Глава 1. Введение в мир Perl

Учебник по Perl

Содержание [скрыть]

Что такое Perl? Это сокращенное название языка программирования Practical Extraction and Report Language (Практический язык извлечений и отчетов). Что подразумевается под "извлечениями" и "отчетами"? Почему *практический* язык? Для чего он предназначен? Какие задачи можно решать с его помощью? Эти и многие другие вопросы возникают, естественно, у любого человека, хоть немного знакомого с информатикой, когда он впервые сталкивается с новым для него языком программирования. Эта глава и задумывалась как ответ на поставленные выше вопросы, так как зная, что может, для чего предназначен язык программирования (а время универсальных языков, кажется, миновало), программист, в конечном счете, решает, а стоит ли тратить время на его изучение. Хотя здесь также встают вопросы о легкости и быстроте освоения нового языка, доступности компиляторов, существовании службы его поддержки, стоимости и т. д. Об этом также пойдет речь в этой главе, которая познакомит читателя с огромным миром Perl-программирования, и станет той отправной точкой, с которой он, мы надеемся, стремительно и без оглядки войдет в него и останется в нем навсегда.

Язык Perl родился в недрах операционной системы Unix как реакция одного талантливого программиста на ограниченную возможность стандартных средств системного администрирования в этой операционной среде. Авторы прекрасно осознают, что большинство читателей знакомы с Unix, возможно, только по названиям книг, лежащих на полках магазинов, так как традиция изучения информационных технологий в нашей стране связана больше с операционными системами семейства Microsoft Windows [Под семейством операционных систем Microsoft Windows понимаются операционные системы Windows 95/98/NT.], чем с системой UNIX, которая является базой изучения информатики в западных университетах. Поэтому для воспитанных в традициях Windows читателей мы сделаем небольшое отступление и кратко охарактеризуем процедуру администрирования UNIX, которая радикально отличается от аналогичной работы в операционной системе Windows.

Под *администрированием* понимается настройка операционной системы через установку значений ее параметров таким образом, чтобы она отвечала потребностям отдельного пользователя или группы пользователей. В системах семейства Windows подобная работа выполняется с помощью Реестра, представляющего собой базу данных двоичных данных, а для изменения параметров используется специальная программа regedit. В системе UNIX настройка осуществляется через специальные конфигурационные файлы, являющиеся обычными текстовыми файлами, и все изменения осуществляются выполнением команд, написанных на специальном языке оболочки (shell) и выполняемых, как правило, из командной строки. (Несколько лет назад на персональных компьютерах была широко распространена операционная система MS-DOS фирмы Microsoft, в которой для ввода команд также использовалась командная строка, поэтому читателю, работавшему в этой операционной системе, командная строка знакома.) В системе UNIX пользователь может создавать собственные команды на основе команд интерпретатора shell, сохранять их в обычных текстовых файлах и впоследствии выполнять также, как обычные стандартные команды операционной системы через командную строку. Следует отметить, что оболочка shell операционной системы UNIX является интерпретатором, в связи с чем команды пользователя имеют еще одно название – их называют *сценариями* или *скриптами* (script). Администратору операционной системы UNIX приходится писать большое количество скриптов, которые обрабатывают другие скрипты – текстовые файлы. Для этих целей обычно кроме командного языка оболочки shell используются специальные программы обработки текстовых файлов:

- awk – программа сопоставления с образцами и генератор отчетов;
- sed – пакетный редактор текстовых файлов.

Обе эти программы являются фильтрами, которые последовательно считывают строки входных файлов и выполняют применимые к строке действия, определенные с помощью команд этих программ. Основными действиями являются выделение цепочек символов по заданным шаблонам, замена их по определенным правилам и генерирование новых файлов.

Теперь можно перейти и к объекту нашего изучения – языку Perl, тем более что, как нам кажется, читателю уже должно быть понятно, почему он называется языком извлечений и отчетов. И начнем мы с истории его создания и разработки, которая, по существу, позволяет полнее понять его содержание.

История языка Perl

Perl был разработан Ларри Уоллом (Larry Wall) в 1986 году, когда он являлся системным администратором одного проекта UNIX, связанного с созданием многоуровневой безопасной сети, объединявшей несколько компьютеров, разнесенных на большие расстояния. Работа была выполнена, но потребовалось создание отчетов на основе большого числа файлов с многочисленными перекрестными ссылками между ними.

