Елена Бенкен

PHP, MySQL, XML
ПРОГРАММИРОВАНИЕ ДЛЯ ИНТЕРНЕТА
2-е издание

Санкт-Петербург
«БХВ-Петербург»
2008

УДК 681.3.068+800.92
ББК 32.973.26-018.1
Б46

Бенкен Е. С.

Б46 PHP, MySQL, XML: программирование для Интернета: 2-е изд. перераб. и доп. — СПб.: БХВ-Петербург, 2008. — 352 с.: ил. + CD-ROM

ISBN 978-5-9775-0280-1

Рассмотрены актуальные вопросы применения PHP для работы с базами данных MySQL и XML-документами. Описана установка и настройка сервера Apache с модулем PHP 5 и сервера MySQL 5. Изложены основы языка PHP и его расширения. Подробно излагается работа с базами данных MySQL от построения запросов до использования утилит командной строки. Приведены базовые сведения о языке XML. Описан формат новостной ленты RSS и представлены практические примеры обработки XML-документов с помощью расширений PHP 5, таких как SimpleXML, DOM-функциями и функциями событийного программирования SAX. Во втором издании расширено описание объектной модели PHP, уделено внимание проблеме русификации Web-приложений. Компакт-диск содержит дистрибутивы Web-сервера, модуля PHP и сервера MySQL, распространяемые по лицензии GNU/GPL, а также примеры из книги.

Для Web-программистов

УДК 681.3.068+800.92
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Анна Кузьмина*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Виктория Пиотровская*
Дизайн обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 28.05.08.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 28,38.
Тираж 2500 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.002108.02.07 от 28.02.2007 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ОАО "Техническая книга"
190005, Санкт-Петербург, Измайловский пр., 29.

ISBN 978-5-9775-0280-1

Оглавление

Посвящаю, как и все, что я делаю,

моим детям Владимиру и Константину

Вступительное слово

Автор книги, Елена Сергеевна Бенкен, является высококвалифицированным специалистом и обладает многолетним успешным опытом преподавания. Это профессионал, который не только отлично разбирается в предметной области, но и обладает способностью увлечь аудиторию и донести свои знания до нее. Методика подачи предложенного в книге материала была апробирована на сотнях слушателей самых разных компаний, которые проходили обучение в нашем учебном центре. Благодаря этому при написании книги учитывалось, какой материал был наиболее тяжел для освоения слушателями, какие возникали проблемы, какие навыки будут важны для практической работы. Система изложения построена таким образом, чтобы читатели получили не только хорошо структурированный материал, но и приобрели реальные практические навыки. Поэтому в книге предусмотрены различные задания для самостоятельной работы.

От имени администрации и преподавательского состава учебного центра мы рекомендуем книгу Елены Сергеевны Бенкен как специалистам, так и начинающим изучение IT-технологий. Книга написана хорошим профессиональным языком, отличается последовательностью изложения, легко читается. Примеры и задания помогают в восприятии материала и иллюстрируют возможности практического применения полученных знаний. Книга Елены Сергеевны входит в серию книг, подготовленных преподавателями факультета переподготовки специалистов СПбГТУ (**www.avalon.ru**). Мы гордимся уровнем квалификации и профессионализмом наших преподавателей, написанные ими книги пользуются заслуженной популярностью, и поэтому мы решили начать объединение этих книг под единой серией.

Как сотрудники крупного учебного центра мы понимаем, как важен контакт между преподавателем и обучаемыми, как необходима для успешного усвоения материала возможность задать преподавателю вопрос и получить квалифицированный ответ. Поэтому если после прочтения книги у вас появились

вопросы, на которые вы не нашли ответа, если вы хотите получить еще больше практических заданий, чтобы повысить свою квалификацию, если вы просто хотите поблагодарить автора данной книги, то можете сделать это на форуме нашего учебного центра в соответствующем разделе: **http://forums.avalon.ru**.

На ваши вопросы ответит лично автор книги, а также другие преподаватели нашего учебного центра.

Директор ИГУИ СПбГПУ	Александр Речинский
Руководитель направления "Разработка программного обеспечения"	Егор Орлов
Руководитель направления "Сетевые технологии"	Дмитрий Кетов

Об учебном центре

Институт государственного управления и информатизации Санкт-Петербургского государственного политехнического университета (известный в Интернете как AVALON.RU) представляет собой сегодня многопрофильный учебный центр, занимающий лидирующие позиции в области обучения информационным технологиям в Северо-Западном регионе. В 2007 году на различных образовательных программах у нас успешно прошли обучение более 7000 человек.

