SSH (Secure Shell)

Утилиты из этого набора осуществляющих передачу данных и исполнение программ по специальному зашифрованному протоколу, часто используются другими программами для создания зашифрованных каналов для передачи данных. Требуют наличия запущенного сервиса **sshd**.

13.1 Вход на удаленный хост — ssh

Синтаксис:

```
ssh [-C] [-p порт] [-v] [-Y] [пользователь@]хост [команда]
```

Описание:

Команда осуществляет терминальный вход указанным пользователем (по умолчанию тем же) на указанный хост по указанному порту. Если при вызове указана команда, она будет выполнена на удаленном хосте.

Опции:

-C	Осуществлять компрессию данных при передаче
-p порт	Порт, по которому производится соединение
-Y	Включить доверенное перенаправление сетевых пакетов графической системы X-Window
-v	Выводить отладочную информацию

13.2 Обмен данными с удаленным хостом — scp

Синтаксис:

```
scp [-Cprv] [-P порт] [пользователь@]хост:[путь] [...] [пользователь@]хост:[путь]
```

Описание:

Осуществить передачу файлов и каталогов с компьютеров-источников на компьютер-приемник. Если компьютер не указан, по умолчанию используется локальная машина. Если не указан пользователь, —

для аутентификации используются имя пользователя, запустившего данную команду. В том случае, если не указан каталог на принимающей стороне, используется домашний каталог пользователя.

Опции:

-c	Осуществлять компрессию данных при передаче
-P	Сохранять время модификации и доступа, а также моды доступа
-P <i>порт</i>	Порт, по которому производится соединение
-r	Рекурсивное копирование каталогов
-v	Выводить отладочную информацию

Глава 14

Традиционные сетевые утилиты

Исторически наиболее ранними утилитами, обеспечивавшими функциональность, необходимую для передачи данных и удаленного исполнения команд, являются **telnet** и **ftp**. В настоящее время эти средства могут оказаться полезными для работы со старыми программными системами и некоторыми аппаратными комплексами. Требуют наличия соответствующих настроенных серверов на принимающей стороне.

Простейший пользовательский почтовый агент **mail** может быть полезен для автоматизации процесса отправки почты.

14.1 Вход на удаленный хост — telnet

Синтаксис:

telnet [хост] [порт]

Описание:

Команда осуществляет терминальный вход на указанный хост по указанному порту. При отсутствии аргументов или при вводе escape-символа (упоминается в сообщении, выдаваемом при запуске, обычно 'Ctrl-] — ^)'), telnet переходит в командный режим. В командном режиме можно получить подсказку о поддерживаемых командах, набрав '?' или 'help'.

Основные команды:

<code>close</code>	Закрывает текущее соединение
<code>logout</code>	Выход из сессии и закрытие соединения
<code>display</code>	Показывает текущие параметры
<code>mode</code>	Установка моды ввода символов и строк
<code>open хост</code>	Открывает соединение
<code>quit</code>	Выход из программы
<code>send</code>	Послать спец. символ
<code>set</code>	Установить параметры
<code>unset</code>	Сбросить параметры
<code>status</code>	Вывести текущий статус
<code>toggle</code>	Переключить параметры
<code>slc</code>	Установить режим обработки спец. символов
<code>z</code>	Подвесить сессию
<code>!</code>	Запустить внешнюю программу
<code>environ</code>	Изменить состояние переменных
<code>bye</code>	Выход из программы

14.2 Обмен файлами — ftp

Синтаксис:

ftp [хост]

Описание:

Команда позволяет пересылать файлы между машинами в сети в интерактивном режиме по протоколу FTP. При отсутствии аргументов программ переходит в командный режим, не осуществляя входа на удаленный *хост*. Подсказку можно получить, набрав '?' или 'help'.

Основные команды:

open <i>хост</i>	Открыть соединение с указанным <i>хостом</i>
close	Закрывает текущее соединение
ls [<i>каталог</i>]	Получить листинг файлов в <i>каталоге</i> на удаленной машине. Если каталог не указан, выводится листинг текущего каталога
cd <i>каталог</i>	Перейти в <i>каталог</i> на удаленной машине
lcd <i>каталог</i>	Перейти в <i>каталог</i> на локальной машине
pwd	Вывести текущий каталог на локальной машине
get <i>файл</i>	Получить указанный <i>файл</i>
put <i>файл</i>	Отправить указанный <i>файл</i>
delete <i>файл</i>	Удалить указанный <i>файл</i> на удаленной машине
reget <i>файл</i>	Продолжить получение указанного <i>файла</i> с прерванного места
mget <i>файлы</i>	Получить указанные <i>файлы</i>
mput <i>файлы</i>	Передать указанные <i>файлы</i>
mkdir <i>каталог</i>	Создать указанный <i>каталог</i> на удаленной машине
rmdir <i>каталог</i>	Удалить указанный <i>каталог</i> на удаленной машине
prompt	Отключить/включить режим подтверждения для операций mget и mput
hash	Отображать ход пересылки символами '#'

14.3 Отправка и чтение электронной почты — mail

Синтаксис:

mail [-f] [-s *тема*] [*пользователь*[@*хост*]]

Описание:

Программа служит отправки и получения электронной почты. При отсутствии аргументов происходит чтение из почтового ящика пользователя поступившей ему корреспонденции, указание имени пользователя без указания хоста говорит об отправке e-mail *пользователю* на локальной машине.

Посылка сообщения может производиться либо из стандартного ввода (в том числе, путем перенаправления ввода из файла или программного потока), либо из файлов в командном режиме. Ввод письма заканчивается точкой в начале пустой строки. Процесс работы в командном режиме можно прервать соответствующими командами или стандартными средствами прерывания процессов.

Опции:

-f	Просмотреть сохраненные письма
-s <i>тема</i>	Указание <i>темы</i> письма, помещаемого в поле Subject:

Основные команды:

~r <i>файл</i>	Прочсть текст письма из <i>файла</i>
~p	Показать текст набираемого письма
<i>номер</i>	Вывести письмо с указанным <i>номером</i>
ENTER, +	Вывести последующее письмо
-	Вывести предыдущее письмо
s <i>файл</i>	Сохранить текст письма в <i>файле</i>
d <i>номера</i>	Удалить письма под указанными <i>номерами</i>
q	Выйти с сохранением изменений
x	Выйти без сохранения
?	Получить подсказку

Глава 15

Средства получения информации из сети Интернет

Важной задачей при работе с Интернет-источниками является массовое копирование и обработка данных из них, а также автоматическое зеркалирование (mirroring) таких ресурсов или их частей.