Первоначально Ларри предполагал использовать для этих целей фильтр awk, но оказалось, что последний не мог управлять открытием и закрытием большого числа файлов на основе содержащейся в

них же самих информации о расположении файлов. Его первой мыслью было написать специальную системную утилиту, решающую поставленную задачу, но вспомнив, что до этого ему уже пришлось написать несколько утилит для решения задач, не "берущихся" стандартными средствами UNIX, он принял кардинальное решение – разработать язык программирования, который сочетал бы в себе возможности обработки текстовых файлов (sed), генерации отчетов (awk), решения системных задач (shell) и низкоуровневое программирование, доступное на языке C. Результатом этого решения и явился язык Perl, интерпретатор для которого был написан на C.

По утверждению самого Ларри Уолла при создании языка Perl им двигала лень – не в прямом смысле, а в смысле того, что для решения стоявшей перед ним задачи следовало бы написать большое количество программ на разных языках, входящих в состав инструментальных средств UNIX, а это достаточно утомительное занятие.

Новый язык программирования сочетал в себе возможности системного администрирования и обработки файлов – две основные задачи, решаемые обычно при программировании в системе UNIX. Причем следует отметить, что язык Perl появился из практических соображений, а не из-за желания создать еще одно "красивое" средство для работы в UNIX, поэтому-то он и получил широкое распространение среди системных администраторов, когда Ларри Уолл предоставил его широкому кругу пользователей. С появлением языка Perl появилась возможность решать задачи с помощью одного инструмента, и не тратить время на изучение нескольких языков среды программирования UNIX.

Первая версия языка не содержала многих возможностей, которые можно найти в последней версии Perl, с которой читатель познакомится в нашей книге и которая идентифицируется как версия 5.005_03 и считается устойчивой. Первоначально язык включал:

- простой поиск по строковым образцам (шаблонам) в файлах;
- дескрипторы файлов;
- скалярные переменные;
- форматы.

Вся документация умещалась на 15 страницах, но Perl решал задачи быстрее, чем sed или awk, и быстро стал использоваться не только для решения задач системного администрирования.

В дальнейшем сам Ларри Уолл позаимствовал у Генри Спенсера (Henry Spencer) пакет для работы с регулярными выражениями и модифицировал его для языка Perl. Другие функциональные возможности были разработаны не только Ларри Уоллом, но и его друзьями и коллегами, и включены в состав языка. Опубликование в Internet привело к появлению сообщества единомышленников, которые не только эксплуатировали, но и развивали язык. Он и по настоящее время продолжает интенсивно развиваться за счет разработки пакетов, реализующих новые применения языка к развивающимся

информационным технологиям. В табл. 1.1 представлена динамика появления новых версий языка Perl, начиная с самой первой:

Таблица 1.1. *Версии языка Perl и даты их выпуска*

Версия	Дата выпуска
Perl 1	Январь, 1988
perl 2	Июнь, 1988
perl 3	Октябрь, 1989
perl 4	Март, 1991
Perl 5	Октябрь, 1994

В настоящее время, как уже отмечалось ранее, устойчивой версией считается версия Perl 5.005_03, но уже существует версия 5.005_67. Их все можно получить с основного узла Web, поддерживающего язык Perl, по адресу http://www.Perl.com.

Замечание

В литературе по языку Perl принято, ссылаясь на сам язык, писать его с прописной буквой (Реп), а строчными буквами (Perl) обозначать интерпретатор языка. По образному высказыванию самого Ларри Уолла: "Perl — это ничего более как всего лишь интерпретатор Perl".

Характерные черты Perl

Perl — это интерпретируемый язык, оптимизированный для просмотра содержимого текстовых файлов, выделения из них информации и генерирования отчетов на основе этой информации, а также просто хороший язык для выполнения многих задач системного администрирования UNIX. Он обладает большим набором преимуществ как язык сценариев общего назначения, которые проявляются через его характерные черты и возможности.