Благодаря высококвалифицированному преподавательскому составу, мощной технической базе и многолетнему опыту мы предлагаем действительно качественное обучение. Все наши преподаватели являются специалистами в своих областях и грамотными педагогами. Это подтверждается различными международными сертификатами учебного центра и преподавателей, а также тысячами положительных отзывов слушателей. За последние десять лет наш учебный центр выработал определенный стиль, методику преподавания, у нас работает команда высококвалифицированных преподавателей. Основные направления деятельности:

- *"Академия информатики для школьников"*. Программа обучения школьников информационным технологиям: базовый уровень, разработка программного обеспечения, сетевые технологии, дизайн. Общий объем программы составляет 16 семестров обучения. На сегодняшний день в программе участвуют около 700 школьников;
- *"Краткосрочные компьютерные курсы"*. На постоянной основе проводится более 30 курсов и семинаров различного уровня сложности по разным направлениям: офисное, интернет-технологии, компьютерный дизайн, 3D-графика, компьютерные системы и сети. Одним из направлений деятельности является комплексное обучение персонала по заказам компаний;

- *"Авторизованные и авторские курсы для IT-специалистов".* Предлагается более 50 курсов и семинаров разного уровня сложности. Проводится обучение программным продуктам и технологиям как в ранге авторизованных курсов (Microsoft, Citrix, D-Link, Linux и т. д.), так и в виде авторских курсов (Cisco, Autodesk, Adobe, Oracle, Analog Devices, EMC и т. д.);
- *"Школа практического программирования".* Совместный проект факультета с ведущими предприятиями города. Представляет собой систему тренингов, проводимых совместно преподавателями факультета и сотрудниками предприятий, с целью моделирования ситуаций работы над реальными проектами в области разработки программного обеспечения;
- *"Второе высшее образование в области информационных технологий".* Обучение проводится в течение 2,5—3 лет по специальностям: "Математическое обеспечение и администрирование информационных систем", "Вычислительные машины, системы, комплексы и сети", "Дизайн";
- *"Второе высшее образование в экономике".* Обучение проводится в течение 2,5—3 лет по специальностям: "Экономика фирмы", "Государственное и муниципальное управление";
- *"Высшее образование в области информационных технологий".* Обучение проводится по специальности "Математическое обеспечение и администрирование информационных систем";
- *"Высшее образование в области экономики".* Обучение проводится по специальности "Государственное и муниципальное управление";
- *"Авторизованное и авторское тестирование".* Кроме авторизованного тестирования (авторизованный центр тестирования Prometric) проводится и авторское тестирование более чем на 100 различных тестах, разработанных преподавателями института.

Введение

Для кого написана эта книга

Книга предназначена для тех, кому надо самостоятельно осваивать принципы работы Интернета, программирование на PHP и работу с базами данных, в первую очередь MySQL. Полезной книга может оказаться и тем, кому сегодня начальник сказал, что через месяц предстоит работа с XML-документами, причем данные из них придется помещать на Web-сайте вашей фирмы.

Предполагается, что читатель знаком с языком HTML, имеет навыки написания Web-страниц на HTML, а также твердо владеет основными пользовательскими навыками работы на компьютере с операционной системой Windows.

В Интернете имеется достаточно информации, чтобы научиться использовать любые технологии. Проблема в том, что документация на PHP и MySQL написана отнюдь не для новичков, не говоря уже о документации на языке XML.

Автор этой книги ежедневно занят ответами на вопросы студентов относительно Apache, PHP и MySQL. Поэтому есть некоторая надежда, что в книге удастся объяснить то, что чаще всего является камнем преткновения при первом знакомстве с предметом.

Книга предназначена для самостоятельного и независимого изучения материала. Это означает, что, скачав из Интернета необходимые пакеты, вы сможете установить их и начать работать, не заглядывая в другие книги.

Внимание! Ни в одной из глав не содержится исчерпывающей информации по обсуждаемому вопросу! Цель автора — научить читателя основам программирования на PHP так, чтобы он получал удовольствие за свои деньги.

Итак, наша задача:

- установить Web-сервер Apache, модуль PHP и сервер MySQL в операционную систему Windows, научиться управлять этими серверами и узнать основы их администрирования;
- узнать принципы передачи информации в Интернете;

- научиться писать и запускать сценарии на PHP;
- познакомиться с основами SQL, научиться создавать базы данных, составлять запросы к ним и работать с базами данных с помощью PHP-сценариев;
- изучить основы языка XML, обрабатывать XML-документы с помощью PHP.

Как работать с книгой

Для полноценной работы вам потребуется компьютер с операционной системой Windows, на котором у вас есть право устанавливать программы. Кроме того, вам потребуется выход в Интернет для того, чтобы скачать оттуда дистрибутивы используемых программ.

В ходе чтения следует выполнять на компьютере примеры, описываемые в книге. Каждый пример стоит изменять и переделывать самому с тем, чтобы лучше понимать, как он работает или как сделать так, чтобы все перестало работать.

Автор приложил все усилия, чтобы изложить материал с наибольшей точностью, но не исключает возможности ошибок и опечаток. Автор также не несет ответственности за последствия использования сведений, изложенных в книге.