15.1 Текстовый веб-браузер — lynx

Синтаксис:

lynx [-dump] [URL_интернет-ресурса]

Описание:

Данная программа позволяет интерактивно работать с Интернет-ресурсами, отдаваемым по протоколам HTTP и FTP, в текстовом режиме. Поддержка фреймов минимальна, JavaScript не поддерживается.

Опции:

-dump	Переводит полученные данные в текстовый вид, выводит текст на стандартный вывод и завершает свою работу
-------	---

Основные команды:

q	Завершение работы программы
Стрелка вверх, вниз	Перемещение
Стрелка вправо	Переход по ссылке
Стрелка влево	Возврат
h	Help — подсказка
o	Options — установки
p	Print — печать
G	Go — переход на указанный URL
M	Main — переход на главную страницу
D	Download — загрузка и сохранение ресурса указанного в ссылке
/	Поиск на странице

15.2 wget

Синтаксис:

```
wget [-c] [-m] [-np|--no-parent] [-k|--convert-links] [--follow-ftp] [-A|--accept
список_разрешенных_суффиксов] [-R|--reject список_запрещенных_суффиксов] URL
```

Описание:

Программа позволяет осуществлять выкачивание Интернет-ресурсов или их частей по протоколам HTTP, HTTPS и FTP. Поддерживается передача данных с использованием проху-серверов.

Опции:

-c	Осуществлять докачку, в случае обрыва передачи данных
-m	Зеркалирование (mirroring) ресурса, с учетом ссылок
-np --no-parent	При рекурсивном выкачивании ресурса не подниматься выше уровня указанного ресурса
-k --convert-links	Автоматически конвертировать ссылки для локального просмотра
--follow-ftp	Переходить по ссылкам на FTP-ресурсы. По умолчанию такой переход не осуществляется
-A --accept суффиксы	Разделенный запятыми список суффиксов файлов, разрешенных к выкачиванию. Могут использоваться метасимволы '*', '?', '[' и ']'
-R --reject суффиксы	Аналогично для списка суффиксов файлов, запрещенных к выкачиванию

Переменные окружения:

Для указания проху-серверов используются переменные: `http_proxy`, `https_proxy` и `ftp_proxy`, для протоколов HTTP, HTTPS и FTP соответственно. Например:

```
export ftp_proxy=http://proxy.my.org:3128/
```

15.3 lftp

Синтаксис:

```
lftp [-d] [-e команда] [-p порт] [-u пользователь[,пароль]] [хост]
```

```
lftp -f командный_файл
```

```
lftp -c команды
```

Описание:

Программа для обмена данными по сети Интернет по протоколам FTP, FTPS, FISH, HFTP, HTTP, HTTPS и SFTP с набором команд, во многом аналогичном классическому ftp.

Опции:

-d	Включить выдачу отладочной информации
-e команда	Выполнить указанные команды и не выходить
-p порт	Осуществлять соединение по указанному <i>порту</i>
-u <i>пользователь, пароль</i>	Использовать для входа имя <i>пользователя</i> и <i>пароль</i> , если он указан
-f файл	Выполнить команды из <i>файла</i> и выйти
-c команды	Выполнить указанные <i>команды</i> и выйти

Основные команды:

openхост	Открыть соединение с указанным <i>хостом</i>
close	Закрывает текущее соединение
ls[каталог]	Получить листинг файлов в <i>каталоге</i> на удаленной машине. Если каталог не указан, выводится листинг текущего каталога
cdкаталог	Перейти в <i>каталог</i> на удаленной машине
lcdкаталог	Перейти в <i>каталог</i> на локальной машине
pwd	Вывести текущий каталог на локальной машине
get[-с]файл	Получить указанный <i>файл</i> . При указании ключа -р происходит докачивание данных
mirror [источник] [приемник]	Провести зеркальное копирование, включая подкаталоги, <i>источника</i> в <i>приемник</i>
put[-с]файл	Отправить указанный <i>файл</i> . При указании ключа -р происходит докачивание данных
deleteфайл	Удалить указанный <i>файл</i> на удаленной машине
regetфайл	Продолжить получение указанного <i>файла</i> с прерванного места
mget[-с]файлы	Получить указанные <i>файлы</i> . При указании ключа -р происходит докачивание данных
mput[-с]файлы	Передать указанные <i>файлы</i> . При указании ключа -р происходит докачивание данных
mkdir[-р]каталог	Создать указанный <i>каталог</i> на удаленной машине. При указании ключа -р происходит создание всей цепочки вложенных каталогов
rmdirкаталог	Удалить указанный <i>каталог</i> на удаленной машине
prompt	Отключить/включить режим подтверждения для операций mget и mput

15.4 rsync

Синтаксис:

```
rsync [-avz] [--delete] [пользователь@]хост:[путь_до_файла_или_каталога]
```

Описание:

Программа **rsync** используется для пересылки по сети файлов и каталогов с одного компьютера на другой. Важным свойством **rsync**, отличающим эту команду от других подобных ей, является то, что она пересылает не весь файл целиком, а только ту ее часть, которая отличает версию файла на принимающем компьютере от версии на передающем компьютере.

Опции:

<code>-a</code>	Архивная мода, что означает рекурсию, сохранение линков, времени создания файлов, прав доступа к файлам, пользователя и группы файлов, файлов устройств
<code>-v</code>	Отображать ход передачи данных
<code>-z</code>	Осуществлять компрессию данных при передаче
<code>--delete</code>	Удалить на компьютере-приемнике все файлы и каталоги, удаленные на компьютере источнике

Часть IV

Графическая среда ОС Linux/UNIX

Графическая система X-Window была создана в 1984 году в MIT. Текущая версия X Version 11 (X11) была реализована в 1987 г. Система представляет собой сетевую графическую систему и является промышленным стандартом. Разработкой X-Window занимался консорциум, состоявший из корпораций AT&T, DEC, HewlettPackard, IBM и Sun. Теперь поддержка и развитие осуществляется комитетом X/Open.

При входе в сессию X-Window вы видите рабочее поле с некоторым набором окон, порожденных процессами, запущенными в ходе исполнения вашего персонального стартового скрипта, или системного, в том случае если личный скрипт отсутствует.

Сам сервер и все отображаемые окна представляют собой отдельные процессы, которые можно увидеть и оценить их состояние при помощи команды **ps** (см. [Разд. 8.1](#)), послать им сигналы остановки и прерывания командой **kill** (см. [Разд. 8.5](#)).

Одно из окон (в том числе и корневое окно рабочего поля) может находиться в «фокусе», в этом случае оно может перехватывать события, генерируемые устройствами ввода — мышью, клавиатурой и т.д. Перемещение фокуса может происходить самым различным образом, определяемым текущим window manager'ом — традиционное щелчком мыши, реагируя на перемещение мыши над окном, на «горячие клавиши» и т.д. Любое приложение, запущенное на одном компьютере, может осуществлять полнофункциональный графический вывод на любом другом компьютере, поддерживающем систему X-Window вне зависимости от архитектур и типов операционных систем.