Первым в цепочке достоинств языка Perl мы назовем его *интерпретируемость.* Конечно, некоторые программисты, прочитав это, скажут: "Ну вот, нашли себе достоинство. Посмотрим, как быстро будет выполняться программа Perl длиной, скажем, в тысячу операторов?". Что ж, замечание существенное, если рассматривать Perl как язык создания больших информационных систем, и совершенно не выдерживающее критики, если вспомнить, для чего он предназначен — задач администрирования и обработки текстовых файлов — небольших по размерам сценариев, решающих нетрадиционные задачи, для программирования которых могло бы потребоваться взаимодействие нескольких специализированных языков. Разработка подобных решений с помощью компилируемых языков программирования потребовала бы на много больше времени, чем использование одного интерпретируемого: ведь цикл разработки программ на таком языке короче и проще, чем на

компилируемом. Мы постепенно создаем программу, добавляя необходимые операторы, и сразу же получаем результаты, когда она завершена: интерпретатор Perl постепенно компилирует все операторы во внутренний байт-код и программа готова к выполнению, как только в ней поставлена последняя точка (точнее точка с запятой, завершающая Последний оператор). Для небольших по объему программ – это достаточное преимущество, так как отладка занимает много времени. Да, интерпретируемая программа, естественно, будет выполняться медленнее программы, представленной в формате двоичного файла и выполняющейся без предварительной обработки интерпретатором, но если в этом возникнет необходимость, то можно решение на языке Perl использовать в качестве прототипа для компилируемого языка, например C. Суммируя все сказанное, можно заключить, что Perl позволяет легко и быстро получить требуемое решение задачи, сочетая в себе элементы компилируемых и интерпретируемых языков программирования.

> **Замечание**
>
> Интерпретатор Perl, как, вероятно, заметил внимательный читатель, отличается от традиционных интерпретаторов тем, что программа транслируется в промежуточный байт-код, и только после этого выполняется. В традиционных интерпретаторах каждый вводимый оператор интерпретируется и сразу же выполняется, что может приводить к синтаксическим ошибкам во время выполнения. Perl-программа свободна от этого "недостатка", так как все синтаксические ошибки обнаруживаются во время трансляции в байт-код.

Вторым преимуществом использования Perl для решения соответствующих задач (мы имеем в виду сетевые возможности) является его доступность для большинства серверных платформ:

- практически все варианты UNIX;
- MS-DOS;
- Windows NT;
- Windows 95/98;
- OS/2;
- Macintosh.

Для всех перечисленных платформ разработаны и свободно распространяются интерпретаторы Perl вместе с документацией по их установке и работе, что приятно отличает его от других программных средств. И здесь уместно сказать несколько слов об условиях использования и распространения самого Perl и разработанных на нем программ. *(О том, где можно найти и получить интерпретатор Perl, см. главу 16.)*

Одним из способов распространения свободно распространяемого программного обеспечения, а именно таков интерпретатор Perl, является использование Общей открытой лицензии GNU. По условиям этой лицензии файлы исходного текста программного продукта распространяются совершенно свободно и могут быть использованы любым лицом. Однако любые версии программы, созданные путем модификации

этого кода, должны реализоваться также на условиях Общей открытой лицензии GNU, т. е. следует предоставлять файлы исходных текстов нового продукта любому, кто их захочет иметь. Этого зачастую вполне достаточно, чтобы защитить интересы автора первоначального программного продукта, однако может приводить к большому количеству производных версий исходного продукта, что приводит к "отчуждению" автора исходного продукта от процесса модификации его детища. Более того, в связи с большим количеством разнообразных версий, пользователям становится трудно определить, какая версия пакета является на текущий момент окончательной, будут ли написанные им сценарии, если речь идет о Perl, правильно работать с имеющейся у него версией, и т. п.

В связи с изложенными недостатками лицензии GNU, интерпретаторы языка Perl выпускаются на условиях лицензии Artistic License (Артистической лицензии), которая является некоторой вариацией лицензии GNU, и ее смысл заключается в том, что любой, кто выпускает пакет, полученный на основе Perl, должен ясно осознавать, что его пакет не является истинным пакетом Perl. Поэтому все изменения должны быть тщательно документированы и отмечены, выполнимые модули, в случае изменения, должны быть переименованы, а исходные модули должны распространяться вместе с модифицированной версией. Эффект от подобных условий заключается в том, что автор первоначального продукта всегда определяется как его владелец. При использовании Artistic License все условия Общей открытой лицензии GNU остаются в силе, т. е. она продолжает применяться.

Третьим преимуществом языка Perl можно назвать его практическую направленность, т. е. он создавался из практических соображений решения задач администрирования и разработки приложений для UNIX, а это означает, что он обладает следующими важными свойствами:

- полнотой;
- простотой использования;
- эффективностью.

Под полнотой Perl понимается его способность решать все возникающие в системе UNIX в связи с ее администрированием задачи. И это действительно так! Ведь язык Perl, как отмечалось выше, вобрал в себя все наилучшие возможности стандартных средств администрирования UNIX, перечисленных в табл. 1.2.