Источники информации

Главное, чему должен научиться человек, знакомящийся с интернет-технологиями, — это усвоению информации. Все остальное приложится.

Существуют два типа источников информации: первоисточники (если пользоваться термином прошедшей эпохи), содержащие исходные коды программ и скомпилированные из них пакеты, а также документация, поставляемая фирмой-изготовителем или созданная сообществами программистов. Скачивать программы следует только с серверов известных сообществ или фирм-производителей. С документацией сложнее: лучше читать ее в оригинале на английском языке, в имеющихся переводах порой теряется не только изящество изложения, но и смысл. К тому же документация написана для тех, кто уже много что знает, а теперь вот еще решил изучить и этот вопрос. Некоторые книги тоже можно отнести к первоисточникам — это книга для подготовки к сертификационному экзамену по PHP, а также некоторые книги, написанные разработчиками PHP и MySQL. Правда, эти книги страдают теми же недостатками, что и документация производителя.

Итак, первоисточники в Интернете:

- **http://apache.org**;
- **http://php.net**;
- **http://dev.mysql.com**;
- **http://w3schools.com** — школы W3C по XML;
- **http://www.w3.org/** — содержит всеобъемлющую информацию по стандартам Интернета.

Печатная литература — Zend PHP certification. Study guide. Sams Publishing, Indianapolis, Indiana 46240 USA.

Вторая категория источников информации — труды тех, кто работает с PHP и MySQL. Можно рекомендовать следующие русскоязычные сайты:

- **http://phpclub.ru**;
- **http://opennent.ru** — этот сайт в основном посвящен операционным системам, в первую очередь семейству UNIX, но посмотрите внимательно — здесь прекрасные статьи, отслеживаются все новости и даются ссылки на оригиналы и переводы статей;
- **http://phpworld.ru** — сайт посвящен PHP 5.

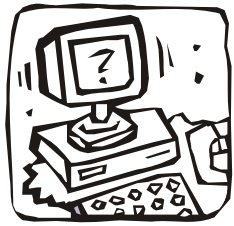
Перевод на русский язык документации по XML: **http://xml.nsu.ru** — переводы школ консорциума W3C по XML.

Благодарности

- Моему мужу Александру, чья непоколебимая вера в меня помогла пережить немало сложных моментов, а также моим родителям.
- Константину Хоматьяно, познакомившему меня с PHP. Костя, я надеюсь на снисхождение.
- Егору Орлову — за идею книги и помощь в ее создании.
- Нашим студентам и слушателям — за удовольствие наблюдать огонь понимания и интереса, который зажигается в их глазах.
- Игорю Шишигину, проведшему автора по всем положенным кругам публикации книги.

ЧАСТЬ I

Основы языка PHP



Глава 1

Основы клиент-серверного взаимодействия в Интернете

При создании Web-страницы на HTML или сценария на JavaScript текст с разметкой и программным кодом сохраняется в файле с расширением html. Для того чтобы просмотреть результат работы, в Windows достаточно открыть страницу двойным щелчком по значку файла в окне Проводника. При этом страница, содержащая каскадные стили или сценарий на JavaScript, может быть и динамической, т. е. ее вид меняется в результате действий пользователя или просто с течением времени. Пользователь может просмотреть исходный код страницы и понять, в результате работы какого именно фрагмента кода происходят те или иные изменения внешнего вида страницы.

Но в Интернете есть множество страниц, работающих иначе. Например, на Web-сайте может проводиться опрос: "Какую марку автомобиля вы предпочитаете?" Ответ вписывается в поле формы. Введенный текст отправляется по щелчку кнопкой с надписью "Отправить ответ". После этих действий на экране появляется новая страница, на которой, например, написано, что выбранный вариант ответа оказался в числе ответов еще 15% посетителей этого сайта, и выводится диаграмма распределения предпочтений посетителей. При этом в исходном коде страницы нет ничего поясняющего, откуда взялись эти сведения.

Выяснение, что же происходит в этом случае, начинается с изучения того, как браузер на пользовательском компьютере взаимодействует с сервером, с которого была получена страница с информацией о выборе марки автомобиля посетителями сайта.

Но сначала дадим определения некоторых понятий. Чем точнее сделаны определения, тем легче будет разобраться в обсуждаемом предмете.

Необходимые определения

Будем называть компьютер, работающий в Интернете, *хостом*.

Сéрвер (от англ. *serve* — служить) — это компьютер и программное обеспечение на нем, предоставляющие клиенту доступ к определенным ресурсам.

Web-сервер — это сервер, предоставляющий доступ к сайтам World Wide Web (WWW). Когда пользователь дает браузеру команду открыть тот или иной документ на сайте, браузер подключается к соответствующему серверу и запрашивает у него содержимое документа.

IP-адрес

Каждый хост характеризуется уникальным адресом. По действующему ныне стандарту этот адрес состоит из четырех целых положительных чисел, разделенных точками. Каждое число не может превышать 255. Заданный таким образом адрес называют *IP-адресом* хоста. Например: 192.168.1.66.