X-Window не содержит GUI (Grahical User Interface) и не диктует стиля оформления Desktop'a или приложений. Исходно система предоставляет несколько базовых интерфейсов для создания приложений: библиотеку нижнего уровня Xlib и набор X Toolkit, который в свою очередь состоит из X Toolkit Intrinsics (Xt) и Athena Widget set (Xaw).

Последний является набором элементарных объектов, из которых строятся многие приложения. Именно разные наборы widget'ов определяют стиль оформления. В частности, Athena Widget set задает специфические механизмы управления графическими элементами, не слишком привычные для пользователей современных графических интерфейсов и на сегодняшний день выглядящие несколько архаично, в частности — двумерный характер виджетов (кнопок, областей прокрутки и т.п.). Достаточно неожиданно выглядит поведение областей прокрутки: первая кнопка мыши отвечает за прокрутку вперед, третья — назад, а средняя в нажатом состоянии — прокручивание в произвольном направлении. Кроме Athena создано большое количество наборов widget'ов, включая наиболее распространенные в настоящий момент GTK и Qt.

Запуск сессии X-Window может происходить прямо при загрузке системы при помощи демонов gdm, kdm, xdm и др. или из текстовой сессии пользователя при помощи командных скриптов startx или xinit.

Глава 16

Основные понятия X-Window

16.1 Устройства ввода

Система X-Window требует наличия 3-х кнопочной мыши. При ее отсутствии X-Window разных производителей предлагают использовать различные комбинации: одновременное нажатие на 1 и 2 кнопки мыши, Shift+1 кнопка мыши и т.п.

16.2 Понятие DISPLAY

X-Window является системой, ориентированной на работу в сети. Уникальным идентификатором графического экрана, на котором происходит отображение работы приложения X-Window, является дисплей, задаваемый переменной окружения `$DISPLAY` или опцией `-display`, указанной при запуске приложения. Этот идентификатор имеет следующий вид:

`[имя_хоста]:номер_дисплея[.номер_экрана]`

где:

- имя_хоста — имя машины, к которой физически подключен дисплей в сети. Если имя не указано, умолчанием принимается локальный хост;
- номер_дисплея — уникальный номер набора физических дисплеев, управляемых одними устройствами ввода;
- номер_экрана — идентификатор конкретного физического дисплея.

16.3 Шрифты X-Window

Уникальное имя шрифта X-Window выглядит следующим образом:

`прзв-сем-вес-нкл-тлщ-д_с-пкс-пкт-г_р-в_р-спс-с_т-код-стр`

где:

- прзв — производитель
- сем — семейство
- вес — вес

- нкл — наклон
- тлщ — толщина
- д_с — добавочный стиль
- пкс — пиксели
- пкт — пункты
- г_р — горизонтальное разрешение
- в_р — вертикальное разрешение
- spc — spacing
- с_т — средняя толщина
- код — кодировка
- стр — кодовая страница

В спецификациях шрифтов могут использоваться метасимволы '*' и '?'. Просмотреть список доступных шрифтов можно при помощи утилиты **xlsfonts**. Выбрать необходимый — **xfontsel**. Просмотр — **xfd**. Манипуляции с путем поиска шрифтов можно производить с помощью утилиты **xset**.

В современных графических системах некоторыми приложениями (например OpenOffice.org) используется также другая система управления шрифтами — FontConfig, с более традиционными для Windows и MacOS формами именования шрифтов по его семейству и размеру. Основные утилиты для работы с данным механизмом — **fc-cache**, **fc-list**, **fc-cat** и **fc-match**.

16.4 Цвета X-Window

Цвета X-Window доступны как в виде символьных имен, так и в виде числовых значений. Соответствие первых последним устанавливается в файле `rgb.txt`. Получить карту таких соответствий можно при помощи команды **showrgb**.

В числовом виде цвета могут задаваться в цветовом пространстве RGB в форматах:

#RGB

#RRGGBB

#RRRGGBBB

#RRRRGGGGBBBB

где R, G и B — шестнадцатеричные числа, характеризующие интенсивности красной, зеленой и голубой составляющей.

16.5 Ресурсы X-Window

Важнейшим средством управления свойствами и поведением приложения являются ресурсы X-Window. Они представляют собой объектно-ориентированную структуру, состоящую из составных идентификаторов и значений величин:

объект.подобъект[.подобъект...].атрибут: значение где:

- объект — программа-клиент или имя, установленное в опции **-name**;

- подобъект — последовательные уровни в иерархии виджетов (обычно основные составляющие приложения — окна, меню, scrollbar'ы и т.д.);
- атрибут — свойства последнего подобъекта, например, цвет или надпись;
- значение — величина, устанавливаемая для атрибута, т.е. цвет, текст или пр.

В именах объектов, подобъектов и атрибутов могут использоваться метасимволы '*' и '?'.

Обычно необходимые ресурсы устанавливаются при старте X-Window из файла \$HOME/.Xdefaults или \$HOME/.Xresources, ими можно манипулировать при помощи утилит **appres**, **xrdb**, **editres** и стандартных опций X-Toolkit.

В графической системе GNOME используется собственный механизм поддержки ресурсов GConf.

16.6 Стандартные опции Xt

Приложения X-Window, собранные с поддержкой библиотеки Xt, по умолчанию поддерживают следующий набор опций:

Опции:

-display <i>дис-плея</i>	Дисплей, на котором будет работать запускаемое приложение. Доступ к этому дисплею должен быть разрешен, например командой xhost
-geometry <i>шир.хвыс.+ -X+ -Y</i>	Задаёт размеры и положение окна приложения в виде положительного или отрицательного сдвига по осям X и Y
-bg background <i>цвет</i>	Цвет фона
-bg foreground <i>цвет</i>	Цвет символов
-bd bordercolor <i>цвет</i>	Цвет окантовки
-bw borderwidth <i>ширина</i>	Ширина окантовки в пикселах
-fn font <i>шрифт</i>	Шрифт, используемый приложением
-iconic	Вызванное приложение появится сразу свернутым в иконку
-name <i>имя</i>	Имя приложения, под которым будут специфицированы ресурсы
-rv reverse	Указание программе проинвертировать фон и картинку. Не все приложения могут обрабатывать эту опцию корректно
+rv	Указание не пытаться проинвертировать изображение, если это установлено по умолчанию
-title <i>заголовок</i>	Заголовок приложения, обычно используемый window manager'ом
-xnl language	Установка параметров локализации для ресурсов и имен файлов
-xrm	Строка ресурсов, передаваемая непосредственно приложению

Глава 17

Базовые приложения X-Window

17.1 Xserver

Главным компонентом в системе X-Window является **Xserver**. Он отвечает за взаимодействие с оборудованием, обрабатывает события от устройств ввода, передает их подключившимся к нему приложениям и отрисовывает на графическом устройстве (оно может быть виртуальным) графические примитивы, передаваемые приложениями.

