Глава 8. Форматы

Учебник по Perl

Содержание [скрыть]

Как мы помним, дословный перевод аббревиатуры языка Perl включает в себя слова "язык отчетов", т. е. язык Perl предназначен не только для извлечения и обработки информации из текстовых файлов, но и для генерирования отчетов на основе этой информации. Пока что мы для вывода информации использовали функцию print (), которая не очень-то удобна для создания отчетов – определенным образом отформатированной выходной информации. (Можно было бы воспользоваться функцией форматированного вывода printf о, но мы решили не нагружать нашего читателя изучением языковых средств, которыми он редко будет пользоваться, тем более что всегда можно обратиться к документации Perl.)

Для создания простых отчетов в Perl предусмотрены *форматы,* которые позволяют в тексте программы практически визуализировать внешний вид выводимого отчета, так как определение формата в Perl очень близко к тому, что отображается при выводе. Форматы позволяют задавать верхний колонтитул каждой страницы отчета, куда можно поместить название документа, номер страницы и другую полезную информацию. Perl может отслеживать количество строк на странице отчета и автоматически переходить на новую страницу по заполнению всех строк предыдущей.

Использование форматов для создания отчетов очень просто. Первое, что необходимо сделать, – определить формат и переменные, которые в нем используются. Далее следует инициализировать в программе эти переменные и осуществить форматированный вывод в файл, определенный своим дескриптором, с помощью функции write ().

Эти и другие, связанные с форматами, вопросы и являются предметом изучения в этой главе. Начнем мы с основного вопроса – объявление форматов в программе Perl.

Объявление формата

Формат – это одна из двух языковых единиц (вторая – подпрограмма sub), которая требует обязательного объявления в программе Perl. Он используется в качестве "руководства" функцией write о, которая выводит на экран монитора, принтер или в файл информацию из программы в соответствии с записанными в формате "инструкциями" форматирования строк вывода. При объявлении формата определяется, как должна быть отформатирована каждая его строка при отображении на устройстве вывода.

Формат объявляется в программе с помощью ключевого слова format, после которого следуют "инструкции" по форматированию определенных в нем строк. Завершается объявление формата строкой, первым символом которой является точка ".". Общий синтаксис конструкции объявления формата следующий: /

```
format ИМЯ_ФОРМАТА •= ФОРМАТЫ_СТРОК
```

Параметр ИМЯ_ФОРМАТА представляет собой правильный идентификатор Perl. Он должен в точности соответствовать имени дескриптора файла, который используется в качестве единственного параметра в функции вывода write o. Например, если форматированный отчет выводится в файл, определенный в программе дескриптором FILE, то и имя формата должно быть также FILE. Функцию write o можно вызывать без параметра. В этом случае вывод осуществляется на стандартное устройство вывода (STDOUT), и имя формата в этом случае должно быть равным STDOUT. Если функцией select o установлен дескриптор файла вывода по умолчанию, то вывод функцией write o без параметра будет осуществляться в этот файл, причем имя формата вывода должно быть изменено на имя дескриптора файла.

Замечание

Так как операция format не является вычисляемой операцией Perl, то объявление формата может осуществляться в любом месте программы. Обычно все объявления форматов задают либо в начале, либо в конце программы.

Замечание

Имя формата, как отмечалось, может быть любым правильным идентификатором Perl, однако обычно его определяют прописными буквами, что способствует лучшей читаемости программы.

В теле формата (до завершающей строки с точкой) определяются форматы для *каждой* строки вывода. Формат строки состоит из двух строк: первая, называемая *строкой шаблонов,* определяет, как отображается информация, вторая, называемая *строкой переменных,* задает переменные, содержащие выводимую информацию. Вместе эти две строки определяют формат и содержимое одной строки вывода функцией write o.

Строка шаблонов печатается точно так, как она выглядит в тексте программы (включая пробельные символы), за исключением некоторых полей, в которые подставляются значения переменных из строки переменных. Эти *поля* (иногда их называют *шаблоны,* что и дало название соответствующей строке формата) начинаются с символа "в" или "Л", за которым следуют *символы форматирования* (табл. 8.1), определяющие ширину поля вывода значения переменной в символах и выравнивание выводимого значения внутри поля. Количество символов форматирования определяет ширину поля вывода, причем для одного поля все символы должны быть одинакового типа.