Если ваш домашний компьютер выходит в Интернет, то он получает IP-адрес от провайдера.

Запоминать IP-адреса трудно, да и нет необходимости: всем IP-адресам Web-серверов в Интернете однозначно ставится в соответствие *полное доменное имя*, например, **www.avalon.ru**.

Среди IP-адресов есть специальные, например, так называемый *адрес интерфейса обратной петли*. Это адрес, по которому каждый компьютер может обратиться к самому себе, используя сетевое программное обеспечение. Этим обращением можно проверить, функционируют ли программы поддержки работы с сетью.

Представьте себе, что вы ждете гостей, а их все нет и нет. Решив выяснить причину этого, вы начинаете с проверки, исправен ли звонок на входной двери. Выйдя на лестничную площадку, вы нажимаете кнопку звонка. На языке компьютерных сетей это и есть обращение к своему хосту по адресу интерфейса обратной петли.

Обратиться по этому адресу можно только к самому себе так же, как человек может обратиться "я" только к себе. Для интерфейса обратной петли выделен адрес 127.0.0.1. Ему соответствует доменное имя localhost. Мы будем вводить в адресной строке браузера это имя для того, чтобы обратиться к Web-серверу, установленному на нашей машине.

Протокол

Браузер является программой-клиентом. Клиент посылает запросы серверу.

Программа-клиент взаимодействует с сервером, используя определенный протокол.

Протоко́л — это набор правил для передачи информации. Программа-клиент и программа-сервер могут работать как на одном и том же компьютере, так и на разных.

Таким образом, клиентский и серверный компьютеры взаимодействуют друг с другом, используя адреса и доменные имена для поиска ресурсов.

Порт

На одном и том же сервере может работать несколько программ разного назначения. Для определения того, какая программа требуется клиенту, используется понятие порта. *Порт* — это номер, указывающий на программу, к которой хочет обратиться клиент. Например, Web-сервер обычно идентифицируется портом 80. В свою очередь, клиентский браузер тоже использует порт, но программе-клиенту выделяются порты с номерами, превышающими число 1024.

Информация о номере порта, по которому работает Web-сервер, вовсе не является избыточной для начинающего Web-программиста. Нам придется еще не раз вспоминать об этом номере. Дело в том, что по каждому порту может работать только один сервер (иногда вместо слова "сервер" говорят "служба"). Если вы пытаетесь запустить вторую копию Web-сервера, то получите возмущенное сообщение системы с указанием того, что 80-й порт занят. Сообщение может быть неудобочитаемым, так что единственная ниточка, позволяющая вам понять, что происходит, — это указанный номер порта.

Итак, передача информации между клиентом и сервером осуществляется по определенным правилам — *протоколу*. Основным протоколом, по которому взаимодействуют компьютеры при запросе и получении Web-страницы, в настоящее время является HTTP-протокол (HyperText Transfer Protocol, протокол передачи гипертекста) версии 1.1.

Правила взаимодействия между компьютерами во Всемирной паутине регламентируются документами, издаваемыми консорциумом World Wide Web Consortium (W3C), осуществляющим стандартизацию средств Интернета.

Протокол HTTP

Полное описание HTTP содержится в спецификации, опубликованной на сайте **http://www.w3.org/Protocols/** или в RFC 2616 (**ftp://ftp.isi.edu/in-notes/rfc2616.txt**). В данной главе лишь кратко и неформально излагаются необходимые сведения.

Запрос клиента

Работая с браузером, пользователь набирает адрес интересующего его ресурса и дает команду выполнить запрос. В результате этих действий в браузере формируется запрос серверу, состоящий из нескольких строк.

Первая строка обычно начинается с команды `GET` или `POST`. Эти команды иначе называют *методами*. Оба метода используются для запроса страниц с Web-сервера, а различия между ними мы кратко рассмотрим в этом разделе и подробнее — в дальнейшем, когда вы сможете наблюдать их на примерах. Первая строка запроса может выглядеть так:

```
GET avalon.ru/index.html HTTP/1.1
```

Здесь `avalon.ru/index.html` — URI (Uniform Resource Identifier, универсальный идентификатор ресурса).

После этой строки идут строки заголовков (Headers). Каждый заголовок имеет определенное стандартом имя, после которого ставится двоеточие, а после двоеточия пишется значение, которое требуется знать серверу. Эти значения передают серверу информацию о браузере: каков тип браузера, какие данные он может обработать, на каком языке и т. д. В зависимости от того, что сообщит о себе браузер, сервер и будет решать, какие данные следует отобрать для передачи клиенту.

Приведем примеры используемых заголовков.

Заголовок `User-Agent` сообщает о типе браузера клиента:

```
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.7.2)
Gecko/20040803
```

В заголовке `Accept` указываются типы данных, которые следует передавать клиенту:

```
Accept: image/gif, image/jpeg, image/*, */*
```

Типы данных указываются в формате стандарта MIME.