Опции общего назначения:

:номер_дисплея	Запустить Xserver на дисплее, номер которого указан (по умолчанию — 0)
-a скорость	Установка скорости перемещения мыши
-auth файл_авторизации	Файл, содержащий записи для авторизации
-c	Выключить щелчки при нажатии на клавиатуру
c громкость	Установка громкости щелчков (0-8)
-co файл	Указать имя файла с указанием соответствия символьных имен цветам, заданным в цифровом виде в формате <code>rgb.txt</code>
-dpi разрешение	Установка разрешения экрана в точках на дюйм. Устанавливается тогда, когда сервер не может определить эту величину автоматически
-f громкость	Установить громкость звукового сигнала (0-7)
-fc шрифт_курсора	Установить умолчание для шрифта курсора
-fn шрифт	Установить умолчание для шрифта
-fp путь	Указать путь поиска шрифтов в виде списка ресурсов и каталогов, разделенных пробелами
-help	Вывести справку
-I	Проигнорировать все оставшиеся элементы командной строки
-logo	Включить отображение X-Window logo при запуске screensaver'a
nologo	Отключить отображение X-Window logo при запуске screensaver'a
-p минут	Установить время цикла screensaver'a
-r	Включить автоповтор
r	Отключить автоповтор
- минут	Установить время запуска screensaver'a

Сетевые опции (XDMCP-протокола):

-query хост	Послать запрос на соединение по XDMCP-протоколу на указанный хост
-broadcast	Послать широковещательный XDMCP запрос. Подключение производится к первому ответившему хосту
-indirect хост	Послать XDMCP запрос IndirectQuery на указанный хост. Отображаются все видимые по XDMCP хосты с возможностью подключения к ним
-port порт	Указать нестандартный порт для обмена по протоколу XDMCP
-once	Завершить работу сервера по окончании сессии
-class класс_дисплея	Указать дополнительный идентификатор в пространстве ресурсов (по умолчанию "MIT-Unspecified")

Сервера поддерживающие механизм XKB могут вызываться со следующими опциями:

[+-]kb	Включает(+) или отключает(-) использование расширения XKEYBOARD
-xkbdirкаталог	Базовый каталог с описаниями клавиатур
-noloadxkb	Отменить загрузку XKB-keymap при старте сервера
-xkbdbфайл	Файл используется в качестве раскладки клавиатуры по умолчанию
-xkbmapфайл	Загрузка указанного в файле описания клавиатуры

При запуске X-сессии при помощи скриптов **startx** или **xinit** описанные выше параметры можно передать после разделителя в виде двух стоящих подряд знаков '-', например:

```
startx -- :2 -fp /usr/share/fonts/ms
```

17.2 Window Manager

Первым приложением, с которым обычно сталкивается пользователь, является Window Manager. Именно он позволяет интерактивно управлять размещением и размером окон, сворачиванием их в иконки, задает стиль их оформления и обработку событий, посылаемых устройствами ввода. Со стандартным MIT дистрибутивом поставляется twm (см. Рис. 17.1).

Запуск сессий современных графических пользовательских интерфейсов KDE и GNOME непременно сопровождается запуском соответствующих Window Manager'ов: **kdeinit** запускает **kwin**, а **gnome-session** может стартовать любой совместимый с ним Window Manager, обычно используется **metacity**.

17.3 Эмулятор терминала — xterm

Синтаксис:

```
xterm [-e команда [аргументы...]]
```

Описание:

Эмулятор терминала **xterm** разрабатывался как основное средство, осуществляющее связь между традиционным интерфейсом командной строки (CLI) и графическим интерфейсом X-Window. Для обеспечения правильной работы приложений переменной окружения **\$TERM** должно быть присвоено значение **xterm**. Приложение эмулирует работу алфавитно-цифрового терминала VT102 и графического — Tektronix 4014. По умолчанию **xterm** стартует в алфавитно-цифровом режиме.

В графических окружениях GNOME и KDE есть функциональные аналоги **xterm** — **gnome-terminal** и **konsole** соответственно.

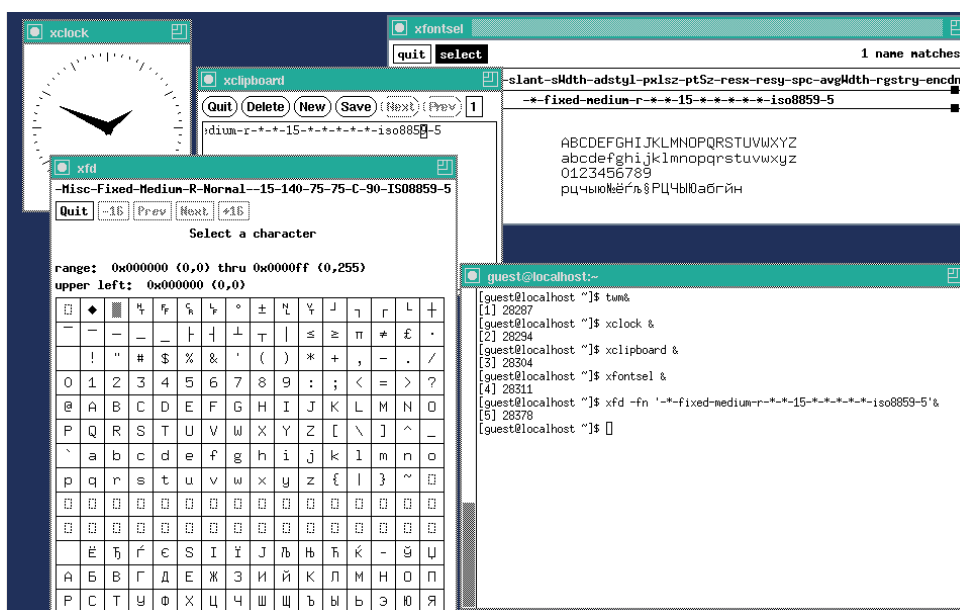


Рис. 17.1. Сессия TWM

Опции:

-e команда [аргументы. . .]	Выполнить указанную команду с аргументами
------------------------------------	---

Основные командные последовательности:

SHIFT+PageUp	Прокрутить окно на страницу вверх
SHIFT+PageDown	Прокрутить окно на страницу вниз
Ctrl+Button1	Вывести меню "Main Options". В данном меню можно перерисовать окно, послать сигнал прерывания текущему процессу и выйти из xterm
Ctrl+Button2	Позволяет управлять параметрами эмулятора терминала, производить сброс терминала, переключаться между текстовыми и графическими модами эмуляции и пр.
Ctrl+Button3	Выбор размера шрифта

17.4 Уничтожение окон программ-клиентов — xkill

Описание:

После запуска данного приложения вид курсора меняется на крестик, после щелчка которым по какому-либо окну оно завершается.