Переменные, определяющие значения для полей строки шаблонов, задаются через запятую в строке переменных. Порядок их задания соответствует порядку задания полей вывода в строке шаблонов: значение первой переменной выводится в первое поле, второй – во второе и т. д. Все переменные в строке переменных вычисляются в списковом контексте. Это позволяет задавать выводимые значения в элементах массива скаляров.

Замечание

Если строка шаблонов не содержит полей, то для нее не надо задавать строку переменных. Она отображается в точности так, как она задана в формате.

Таблица 8.1. *Символы форматирования*

Символ	Описание
>	Определяет символьное поле, в котором выводимое значение выровнено по правому краю
<	Определяет символьное поле, в котором выводимое значение выровнено по левому краю
#	Определяет числовое поле (выводимое значение должно быть числом)
.	Определяет положение десятичной точки в числовом поле (###.##)
I	Определяет символьное поле, в котором выводимое значение выровнено центру

Небольшой пример прольет свет на все вышесказанное – лучше один раз увидеть, чем сто раз услышать. Предположим, что у нас имеется файл (назовем его books), содержащий информацию о книгах, продаваемых неким книжным магазином. Каждая строка этого файла содержит информацию об одной книге: автор(ы), название, издательство, год выпуска и стоимость. Все поля записи разделены символом двоеточие ":". Одна из строк этого файла может выглядеть так:

```
В.Долженков Ю.Колесников:Excel 2000:BHV:1999:90
```

Нам необходимо распечатать отчет о всех продаваемых книгах. Воспользуемся форматами Perl. Программа примера 8.1 реализует поставленную задачу.

```
#! perl -w . "
open BOOKS, "<books"; # Открытие файла на чтение
```

```
while (<BOOKS>) {
{$author, $title, $pub, $year, $price) = split(':'); t Разбиение строки по символу ':'
write; # Форматный вывод строки
} '.....
format STDOUT =
@<«««««««««««««« | e»»»»»»» i @i 111111 | @#### | @###.##p.
$author, $title, $pub, $year, $price
```

Результат отображения отчета на экране монитора выглядит следующим образом:

```
В.Долженков Ю.Колесников |, Excel 2000 | BHV | 1999 | 90.00p.
А.Матросов А.Сергеев М.Чау | HTML 4.0 I BHV | 1999 | 70.00p.
Т.Кристиансен Н.Торкингтон | Perl | Питер | 2000 | 100.00p.
```

Обратите внимание, что все символы строки шаблонов печатаются именно в тех позициях, в которых они заданы, а в поля этой же строки, определенные символом "@", подставлены значения соответствующих переменных. Эти значения отображаются в соответствии с заданными символами форматирования: для переменной $authqr вывод выровнен по левому краю (<), для $ title по правому краю (>), для $pub пр центру (|) соответствующего поля. Значения переменных $yeaг и $price форматируются в соответствии с числовым форматом. Если бы эти переменные не содержали числовые значения, то интерпретатор perl вывел бы предупреждающее сообщение.

> **Замечание**
>
> Символ начала поля в строке шаблонов ("@" или "л") учитывается при подсчете ширины поля вывода.

Еще один нюанс, связанный с форматированием значений переменных. Если она содержит строковые данные, количество символов которых превосходит заданную ширину поля вывода, то лигцние символы отсекаются справа. Именно это и произошло при выводе второй записи файла books: фамилия третьего автора напечатана не полностью.

Как поступать в таких случаях? Можно увеличить ширину поля, если позволяют параметры выводного устройства, а можно воспользоваться еще одним символом форматирования, который как раз и предназначен для решения подобных проблем. Прежде всего следует для задания начала поля использовать символ "л". Его отличие от символа "e" заключается в том, что до начала вывода строковых данных в поле Perl в промежуточном буфере аккумулирует слова из выводимых данных, формируя строку, длина которой не превышает ширину поля. После этого сформированные данные выводятся в строке, а значение переменной вывода модифицируется: она будет содержать оставшиеся слова. При последующем использовании этой переменной в другой строке переменных того же формата будут выводиться сохраненные в ней отсеченные данные. Это позволяет выводить длинные данные в

нескольких строках в одном вертикальном блоке, если задавать для вывода оставшихся данных точно такое же поле, что и при выводе первой порции.