Таблица 1.2. *Стандартные средства администрирования UNIX*

Язык программирования	Характеристика
awk	Язык выделения по образцам информации из текстовых файлов
C	Компилируемый язык общего назначения для решения задач низкого уровня

shell	Основной командный язык запуска программ и скриптов, написанных на других языках программирования
sed	Потоковый редактор обработки текстовых файлов

Эти средства продолжают использоваться, так как каждое из них является прекрасным инструментом для выполнения тех задач, для которых они предназначены, однако все то, что можно выполнить, комбинируя эти средства, можно реализовать в одной Perl-программе, изучив только *один* язык. Но возможности Perl не ограничиваются только задачами администрирования. Подключаемые пакеты и модули позволяют легко и быстро решать и другие задачи, для которых, возможно, пришлось бы использовать язык программирования C. Начиная с версии 5.0, язык Perl поддерживает технологию объектно-ориентированного программирования, причем пакеты и модули можно оформить в виде объектов и использовать без знания содержащегося в них кода (хотя придется изучить большое количество объектных моделей со своими свойствами и методами).

Perl – это язык, на котором программист может *делать* свою работу, причем для выполнения одной и той же задачи Perl предлагает несколько средств ее реализации. Одни из них более сложны, другие – менее. Разработчик может выбрать то, которое ему более понятно и которое ему проще применить, не тратя времени на изучение более сложных возможностей. В этом заключается простота использования Perl, которая позволяет применять его как для реализации одноразовых утилит, так и для создания сложных, часто используемых приложений.

Perl является прямолинейным языком, а это означает, что простые программы не надо оформлять в виде головных процедур main, как это принято в большинстве процедурных языков программирования, или в форме класса, как принято в объектно-ориентированных языках программирования, т. е. не надо тратить время на дополнительное форматирование исходного текста программы, а просто начинать писать операторы Perl, которые будут немедленно обрабатываться интерпретатором. Именно в этом заключена эффективность языка программирования Perl.

Четвертое преимущество использования Perl связана с его дополнительными возможностями, позволяющими выполнять не только традиционные задачи администрирования UNIX и обработки текстовых файлов.

И здесь, в первую очередь, следует обратить внимание на простое включение в Perl-программу вызовов библиотечных процедур языка C, что позволяет использовать огромное количество кода, написанного для этого популярного языка. В поставку Perl входят утилиты, конвертирующие заголовки библиотек C в соответствующие эквиваленты языка Perl. Конвертирование осуществляется с помощью XS-интерфейса, который представляет собой простой программный интерфейс, преобразующий среду вызова функций C в среду вызова подпрограмм Perl. Последующий вызов функций C ничем не отличается от вызова подпрограмм самого Perl. Более того, программы Perl версии 5.0

легко интегрируются в приложения C и C++ через интерфейс, реализованный в наборе функций Perl_call_*.

Для работы с базами данных можно самому написать соответствующее приложение на языке C, а можно воспользоваться свободно распространяемыми модулями дополнительных расширений возможностей Perl, включающих работу с многочисленными популярными системами управления базами данных: Oracle, Ingres, Informix, Interbase, Postgre, Sybase 4 и др.

Способность Perl работать с сокетами TCP/IP сделала его популярным для реализации информационных систем взаимодействия с сетевыми серверами любых типов, использующих сокеты в качестве механизма обмена информацией. Именно эта возможность в сочетании с использованием Perl для создания CGI-сценариев послужила широкому распространению языка на других многочисленных платформах.

И в завершение перечисления достоинств Perl обратим внимание читателя на **пятое** преимущество его использования: так как изначально этот язык являлся свободно распространяемым, то вся наработанная документация также доступна совершенно бесплатно, а так как Perl, как язык сценариев очень популярен, то в Internet находится море документации по его применению для решения разнообразных задач.

(Некоторые адреса можно найти в главе 16.)

Области применения Perl

Наиболее широко Perl используется для разработки инструментов системного администрирования, однако в последнее время он получил огромную популярность в области разработки Internet-приложений: CGI-сценарии,

системы автоматической обработки электронной почты и поддержки узлов Web. В этом параграфе мы кратко охарактеризуем возможности Perl в каждой из указанных областей.