17.5 Получение информации об окне — xwininfo

Описание:

После запуска данного приложения вид курсора меняется на крестик, после щелчка которым по какому-либо окну выводится информация о нем.

Пример:

```
$ xwininfo
```

```
xwininfo: Please select the window about which you
         would like information by clicking the
         mouse in that window.
```

```
xwininfo: Window id: 0x270001f "guest@localhost:/home/guest"
```

```
Absolute upper-left X:  0
Absolute upper-left Y: 30
Relative upper-left X:  0
Relative upper-left Y: 24
Width: 1280
Height: 763
Depth: 24
Visual Class: TrueColor
Border width: 0
Class: InputOutput
Colormap: 0x20 (installed)
Bit Gravity State: NorthWestGravity
Window Gravity State: NorthWestGravity
Backing Store State: NotUseful
Save Under State: no
Map State: IsViewable
Override Redirect State: no
Corners:  +0+30  -0+30  -0-7  +0-7
-geometry 157x41+0+6
```

17.6 Получение списка клиентов — xlsclients

Описание:

Команда выводит список программ-клиентов, подключившихся к данному X-серверу.

Пример:

```
$ xlsclients
localhost.localdomain  gnome-session
localhost.localdomain  scim-panel-gtk
localhost.localdomain  gnome-settings-daemon
localhost.localdomain  vino-server
localhost.localdomain  metacity
localhost.localdomain  gnome-panel
localhost.localdomain  gnome-volume-manager
localhost.localdomain  nautilus
localhost.localdomain  eggccups
localhost.localdomain  bt-applet
localhost.localdomain  gnome-power-manager
localhost.localdomain  nm-applet
localhost.localdomain  pam-panel-icon
localhost.localdomain  puplet
```

```
localhost.localdomain  sealert
localhost.localdomain  WindowNavigationApplets
localhost.localdomain  trashapplet
localhost.localdomain  multiloader
localhost.localdomain  NotificationArea
localhost.localdomain  ClockApplet
localhost.localdomain  mixer_applet2
localhost.localdomain  stickynotes_applet
localhost.localdomain  cpufreq-applet
localhost.localdomain  gweather
localhost.localdomain  vm-applet
localhost.localdomain  gnome-dictionary-applet
localhost.localdomain  gnome-screensaver
localhost.localdomain  notification-daemon
localhost.localdomain  firefox
localhost.localdomain  gnome-terminal
localhost.localdomain  evince
localhost.localdomain  npviewer.bin
localhost.localdomain  eog
```

17.7 Получение информации о клиенте — `xdpyinfo`

Описание:

Вывод информации о текущем дисплее.

Пример:

```
$ xdpyinfo
name of display:      :0.0
version number:      11.0
vendor string:       The X.Org Foundation
vendor release number: 70101000
X.Org version:       7.1.1
maximum request size: 16777212 bytes
motion buffer size:  256
bitmap unit, bit order, padding: 32, LSBFirst, 32
image byte order:    LSBFirst
number of supported pixmap formats: 7
supported pixmap formats:
  depth 1, bits_per_pixel 1, scanline_pad 32
  depth 4, bits_per_pixel 8, scanline_pad 32
  depth 8, bits_per_pixel 8, scanline_pad 32
  depth 15, bits_per_pixel 16, scanline_pad 32
  depth 16, bits_per_pixel 16, scanline_pad 32
  depth 24, bits_per_pixel 32, scanline_pad 32
  depth 32, bits_per_pixel 32, scanline_pad 32
keycode range:      minimum 8, maximum 255
focus: window 0x2700020, revert to Parent
number of extensions: 32
  BIG-REQUESTS
  Composite
```

```
DAMAGE
DOUBLE-BUFFER
DPMS
Extended-Visual-Information
GLX
MIT-SCREEN-SAVER
MIT-SHM
MIT-SUNDRY-NONSTANDARD
RANDR
RECORD
RENDER
SECURITY
SGI-GLX
SHAPE
SYNC
TOG-CUP
X-Resource
XC-APPGROUP
XC-MISC
XFIXES
XFree86-Bigfont
XFree86-DGA
XFree86-DRI
XFree86-Misc
XFree86-VidModeExtension
XINERAMA
XInputExtension
XKEYBOARD
XTEST
XVideo
default screen number:    0
number of screens:       1

screen #0:
  dimensions:      1280x800 pixels (261x163 millimeters)
  resolution:      125x125 dots per inch
  depths (7):      24, 1, 4, 8, 15, 16, 32
  root window id:  0x5d
  depth of root window: 24 planes
  number of colormaps:  minimum 1, maximum 1
  default colormap: 0x20
  default number of colormap cells: 256
  preallocated pixels:  black 0, white 16777215
  options:          backing-store NO, save-unders NO
  largest cursor:   64x64
  current input event mask: 0xfa6033
    KeyPressMask           KeyReleaseMask           EnterWindowMask
    LeaveWindowMask        ButtonMotionMask         KeymapStateMask
    StructureNotifyMask    SubstructureNotifyMask  SubstructureRedirectMask
    FocusChangeMask        PropertyChangeMask       ColormapChangeMask
  number of visuals: 17
  default visual id: 0x23
  visual:
```

```

visual id:    0x23
class:       TrueColor
depth:       24 planes
...

```

17.8 Список шрифтов — xlsfonts

Описание:

Вывод списка шрифтов, доступных серверу.