Заменим формат STDOUT программы примера 8.1 на следующий:

```
format STDOUT =
-<«««««««««««««««« | @»»»»»»» | @||||||| | @##it f @###.##p.
$author, $title, $pub, $year, $price
л«««««««««««««<«« | | I | ~
$author
```

Теперь вывод нашей программы будет выглядеть так:

```
В.Долженков Ю.Колесников I Excel 2000 | BHV | 1999 | 90.00р.
А.Матросов А.Сергеев | ' HTML 4.О Г BHV | 1999 | 70.00р.
М.Чаунин I I II
Т.Кристиансен Н.Торкингтон I Perl | Питер | 20.00 | 100.00р.
```

Символ тильда "~" в конце строки Шаблона подавляет вывод пустых строк. Если не поставить его, то между первой и второй книгой в нашем отчете появится дополнительная строка, как если бы была выведена вторая строка шаблона с пустым значением переменной $author. Символ подавления вывода пустых строк можно задавать в любом месте строки шаблона, помня, что при выводе он отображается, как пробел.

В нашем примере мы знали, что данные в переменной $author Не займут более двух строк при выводе. Поэтому в формате мы использовали эту информацию, добавив еще одну строку шаблона с переменной $author. А что делать, если не известно количество строк продолжения в которых будут выводиться данные? Можно воспользоваться двумя идущими подряд символами тильда вместо одного. В *этщ* случае алгоритм буферизации данных по словам будет продолжаться до завершения вывода всех данных переменной. Если наш формат изменить на следующий

```
format STDOUT = . _....,..,... .;,
л<«««««««««««««««« | @»»»»»» ,| @| | | | || | | в#### I @###.-##p.
$author, $title, $pub, $year, $price
$author
```

а во вторую книгу добавить еще парочку авторов, то вывод записи об этой книге нашей программой будет иметь следующий вид:

```
А.Матросов А.Сергеев | HTML 4.0 I BHV | 1999 | 70.00р.
М.Чаунин В.Долженков I III
Ю.Колесников I III
```

Таким способом можно организовать вывод длинных данных в вертикальные текстовые блоки с переносом по словам.

Немного отвлечемся от создания отчета по книгам, чтобы познакомить читателя с еще одним символом форматирования – символом "*", который позволяет выводить длинные строковые данные в нескольких строках, длина которых равна максимальной ширине вывода устройства отображения (экрана монитора или принтера). Например, если переменная $ record содержит строковые данные длиной более 80 символов и вывод осуществляется на экран монитора, то следующий фрагмент кода Perl

```
write;
format STDOUT =
$*
$record
```

отобразит на экране ее содержимое следующим образом:

```
В.Долженков Ю.Колесников:Excel 2000:BHV:1999:90:
Книга является справочным пособием по MS Excel 2000.
В ней рассматриваются следующие основные темы - настройка интерфейса и его основные элементы.
```

Замечание

Используемый в нашей книге шрифт для представления вывода сценария Perl не позволяет нам отобразить истинный вывод в соответствии с заданным форматом, так как внимательный читатель обнаружит, что длина строки вывода в нашем представлении составляет 73 символа, а не 80, как при выводе на экран монитора.

Вернемся к разработке формата для вывода нашего отчета. Пока что отчет был достаточно маленьким и помещался ha одной странице. Реальные отчеты, конечно, на одной странице не поместятся. Наша программа напечатает и несколько страниц отчета. Дело в том, что создание отчетов в Perl предполагает их вывод на принтер, а поэтому после вывода определенного количества строк оператором write () Perl автоматически выведет символ перехода на новую страницу и печать продолжится на следующей странице. По умолчанию количество строк на странице установлено равным 60. Эта величина хранится в специальной переменной $=, значение которой может быть изменено в любое время.

Итак, мы теперь знаем, что переход на новую страницу происходит автоматически, но нам хотелось бы, чтобы на каждой странице печатался *верхний колонтитул,* в котором отображалось бы наименование отчета и печатались номера страниц. И это возможно в Perl. Следует только задать формат со специальным именем, добавив к имени формата, по которому мы выводим информацию (в

нашей программе STDOUT), суффикс _TOP. Этот формат будет выводиться каждый раз, как начинается печать новой страницы.