MIME (Multipurpose Internet Mail Extensions) — многоцелевые расширения почтового стандарта Интернета. Этот стандарт описывает, как пересылать по электронной почте исполняемые, графические, мультимедийные и другие

данные, не являющиеся простым текстом. Изначально MIME был создан для указания, какого типа документ вложен в сообщение электронной почты. MIME-тип задается в виде "тип/подтип". Например: `text/html`.

Если надо указать несколько MIME-типов, то они разделяются запятыми. Символ звездочки означает любой тип. В приведенном заголовке `Accept` типы данных указываются в порядке предпочтения браузера. То есть при наличии данных, представленных в различных форматах, в первую очередь должны передаваться графические данные в формате `image/gif`. Если данных в этом формате нет, то передаются данные в формате `image/jpeg`, в отсутствие которых можно передавать любой графический формат, а прочие данные браузер готов получать в любом виде.

Стандарт MIME определяет 7 типов данных:

- `application`;
- `audio`;
- `image`;
- `message`;
- `multipart`;
- `text`;
- `video`.

Заголовок `Referer` указывает на страницу, с которой пользователь перешел по ссылке на текущую. Конечно, такой ссылки могло и не быть, тогда этот заголовок не посылается. Пример:

```
Referer: www.host.ru/index.html
```

Каждый заголовок завершается символом конца строки (`\n`), а после заголовков идет пустая строка, т. е. два символа `\n`.

Если клиент делает запрос методом `GET`, то на этом запрос и заканчивается. Если же запрос идет методом `POST`, то после пустой строки могут идти данные, передаваемые в запросе. Например, это могут быть данные о предпочитаемой марке автомобиля или файл с фотографией, который вы закачиваете на сайт для размещения в фотоальбоме. В случае отправки таких данных необходимо передать их размер в соответствующем заголовке.

При использовании метода `GET` данные могут быть переданы прямо в адресной строке так:

```
GET avalon.ru/index/html?данные  HTTP/1.1
```

Теперь давайте представим, что нам надо заполнить поле формы на Web-странице и отправить эти данные на сервер методом `GET`. Метод указывается

в виде значения атрибута `method` в открывающем теге присланной нам формы. В полученной форме есть поле `login` (имя пользователя), мы заполняем его, нажимаем кнопку для отправки и видим в адресной строке браузера, что запрос идет в таком виде:

```
http://www.site.ru/index.html?login=Mike
```

Если же имя пользователя содержит пробел, русские буквы или специальные символы, то эти символы кодируются перед отправкой во избежание ошибочных ситуаций в запросе.

Бродя по Интернету, вы наверняка видели в адресной строке длинные адреса, включающие в себя символы `%`. Это и есть закодированные значения передаваемых параметров. При отправке методом `GET` нескольких параметров мы увидим в адресной строке примерно такую запись:

```
http://www.site.ru/index.html?login=Mike&age=25
```

При посылке пар "параметр=значение" они разделяются символом `&`.

Ответ сервера

Рассмотрим теперь, как Web-сервер обрабатывает запрос клиента.

Web-сервер проверяет, есть ли в его распоряжении запрошенный ресурс, и имеет ли клиент право его получить. Если ресурс может быть предоставлен, то сервер отправляет клиенту ответ, начинающийся со статусной строки:

```
HTTP/1.1 200 OK
```

Здесь `200` — это код ответа, `OK` — расшифровка этого кода.

Если по каким-то причинам запрошенная страница не может быть передана, то код и расшифровка будут другими. Полный перечень кодов содержится в спецификации протокола, но в табл. 1.1 приведены некоторые часто встречающиеся варианты ответа.

Таблица 1.1. *Коды ответа сервера*

Код ответа	Описание
`200 OK`	Клиентский запрос успешен, и в Web-ответе сервера содержатся запрошенные данные
`403 Forbidden`	Запрос отклонен по причине, которую Web-сервер не хочет или не может сообщить клиенту
`404 Not Found`	Документ с указанным адресом не существует
`500 Internal Server Error`	Код указывает на возникновение аварийной ситуации в какой-то части сервера

После строки статуса сервер посылает клиенту заголовочные данные о самом себе и о запрошенном документе, например:

- дата отправки документа: `Date: Fri, 22 Sep 2006 08:13:54 GMT`;
- тип передаваемого клиенту документа: `Content-type: text/html`;
- размер документа — `Content-Length` — количество байтов, пересылаемых в теле ответа. Пример: `Content-Length: 26457`;
- время последнего изменения пересылаемых данных — `Last-Modified`. Пример: `Last-Modified: 22 Sep 2006`.

В случае если запрос оказался успешным, после заголовков Web-сервер отправляет клиенту запрошенные данные.