Пример:

```

-adobe-courier-bold-o-normal--0-0-100-100-m-0-iso8859-2
-adobe-courier-bold-o-normal--0-0-100-100-m-0-iso8859-2
-adobe-courier-bold-o-normal--0-0-75-75-m-0-iso8859-2
-adobe-courier-bold-o-normal--0-0-75-75-m-0-iso8859-2
-adobe-courier-bold-o-normal--0-0-75-75-m-0-koi8-ub
-adobe-courier-bold-o-normal--10-100-75-75-m-60-iso10646-1
-adobe-courier-bold-o-normal--10-100-75-75-m-60-iso8859-1
-adobe-courier-bold-o-normal--10-100-75-75-m-60-iso8859-2
-adobe-courier-bold-o-normal--10-100-75-75-m-60-iso8859-2
-adobe-courier-bold-o-normal--10-100-75-75-m-60-iso8859-9
-adobe-courier-bold-o-normal--10-100-75-75-m-60-koi8-r
-adobe-courier-bold-o-normal--10-100-75-75-m-60-koi8-r
-adobe-courier-bold-o-normal--10-100-75-75-m-60-koi8-u
-adobe-courier-bold-o-normal--10-100-75-75-m-60-koi8-u
-adobe-courier-bold-o-normal--10-100-75-75-m-60-koi8-ub
-adobe-courier-bold-o-normal--10-100-75-75-m-60-koi8-ub
-adobe-courier-bold-o-normal--11-80-100-100-m-60-iso10646-1
-adobe-courier-bold-o-normal--11-80-100-100-m-60-iso8859-1
-adobe-courier-bold-o-normal--11-80-100-100-m-60-iso8859-2
-adobe-courier-bold-o-normal--11-80-100-100-m-60-iso8859-2
-adobe-courier-bold-o-normal--11-80-100-100-m-60-iso8859-9
-adobe-courier-bold-o-normal--12-120-75-75-m-70-iso10646-1
-adobe-courier-bold-o-normal--12-120-75-75-m-70-iso8859-1
-adobe-courier-bold-o-normal--12-120-75-75-m-70-iso8859-2
-adobe-courier-bold-o-normal--12-120-75-75-m-70-iso8859-2
-adobe-courier-bold-o-normal--12-120-75-75-m-70-iso8859-9
-adobe-courier-bold-o-normal--12-120-75-75-m-70-koi8-r
-adobe-courier-bold-o-normal--12-120-75-75-m-70-koi8-r
-adobe-courier-bold-o-normal--12-120-75-75-m-70-koi8-u
-adobe-courier-bold-o-normal--12-120-75-75-m-70-koi8-u
-adobe-courier-bold-o-normal--12-120-75-75-m-70-koi8-ub
-adobe-courier-bold-o-normal--12-120-75-75-m-70-koi8-ub
-adobe-courier-bold-o-normal--14-100-100-100-m-90-iso10646-1
-adobe-courier-bold-o-normal--14-100-100-100-m-90-iso8859-1
-adobe-courier-bold-o-normal--14-100-100-100-m-90-iso8859-2
-adobe-courier-bold-o-normal--14-100-100-100-m-90-iso8859-2
-adobe-courier-bold-o-normal--14-100-100-100-m-90-iso8859-9
-adobe-courier-bold-o-normal--14-140-75-75-m-90-iso10646-1
...

```

17.9 Утилита выбора шрифтов — xfontsel

Описание:

Утилита, позволяющая осуществлять выбор шрифта интерактивно. Нажатие на кнопку "Select" помещает выбранную строку описания шрифта в текстовый буфер обмена (см. Рис. 17.2).

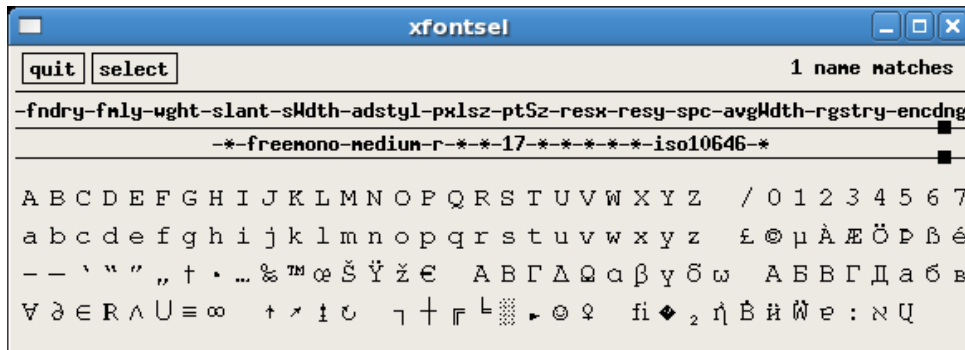


Рис. 17.2. Программа xfontsel

17.10 Средство просмотра шрифтов — xfd

Описание:

Служит для просмотра шрифтов, в том числе и в много-байтных кодировках (см. Рис. 17.3). Шрифт указывается через стандартные опции X Toolkit `-fn` или `-font`.

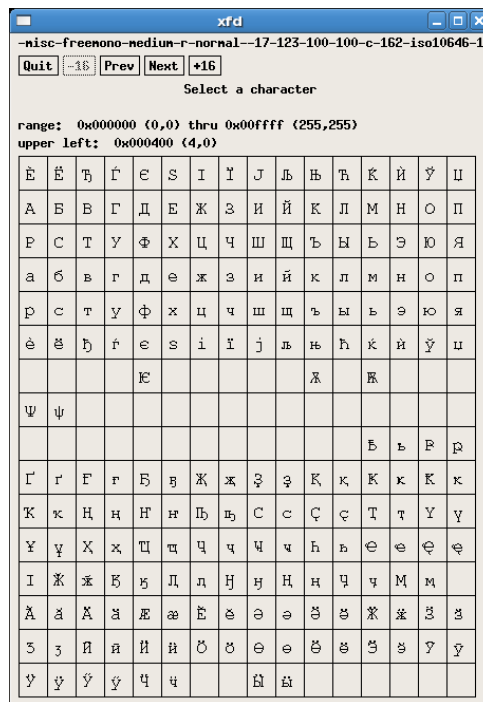
17.11 Получение списка шрифтов — showrgb

Описание:

Служит для отображения таблицы соответствия символьных названий цветов значениям интенсивности по схеме RGB.

Примеры:

```
$ showrgb
255 250 250          snow
248 248 255          ghost white
248 248 255          GhostWhite
245 245 245          white smoke
245 245 245          WhiteSmoke
220 220 220          gainsboro
255 250 240          floral white
255 250 240          FloralWhite
253 245 230          old lace
253 245 230          OldLace
```

Рис. 17.3. Программа `xfd`

```

250 240 230      linen
250 235 215      antique white
250 235 215      AntiqueWhite
255 239 213      papaya whip
255 239 213      PapayaWhip
255 235 205      blanched almond
255 235 205      BlanchedAlmond
255 228 196      bisque
255 218 185      peach puff
255 218 185      PeachPuff
255 222 173      navajo white
255 222 173      NavajoWhite
255 228 181      moccasin

```

17.12 Список текущих ресурсов клиента — `arpres`

Синтаксис:

`arpres` [[класс [объект]] [-1]]

Описание:

Получение списка ресурсов указанных классов, объектов.