Системная поддержка UNIX

Как отмечалось ранее, именно задача соединения в одном языке программирования возможностей различных средств системного администрирования UNIX и послужила толчком к разработке и созданию языка Perl. Он и разрабатывался таким образом, чтобы оптимизировать решение именно этих задач, не прибегая к другим инструментам. На настоящий момент язык Perl является основным средством администрирования UNIX, который может выполнять работу нескольких других традиционных средств администрирования. Именно эта его универсальность и способствовала его широкому распространению среди системных администраторов и программистов UNIX, тем более, что он решает задачи обычно быстрее, чем другие аналогичные средства.

CGI-сценарии

Одной из первых, но продолжающей и по настоящее время широко применяться в Интернете технологией реализации динамических эффектов является технология CGI-сценариев, суть которой заключается в обработке информации, получаемой от пользователя, которую он вводит в поля формы страницы HTML, просматриваемой с помощью программы-обозревателя Internet. Информация из полей формы пересылается на сервер с помощью протокола HTTP либо в заголовке, либо в теле запроса и обрабатывается сценарием, который после анализа полученных данных выполняет определенные действия и формирует ответ в виде новой страницы HTML, отсылаемой обратно клиенту. Сценарий может быть написан, собственно говоря, на любом языке программирования, имеющем доступ к так называемым переменным среды, но сценарии Perl получили наибольшее распространение из-за легкости создания и оптимизационных возможностей языка Perl при обработке текстовых файлов. В Internet можно найти буквально тысячи примеров динамического CGI-программирования на Perl.

Его большая популярность для реализации подобных задач на UNIX-серверах Internet привела к тому, что разработчики серверов Internet, работающих в других операционных системах, стали включать возможность подключения сценариев Perl в свои системы. В настоящее время их можно использовать и на сервере Internet Information Server фирмы Microsoft для операционных систем семейства Windows, и на серверах Apache, NCSA и Netscape для операционной системы UNIX.

Обработка почты

Другая популярная область применения Perl – автоматическая обработка электронной почты Internet. Сценарии Perl можно использовать для фильтрации почты на основе адреса или содержимого, автоматического создания списков рассылки и для решения многих других задач. Одной из наиболее популярных программ для работы с электронной почтой является программа Majordomo, полностью реализованная средствами Perl.

Возможности Perl в этой области огромны и ограничиваются только фантазией разработчика. Можно, например, написать сценарий, который обрабатывает входящую почту и добавляет сообщения на заранее созданную страницу новостей, сортируя их по соответствующим тематикам, что позволяет быстро просматривать почту, не тратя время на чтение каждой полученной корреспонденции. По прошествии определенного времени сообщения удаляются со страницы.

Поддержка узлов Web

Узел Web – это ничто иное, как структурированное хранилище страниц HTML, которые являются обычными текстовыми файлами в определенном специальном формате, понимаемом программами просмотра их содержимого. Perl оптимизирован для обработки большого количества текстовых файлов, – поэтому его использование для анализа и автоматического изменения содержимого узла Web само собой вытекает из тех задач, для решения которых он специально и создавался. Его, например, можно использовать для решения задачи проверки правильности перекрестных ссылок на страницах узла Web, как, впрочем, и для проверки правильности ссылок на другие узлы (правда, здесь придется воспользоваться его сетевыми возможностями работы с сокетами).

Его возможности записи и чтения в/из сокетов позволяют использовать сценарии Perl для взаимодействия с другими узлами и получения информации на основе протокола HTTP. Следует отметить, что существуют даже серверы, написанные на Perl. Как упоминалось ранее, именно эти возможности Perl можно использовать для удаления со страниц HTML узла Web ссылок на несуществующие другие узлы.

Perl может работать и с протоколом FTP. Это позволяет автоматизировать получение файлов с других узлов, а в сочетании с его возможностями обработки текстовых файлов позволяет создавать сложные информационные системы.

* * *

В этой главе мы попытались кратко охарактеризовать сам язык Perl, очертить основные области его применения и привлечь внимание читателя к его дальнейшему изучению и внедрению в собственную практику. В конечном счете только сам программист решает, нужен ему соответствующий язык или нет. Мы думаем, что наш уважаемый читатель уже сделал свой выбор и надеемся, что он не покинет нас до самой последней страницы книги.

Вопросы для самоконтроля

1. Назовите полное наименование языка Perl.
2. Что послужило толчком для разработки и создания Perl?
3. Каково назначение Perl-программы?
4. В чем заключаются преимущества и недостатки интерпретируемых языков?
5. Перечислите основные достоинства языка Perl.
6. Перечислите области применения Perl.