CGI

Большое количество Web-приложений основано на использовании внешних программ, управляемых Web-сервером. Такие приложения генерируют информацию динамически, выбрав ее из баз данных или других источников. Для связи между Web-сервером и вызываемыми программами используется стандарт CGI (Common Gateway Interface, общий интерфейс шлюза).

Интерфе́йс — это способ соединения или связи. Интерфейс определяет границу между сущностями. В нашем случае сущностями являются Web-сервер и PHP-программа, а интерфейс определяет способ их взаимодействия. Стандарт CGI разработан таким образом, чтобы для создания приложений можно было использовать любой язык программирования.

Программу, которая работает совместно с Web-сервером по стандарту CGI, принято называть *сценарием (скриптом)* или CGI-программой. Получив от клиента HTTP-запрос, Web-сервер определяет, что запрос адресован какому-либо сценарию. Например, в запросе клиента содержатся имя пользователя и пароль для регистрации на сервере. Web-сервер запускает запрошенный сценарий на исполнение. При запуске сценария сервер должен выполнить несколько операций: вызвать сценарий и обеспечить его необходимыми данными, посылаемыми от браузера, снабдить сценарий значениями переменных окружения. *Переменные окружения* содержат информацию о браузере, который посылает запрос, и сервере, который его обрабатывает. Рассматриваемый нами сценарий должен проверить, что присланное имя пользователя (логин) зарегистрировано на сервере, а пароль указан верно. Сценарий читает параметры запроса и выполняет проверку, обращаясь к базе данных или файлу с паролями. Результатом работы сценария является HTML-страница.

Сервер должен обработать результат работы скрипта, в том числе обеспечить включение дополнительной заголовочной информации, необходимой для браузера, чтобы тот мог успешно прочитать полученные данные. Web-сервер отправляет обработанный результат в ответ на запрос клиента. В странице ответа может быть указано, что пользователь прошел регистрацию и может быть допущен к определенным ресурсам сервера.

Переменные окружения CGI чувствительны к регистру, и каждая из переменных определена в спецификации CGI.

Вот некоторые из них.

- `QUERY_STIRNG`. Скрипты используют эту переменную для того, чтобы получить информацию в текстовой форме, которая следует справа от знака вопроса после URL, переданного в запросе от пользователя скрипту для обработки. Например, при запросе вида

  ```
  GET avalon.ru/index/html?a=1&b=3 HTTP/1.1
  ```

  значение переменной `QUERY_STIRNG = "a=1&b=3"`.

- `REQUEST_METHOD` используется для того, чтобы определить тип HTTP-запроса, который послан браузером серверу. Например, если браузер посылает запрос методом `GET`, то переменная окружения имеет следующее значение:

  ```
  REQUEST_METHOD = GET
  ```

- Переменная `SCRIPT_NAME` используется для того, чтобы определить путь к скрипту, который будет запущен сервером. Например, если имеется URL **http://www.avalon.ru/our_site/somescript.php**, то переменная окружения примет следующее значение:

  ```
  SCRIPT_NAME = our_site/somescript.php
  ```

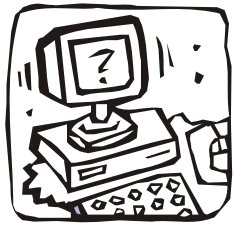
- Переменная `HTTP_ACCEPT` служит для того, чтобы определить, какие MIME-типы может принимать браузер. Они указаны в HTTP-заголовках, которые браузер послал серверу. Например, переменная окружения может принимать следующее значение:

  ```
  HTTP_ACCEPT = audio/mpeg, text/html, text/plain
  ```

- Переменная `HTTP_USER_AGENT` используется для того, чтобы идентифицировать тип браузера, который делает запрос серверу. Значение переменной получается из заголовков запроса клиента. Например:

  ```
  HTTP_USER_AGENT = Mozilla/2.01 Gold(Win95PE)
  ```

Данные из HTML-формы передаются в сценарий в виде набора переменных. CGI-сценарий использует эти данные при выполнении программы.



Глава 2

Установка Web-сервера Apache и модуля PHP 5 в Windows

Данная книга рассчитана на среднестатистического российского Web-программиста, который разрабатывает сценарии для сайта на компьютере под управлением операционной системы Windows (вследствие ограниченной функциональности не подходит версия Windows XP Home Edition). Созданные сценарии переносятся на сервер хостера, работающий под управлением одной из систем семейства UNIX.

Преподавательский опыт автора заставляет просить читателя ставить и использовать именно те программные продукты, которые описаны в этой книге. Установка различных пакетов, содержащих "в одном флаконе" и Web-сервер, и сервер баз данных, и текстовый редактор, приводит к тому, что написать несложный код на PHP начинающий Web-мастер может, а вот понять, как он работает — нет. Кроме того, разобраться, как при необходимости изменить настройки этих пакетов, оказывается сложно, к тому же навыки такой настройки мало что дают, т. к. на рабочих серверах они выполняются иначе. Web-технологии непросты, и попытки их упростить приводят к скверному результату.