Добавим в программу примера 8.1 следующее объявление формата

```
format STDOUTJTOP =
Книги на складе @>»>»
"стр. ".$% Автор Название Издатель Год Цена
```

и явно зададим количество строк на странице, добавив перед циклом while оператор

```
$= = 6;
```

Теперь наша программа напечатает две страницы отчета, причем на каждой из них будет напечатан колонтитул:

```
Книги на складе стр. 1 Автор - Название Издатель Год Цена
В.Долженков Ю.Колесников | Excel 2000 I BHV | 1999 | 90.00р.
А.Матросов А.Сергеев | HTML 4.0 I BHV | 1999 | 70.00р. М.Чаунин ,| | II
——разрыв страницы——
Книги на складе стр. 2 Автор Название Издатель Год Цена
Т.Кристиансен Н.Торкингтон | Perl I Питер | 2000 | 100.00р.
```

Вернемся к объявлению формата для колонтитула. Во-первых, при его задании мы использовали выражение "стр. ".$% в строке переменных. Действительно, хотя формат и не вычисляется, но во время выполнения программы вычисляются значения переменных и все выражения строки переменных формата. Во-вторых, мы использовали специальную переменную $%, которая хранит текущий номер выводимой страницы. Это позволило нам в колонтитуле напечатать номера страниц.

Использование нескольких форматов

Как мы уже знаем, форматы Perl позволяют без каких-либо усилий создавать верхние колонтитулы – следует только объявить формат с суффиксом _TOP. Для создания полноценного документа не мешало бы еще иметь возможность создавать нижние колонтитулы страницы и печатать, например, в конце заказа общую стоимость. К сожалению, такой возможности Perl не предоставляет, но он позволяет переключать вывод с одного формата на другой и в специальной переменной хранит строку, которую печатает перед переходом на новую страницу. А это и позволит нам создать и напечатать и нижний колонтитул, и общую стоимость заказа.

Но прежде мы еще немного поговорим о специальных переменных Perl, которые используются для управления форматом. В переменной $~ хранится имя формата, который используется при выводе функцией write о без параметра:

```
write; # Эквивалентно оператору write STDOUT;
```

По умолчанию в ней хранится имя формата STDOUT, но и вывод функцией write о без параметра происходит на стандартное устройство вывода STDOUT. (Мы помним, что имя формата должно совпадать с именем дескриптора файла в вызове функции write о, а именно такая ситуация по умолчанию и реализуется.) Если мы изменим значение переменной $~ на имя другого формата, то вывод в стандартный файл функцией write о без параметра будет осуществляться в соответствии с указанным форматом, который, конечно, должен быть объявлен в программе. Например, следующий оператор write выводит на стандартное устройство вывода в соответствии с форматом NEW:

```
$- = NEW; write/format NEW =
```

Таким образом, меняя значение переменной $~, можно переключать вывод с одного формата на другой. Этим другим форматом как раз и может быть формат общей стоимости заказа.

Пусть в файле books содержится информация о заказанных книжным магазином книг. В конце отчета по заказу нам теперь необходимо напечатать общую стоимость книг. Решение показано в программе примера 8.2.

```
#! perl -w
open BOOKS, "<books"; # Открытие файла на чтение
$number = 1;
$total = 0;
while (<BOOKS>) {
($author, $title, $pub, $year, $price) = split(':'); # Разбиение строки
t по символу ':'
write; # Форматный вывод строки $total += $price; t Подсчет общей суммы
}
$~ = TOTAL; # Переключение формата
write; # Вывод по формату итоговой строки
format STDOUTJTOP =
Заказ № @#
$ number Автор Название Издатель Год Цена
format STDOUT = '
Л<«««««««««««««« | @»»»»»» | @M!11M I @t### I @###.##p.
$author, $title, $pub, $year, $price
A«««««««««««<«««« | | | | ~~
$author
format TOTAL =
Итого: @###.##p. $total
```

В этой программе после форматной печати содержимого файла books осуществляется переключение на другой формат, по которому выводится строка с общей суммой заказа, подсчитанной в переменной