Опции:

-1	Отображать ресурсы только данного уровня
----	--

Примеры:

```
$ appres
*Button.background:      #ede9e3
*Button.activeForeground: #000000
*Button.highlightColor:  #000000
*Button.highlightBackground: #ede9e3
*Button.foreground:      #000000
*Button.activeBackground: #ffffff
*Label.background:       #ede9e3
*Label.foreground:       #000000
*Label.highlightBackground: #ede9e3
*Label.highlightColor:   #000000
*XmMessageBox.background: #ede9e3
*XmMessageBox.foreground: #000000
*XmToggleButtonGadget.background: #ede9e3
*XmToggleButtonGadget.foreground: #000000
*SimpleMenu*background: #ede9e3
*SimpleMenu*foreground: #000000
*Labelframe.background: #ede9e3
*Labelframe.foreground: #000000
*Labelframe.highlightColor: #000000
*XmPanedWindow.background: #ede9e3
*XmPanedWindow.foreground: #000000
*SmeBSB*shadowWidth:    3
```

17.13 Манипуляции с ресурсами — xrdp

Синтаксис:

xrdp [опции] [файл]

Описание:

Просмотр и установка ресурсов. Для препроцессинга файлов используется стандартный препроцессор языка C. В графической среде GNOME используется `gconftool-2`.

Опции:

-all	«Для всех» ресурсов в опции <code>-query</code> , экранов в <code>-load</code> и <code>-merge</code>
-edit файл	Изменения вносятся в <i>файл</i>
-load	Загрузить ресурсы, заменив ими установленные ранее. Используется по умолчанию
-query	Вывести листинг установленных ресурсов
-remove	Удалить указанные ресурсы

17.14 Редактор ресурсов — editres

Утилита, позволяющая инерактивно просматривать и устанавливать ресурсы указанного приложения (см. [Рис. 17.3](#)). В графической среде GNOME используется `gconf-editor`

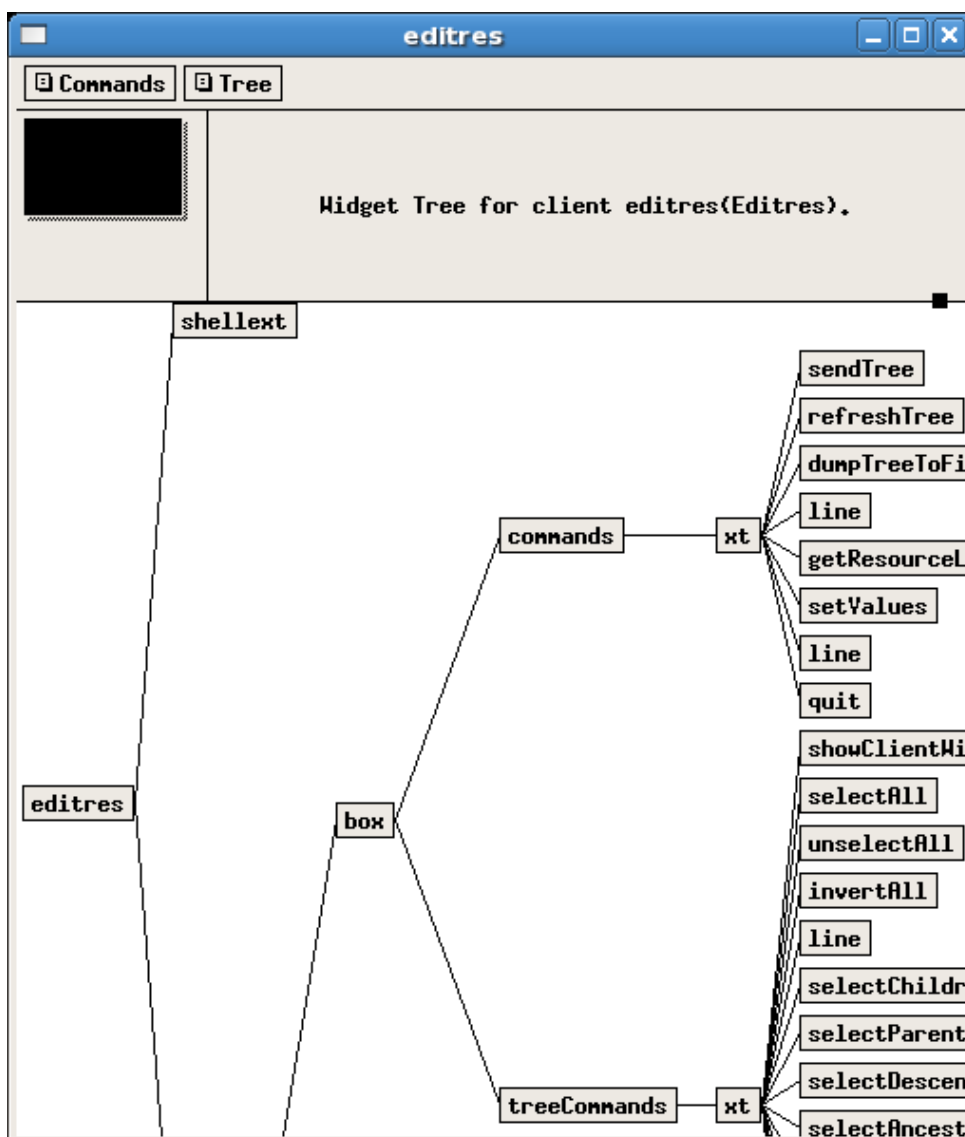


Рис. 17.4. Программа editres

17.15 Установка свойств дисплея и устройств ввода — xset

Синтаксис:

xset [опции]

Описание:

Служит для установки свойств дисплея и устройств ввода, аналогичных тем, которые устанавливаются при старте Xserver'a . В современных графических интерфейсах типа GNOME и KDE используются собственные механизмы управления свойствами хранителя экрана (screen saver'a).

Опции:

[+ -]fp[+ - =] <i>каталог...</i>	Подключение и отключение каталогов шрифтов, в том числе и фонт-серверов
fp default	Установка пути поиска шрифтов по умолчанию
fp rehash	Перечитать каталоги
p ячейка цвет	Установка <i>цвета</i> для указанной <i>ячейки</i> карты цветов
s параметр	Установка <i>параметров</i> screen saver'a (blank/noblank, activate/reset, on/off)
q	Вывести текущие установки

Примеры:

```
xset +fp /usr/share/fonts/ms/,tcp:fontserver.mycompany.com:710
```

17.16 Установка свойств корневого окна — xsetroot

Синтаксис:

```
xsetroot [-def] [-gray|grey] [-mod x y] [-solid цвет]
```

Описание:

Установка свойств корневого окна, включая цвет и режимы заполнения. В современных графических интерфейсах типа GNOME и KDE используются собственные механизмы управления свойствами корневого окна (фона рабочего стола).