При установке программ вы должны работать от имени пользователя, имеющего достаточные привилегии, чтобы писать файлы в системный каталог Windows и каталог Program Files системного диска.

Установка сервера Apache

Для установки Web-сервера Apache для работы необходимо скачать дистрибутив с сайта **apache.org**. Нам потребуется бинарный пакет (Win32 Binary) в файле с расширением msi (MS Installer).

> ***Примечание***
>
> При скачивании пакетов используйте только проверенные сайты, этим вы обеспечите безопасность своего компьютера, а также сможете быть уверены, что скачали самые свежие версии программного обеспечения.

Работа сервера Apache версии 2 с модулем PHP до сих пор в документации называется экспериментальной. Автор проводит этот эксперимент несколько лет и может порекомендовать читателю продолжить его, тем более, что ветка Apache 2 создавалась в известной мере для оптимизации работы Web-сервера с учетом особенностей архитектуры операционной системы Windows.

Приведенные далее инструкции касаются установки сервера Apache 2.0.

> ***Примечание***
>
> Установка Apache версии 2.2 и соответствующего ему модуля PHP 5 отличается от описанной далее процедуры и в настоящее время вряд ли может быть рекомендована начинающим Web-мастерам.

После запуска на исполнение пакета с дистрибутивом вам потребуется ответить на ряд вопросов (рис. 2.1).

Если вы работаете дома, то, возможно, у вас нет ни сети, ни, следовательно, домена, в который может входить ваш компьютер. Тогда в первом поле следует указать значение localdomain.

Рис. 2.1. Запрос информации о сервере

Если же ваш компьютер работает в сети, то узнайте у администратора сети название домена и укажите его здесь. Впрочем, если вы работаете в домене, то инсталлятор может и сам вписать имя домена в поле ввода.

Имя сервера — второе, что в окне на рис. 2.1 вам надо указать. Таким именем может быть либо полное доменное имя компьютера, либо его IP-адрес, либо специальное имя localhost.

В третьем поле следует указать электронный адрес администратора сервера — в зависимости от ситуации можете ввести подлинный или вымышленный (последний вполне годится для домашнего сервера, который кроме вас никто не увидит).

Наконец, надо разрешить запускать Apache автоматически при запуске системы как службу Windows и открыть 80-й порт для работы с сервером.

Затем придется ответить на вопрос о том, в какой каталог следует установить Apache. Инсталлятор предложит каталог Program Files на системном диске. Это не очень удобно, но всякие изменения в этом пункте приводят к неоправданным для новичка сложностям.

После прохождения всех окон вы увидите, как установится пакет, а затем и запустится. Об успешности запуска надо судить по черному DOS-окну, в котором могут появиться ошибки. Не появились — отлично, черное окно закроется, что означает успешный запуск сервера.

Откройте окно браузера и отправьте HTTP-запрос вашему серверу, указав адрес: **http://localhost**. Вы должны увидеть то, что показано на рис. 2.2.

Теперь посмотрите, какие процессы запустились в системе после установки Apache: нажмите комбинацию клавиш <Ctrl>+<Alt>+<Del> и в окне диспетчера задач Windows выберите вкладку **Процессы**.

Вы обнаружите несколько процессов, связанных с Apache. Apachemonitor — это процесс, слушающий порт 80 и ожидающий клиентские запросы. На эти запросы он не отвечает сам, а передает запросы дочерним процессам Apache, которые и занимаются их обработкой. Таких дочерних процессов запускается несколько — в расчете на большое количество запросов клиентов. Монитор следит за тем, чтобы количество дочерних процессов всегда было достаточным с учетом загрузки Web-сервера, и при необходимости создает новые дочерние процессы или удаляет лишние.

Посмотрим теперь, что за каталоги мы создали в процессе установки, и что в них находится. Зайдите в каталог Program Files на системном диске и найдите в нем каталог Apache Group, а в нем каталог Apache2. Каталог bin внутри Apache2 содержит исполняемые модули сервера, некоторые библиотеки и вспомогательные программы.

Каталог htdocs — место хранения HTML-страниц и сценариев на языке PHP. Содержимое только этого каталога может передаваться клиенту по запросу.