$totai. Полученный с помощью этой программы заказ показан ниже

```
Заказ № I Автор Название Издатель Год Цена
В.Долженков Ю.Колесников I Excel 2000 I BHV | 1999 I 90.00р.
А.Матросов А.Сергеев I HTML 4.0 I BHV | 1999 | 70.00р.
М.Чаунин I I II
Т.Кристиансен Н.Торкингтон I Perl I Питер | 2000 | 100.00р.
Итого: 260.00р.
```

В завершение разговора о создании отчетов в Perl мы модифицируем программу примера 8.1, приспособив ее для печати отчета на основании информации о книгах из файла books, в котором в записи о книгах добавлено еще одно поле, содержащее краткую аннотацию книги:

```
В.Долженков Ю.Колесников:Excel 2000:BHV:1999:90:Аннотация книги
```

Отчет, формируемый этой программой (пример 8.3), также печатает нижний колонтитул на каждой странице. Для этого мы воспользуемся специальной переменной $^Lь, содержимое которой Perl печатает перед переходом на новую страницу во время форматного вывода. При этом следует уменьшить на количество строк, заданных в этой переменной, количество строк на странице, хранящееся в специальной переменной $=, иначе строки из переменной $^Lь попадут не в конец текущей страницы, а будут напечатаны на следующей странице, не создав никакого нижнего колонтитула.

```
#! perl -w
open FILE, "<books" or die $!;
open REPORT, "xreport" or die $!;
select REPORT; $~ ·= STDOUT; $= = 24;
Sltime = localtime;
$-L = ·("=" x 73) . "\n". "Книготорговая база \"БЕСЫ\"". (" " x 24)."$ltime\n\f";
$count = 0;
while(<FILE>) { .
($author, $title, $pub, $year, $price, $annot) = split(':');
$count++;
write ;
}
close(REPORT); format STDOUT_TOP =
Книги на складе @»»»»
; "стр. ".$%
Автор Название Издатель Год Цена Аннотация________
format STDOUT =
@| ^«««««« /ч^«««««« @<«««« @### @##.##р. ^«««««««««««
```

```
$count.".", $author, $title, $pub, $year, $price, $annot
Л«««««««« Л«««««««« А«««««««««««
$author, $title, $annot
```

Вывод отчета осуществляется в файл с именем report. Обратите внимание на задание переменной $/4L. В ней используется переменная $itime, в которой хранится текущая дата, полученная обращением к функции localtime. Одна страница отчета будет выглядеть следующим образом:

```
Книги на складе . стр. t Автор_______Название___Издатель Год Цена Аннотация_______
1. В.Должен ков Excel 2000 BHV 1999 90.00р. Книга является
Ю.Колесников справочным пособием по MS Excel 2000. В ней рассматриваются следующие основные
темы - настройка интерфейса и его основные элементы.
2. А.Матросов HTML 4.0 BHV 1999 70.00р. Представлен весь А.Сергеев спектр технологий М.Чаунин
создания Web-документов (начиная от простейших - статических - и до документов на основе
динамического HTML), включая форматирование текста, создание списков. Книготорговая база
"БЕСЫ" Sat Mar 18 19:01:37 2000
```

Замечание

Представленные *в* этой главе отчеты являются снятыми копиями экрана монитора, вывод на который осуществляется с использованием моноширинного шрифта. Если вывод осуществляется на принтер, то чтобы отчеты выглядели так, как они должны выглядеть, следует также использовать моноширинный шрифт, например Courier. Если используется пропорциональный шрифт, принятый на многих принтерах по умолчанию, то сформированные сценарием Perl отчеты "поползут", так как в в этих шрифтах каждый символ имеет собственную ширину, тогда как в моноширинных все символы имеют одинаковую ширину.

Отчеты в Perl создаются с помощью форматов, определяющих внешний вид строк вывода и содержащуюся в них информацию. Печатаются отчеты функцией write ( ), которая осуществляет вывод как на стандартное устройство вывода, так и в файл, открытый в программе. В процессе печати отчета можно переключаться между существующими форматами.

Вопросы для самоконтроля

1. Опишите процедуру создания отчетов в Perl?
2. Какой синтаксис имеет оператор format?
3. Какая функция используется для активизации форматного вывода?
4. Как создается верхний колонтитул для страниц отчета?
5. Как создается нижний колонтитул для страниц отчета?
6. Каким образом осуществляется переключение между форматами?

7. Перечислите специальные переменные Perl, которые используются для управления форматным выводом.