Опции:

-def	Установка режимов по умолчанию
-gray grey	Задать серый фон
-mod x y	Установить заполнение сеткой. Величины <i>x</i> и <i>y</i> могут лежать в пределах от 1 до 16
-solid цвет	Заполнить фон <i>цвет</i> 'ом равномерно

Примеры:

```
xsetroot -mod 12 12 -bg darkviolet -fg darkblue
```

17.17 Отслеживание событий — хев

Создает специальное окно, в котором отслеживаются все события от внешних устройств.

17.18 Установка раскладок клавиатуры и мыши — xmodmap

Синтаксис:

```
xmodmap [-e выражение] [-n] [-rm] [-pk] [-rke] [-pp]
```

Описание:

Утилита для просмотра и изменения раскладок клавиатуры и кнопок мыши. В современных графических системах обычно замещается системами, работающими с механизмами X Keyboard Extension. В

современных графических интерфейсах типа GNOME и KDE используются собственные механизмы управления свойствами клавиатуры и мыши.

Опции:

<code>>-е</code> <i>выражение</i>	Выполнить указанное <i>выражение</i>
<code>-n</code>	Отображать ход исполнения, реальных действий не выполнять
<code>-pm</code>	Вывод раскладки клавиш модификаторов
<code>-pk</code>	Вывод раскладки клавиатуры
<code>-pke</code>	Вывести раскладку в форме, пригодной для передачи команде xmodmap для исполнения
<code>-pp</code>	Вывести раскладку кнопок мыши

Примеры:

```
$ xmodmap -pke
keycode 8 =
keycode 9 = Escape
keycode 10 = 1 exclam
keycode 11 = 2 at 2 quotedbl
keycode 12 = 3 numbersign 3 numerosign
keycode 13 = 4 dollar 4 semicolon
keycode 14 = 5 percent
keycode 15 = 6 asciicircum 6 colon
keycode 16 = 7 ampersand 7 question
keycode 17 = 8 asterisk
keycode 18 = 9 parenleft
keycode 19 = 0 parenright
keycode 20 = minus underscore
keycode 21 = equal plus
keycode 22 = BackSpace Terminate_Server
keycode 23 = Tab ISO_Left_Tab
keycode 24 = q Q Cyrillic_shorti Cyrillic_SHORTI
keycode 25 = w W Cyrillic_tse Cyrillic_TSE
keycode 26 = e E Cyrillic_u Cyrillic_U
keycode 27 = r R Cyrillic_ka Cyrillic_KA
...
```

17.19 Установка параметров клавиатуры — `setxkbmap`

Синтаксис:

setxkbmap [опции] [раскладка [вариант [опции_раскладки...]]]

Описание:

Установка параметров клавиатуры через механизм X Keyboard Extension.

Опции:

-print	Вывести текущие установки в формате, пригодном для использования xkbcomp .
-keymap <i>название</i>	Указать раскладку клавиатуры
-variant <i>название</i>	Указать вариант раскладки
-model <i>название</i>	Указать модель клавиатуры
-option <i>название</i>	Задать опции раскладки

Примеры:

Установка раскладки клавиатуры на сервере с использованием карт раскладки, установленных на клиентской машине:

```
setxkbmap us,ru -print | xkbcomp - $DISPLAY
```

17.20 Обновление экрана — xrefresh

Синтаксис:

```
xrefresh [-black] [-none] [-root] [-solid цвет] [-white]
```

Описание:

Утилита позволяет перерисовать весь экран или указанную в опции **-geometry** область.

Опции:

-black	Использовать при перерисовки черный фон.
-none	Используется по умолчанию. Просто перерисовываются все окна.
-root	Использовать при перерисовке цвет корневого окна.
-solid <i>цвет</i>	Использовать при перерисовке указанный <i>цвет</i> .
-white	Использовать при перерисовки белый фон.

Предметный указатель

- .bashrc, 19
- .cshrc, 19
- .login, 19
- .profile, 19
- .tcshrc, 19
- .zprofile, 19
- .zshrc, 19

- alias, 17
- appres, 99
- apropos, 24
- at, 56
- atq, 57
- atrm, 57
- awk, 64

- basename, 73
- batch, 56
- bg, 55
- bunzip2, 46
- bzip2, 46

- cal, 27
- cansel, 49
- case, 71
- cat, 38
- cd, 33
- chmod, 37
- chown, 38
- cmp, 66
- cp, 33
- cpio, 46
- crontab, 57

- date, 26
- diff, 66
- dirname, 73
- DISPLAY, 88

- e-mail, 81
- echo, 73
- ed, 59
- egrep, 44
- emacs, 61
- espr, 73
- eval, 73
- exec, 73
- exit, 10

- fg, 55
- file, 51
- find, 50

- fold, 66
- for, 72
- ftp, 80
- function, 73

- GConf, 90
- gconf-editor, 100
- gconftool-2, 100
- gnome-terminal, 92
- grep, 44
- gunzip, 46
- gzip, 46

- head, 41
- history, 18

- id, 27
- if, 71
- info, 24

- Jobs, 18
- jobs, 54
- joe, 62
- join, 68

- kill, 55
- konsole, 92

- less, 40
- lftp, 83
- ln, 34
- login, 9
- logname, 27
- logout, 10
- lp, 48
- lpq, 48
- lpr, 48
- lprm, 48
- lpstat, 49
- ls, 32
- lynx, 82

- mail, 81
- man, 22
- md5sum, 47
- mkdir, 36
- more, 39
- mv, 35

- od, 43

- passwd, 9

- paste, 69
- pg, 40
- ps, 53
- pwd, 33

- read, 74
- readonly, 74
- rm, 35
- rmdir, 36
- rsync, 84

- scp, 77
- sed, 64
- setxkbmap, 103
- shell, 11
- shift, 74
- showrgb, 98
- sleep, 56
- sort, 67
- split, 42
- ssh, 77
- sshd, 77
- stty, 30
- sum, 47

- tail, 41
- tar, 44
- tee, 42
- telnet, 79
- test, 52
- tr, 68
- tty, 30
- type, 51

- uname, 26
- uniq, 67
- until, 72

- vi, 60

- wc, 42
- wget, 83
- whatis, 23
- which, 51
- while, 72
- who, 29
- Window Manager, 92
- write, 29

- X-Window, 88
- xdpyinfo, 95
- xev, 102
- xfd, 98
- xfontsel, 98
- xkbcomp, 104

- xkill, 93
- xlsclients, 94
- xlsfonts, 97
- xmodmap, 102
- xrdb, 100
- xrefresh, 104
- Xserver, 91
- xset, 101
- xsetroot, 102
- Xt, 90
- xterm, 92
- xwininfo, 93

- регулярные выражения, 44