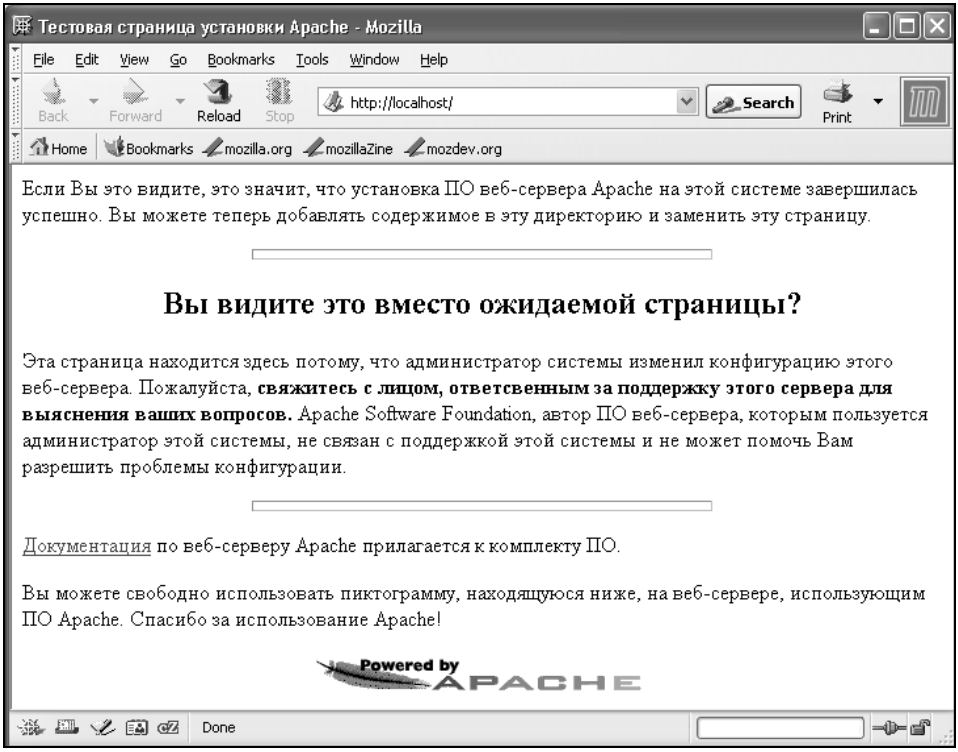


Рис. 2.2. Стартовая страница Web-сервера

Директивы конфигурации Apache

Каталог conf содержит конфигурационный файл httpd.conf сервера Apache. Apache читает этот файл при запуске. Строки конфигурационного файла, начинающиеся символами `#`, — это комментарии, которые Apache не читает. Файл httpd.conf содержит директивы, определяющие режим работы Apache. Настройка Apache в основном состоит в том, чтобы дать директивам файла httpd.conf нужные значения. После установки Apache надо проверить, какие значения получили следующие директивы, и, при необходимости, поменять их значения.

- Директива `ServerAdmin` получает в качестве значения адрес администратора сервера (**you@your.address**), который будет появляться в сгенерированных сервером сообщениях об ошибках.
- Директива `ServerName` задает имя хоста, на котором работает Web-сервер, это должно быть зарегистрированным именем DNS. На домашнем компьютере в качестве имени хоста можно указать имя localhost.

- В директиве `DocumentRoot` задается каталог, из которого берутся передаваемые клиентам документы.
- Директива `DirectoryIndex` указывает имя файла, используемое в качестве страницы-указателя или оглавления.

  Например:

  ```
  DirectoryIndex index.html index.php
  ```

  Эта директива позволяет задать название документа, возвращаемого по запросу, который не содержит в строке URL названия документа.

  Например, в адресе **http://www.shop.com/** отсутствует название документа, поэтому будет возвращен документ, указанный в директиве `DirectoryIndex`. Поскольку в директиве указано имя файла index.html, сервер передаст клиенту документ index.html из каталога DocumentRoot на сервере.

  В директиве `DirectoryIndex` можно задать несколько имен файлов. Если первый документ, указанный в строке, не найден в каталоге, то сервер ищет следующий и, в случае успеха, передает его клиенту.

  Эта настройка также предоставляет важный уровень защиты информации. По умолчанию, если клиент указывает адрес URL каталога, то сервер Apache передает в ответ список файлов, имеющихся в этом каталоге. Создав в каталоге индексный файл, вы лишите пользователей возможности получить список всех файлов в этом каталоге.

  Для разработчика наличие индексного файла неудобно, поэтому удалите из каталога htdocs все индексные файлы. Обратившись после этого к своему серверу, вы больше не увидите страницы с "апачевским" перышком, только информацию об Apache и пустой каталог.
- Надо также найти директиву, отвечающую за выбор кодировки документа по умолчанию. Эта директива позволит генерировать в ответе сервера указание браузеру переключиться в указанную кодировку. В версии Apache 2.0.45 такой директивой является `AddDefaultCharset`.

  Установите ее значение: `AddDefaultCharset cp-1251`.
- Директива `AddType` полезна для добавления новых типов предоставляемых клиентам документов на основе использования MIME-типов. Вам может потребоваться указать серверу, что документ является PHP-сценарием (судя по расширению имени файла), и требуется использовать модуль PHP для его обработки. Директива `AddType` дает такую возможность.
- `ErrorLog`. При помощи этой директивы задается местоположение журнальных файлов, в которых регистрируются ошибки доступа к серверу. В файле, указанном в директиве `ErrorLog`, сервер сохраняет сообщения диагностики, включая сообщения об ошибках, выдаваемые сценариями CGI.