МИРОВОЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР

Изучаем PHP и MySQL

Сможете предотвращать неприятные проблемы, источником которых являются веб-формы

Научитесь создавать динамичные сайты на основе баз данных

Наработаете практический опыт в программировании на PHP и под MySQL

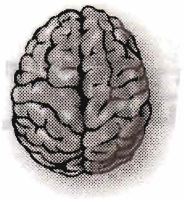
Прочно усвоите ключевой синтаксис и понятия

Закрепите полученные знания при помощи многочисленных упражнений

Линн Бейли и Майкл Моррисон

эксмо O'REILLY®

МИРОВОЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР



Head First PHP & MySQL

Lynn Beighley
Michael Morrison

O'REILLY®

Beijing • Cambridge • Köln • Sebastopol • Taipei • Tokyo

МИРОВОЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР

Изучаем PHP и MySQL

Линн Бейли
и Майкл Моррисон

ЭКСМО

Москва
2010

УДК 004.45
ББК 32.973.26-018.2
Б 35

Перевод с английского и редакция ЧП «Айдиономикс»

Бейли Л.

Б 35 Изучаем PHP и MySQL / Линн Бейли, Майкл Моррисон ; [пер. с англ.]. — М. : Эксмо, 2010. — 800 с. : ил. — (Мировой компьютерный бестселлер).

ISBN 978-5-699-44494-6

Вы хотите уметь создавать не только статичные, но и динамичные, связанные с базами данных сайты? Тогда вам не обойтись без знания PHP и MySQL. Эта книга является уникальным визуальным руководством, благодаря которому вы усвоите данные технологии максимально эффективно. Вы не только изучите теорию, но и наберетесь практического опыта, создав целый ряд приближенных к реальным проектов (от рейтинговой системы до сайта знакомств). Вы освоите в деле все важнейшие концепции программирования на PHP и под MySQL: верификацию форм, работу с сессиями, эффективные запросы к базе данных, операции с файлами и многое другое.

УДК 004.45
ББК 32.973.26-018.2

Все названия программных продуктов являются зарегистрированными торговыми марками соответствующих фирм.

Никакая часть настоящего издания ни в каких целях не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитный носитель, если на это нет письменного разрешения ООО «Издательство «Эксмо».

Authorized Russian translation of Head First PHP & MySQL ISBN 9780596006303 © 2008, Lynn Beighley, Michael Morrison. This translation is published and sold by permission of O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

© ЧП «Айдиономикс», перевод на русский язык, 2010

© Издание на русском языке, оформление. ООО «Издательство «Эксмо», 2010

ISBN 978-5-699-44494-6

Оглавление (кратко)

Оглавление (подробно)

Вступление

Ваш мозг осмысливает PHP и MySQL. В то время, когда вы стараетесь изучить что-то, ваш мозг пытается оказать вам услугу, убеждая вас, что все это не имеет значения. Ваш мозг думает: «Надо лучше сосредоточиться на том, как избежать встречи со свирепым хищником, или подумать, действительно ли подводная йога такая полезная вещь». Так как же убедить мозг в том, что ваша жизнь зависит от знания PHP и MySQL?

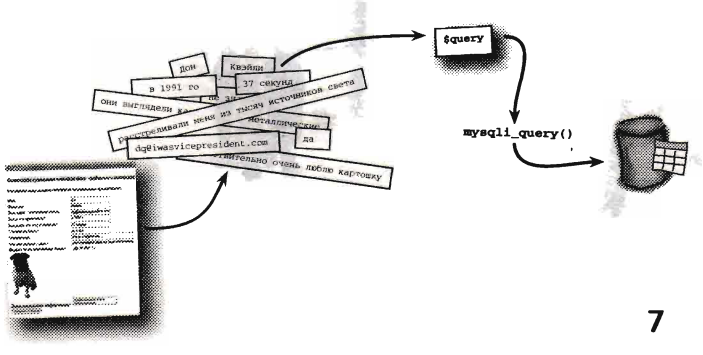
1 Она жива

Вы создали замечательную веб-страницу, используя HTML, и украсили ее с помощью CSS, но заметили, что внешний вид вашего сайта не идет дальше того, чтобы пассивно показывать свое содержимое. Информация передается только в одну сторону, и вам хотелось бы изменить такое положение вещей. По сути, вам хотелось бы знать, о чем думает ваша аудитория. Но для этого вы должны предоставить посетителям возможность вносить данные в веб-форму. Вам также необходимо, чтобы эта информация была обработана и передана вам. Похоже, вам нужно больше, чем просто HTML, чтобы поднять ваш сайт на уровень выше.

соединение с MySQL

2 Как все это согласуется между собой

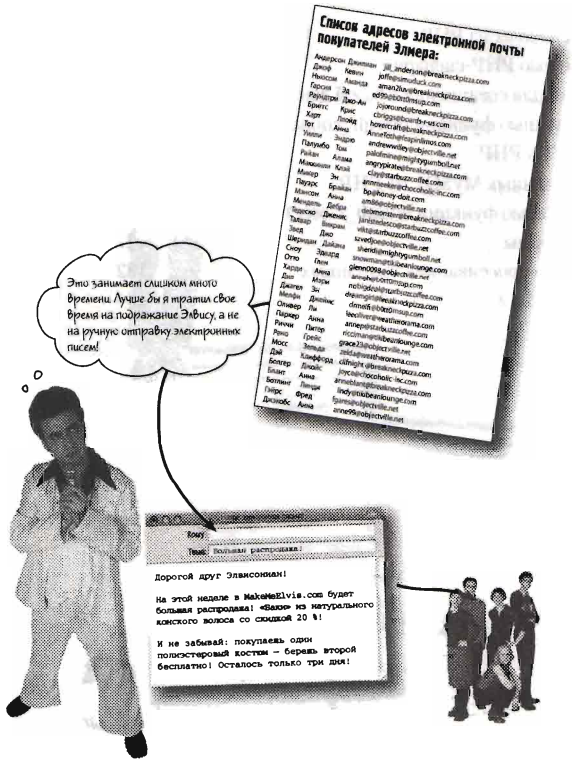
Перед тем как начинать создание системы, очень важно понимать, как отдельные ее элементы согласуются между собой. Вы создали свой первый PHP-сценарий, и он хорошо работает. Но получение результатов обработки вашей формы больше вас устраивать не может. Вам необходим способ, позволяющий сохранять эти результаты так долго, как это вам необходимо. С другой стороны, этот способ должен позволять вам обращаться к данным по мере необходимости. Хорошо известная система управления базами данных MySQL может надежно хранить ваши данные. Но вам необходимо как-то подключить ваш сценарий к этой системе, чтобы воспользоваться теми возможностями, которые она предоставляет.



создание и заполнение базы данных

3 Создание ваших собственных данных

У вас не всегда будут необходимые вам данные. Иногда вам придется создавать данные, прежде чем вы сможете их использовать. А иногда вам придется создавать таблицы, чтобы хранить эти данные. А иногда вам придется создавать базу данных, чтобы хранить данные, которые вам нужно создавать, прежде чем вы сможете их использовать. Запутались? Разберетесь! Приготовьтесь узнать, как создавать свои собственные базы данных и таблицы. И после всего этого вы создадите свое первое PHP- и MySQL-приложение.



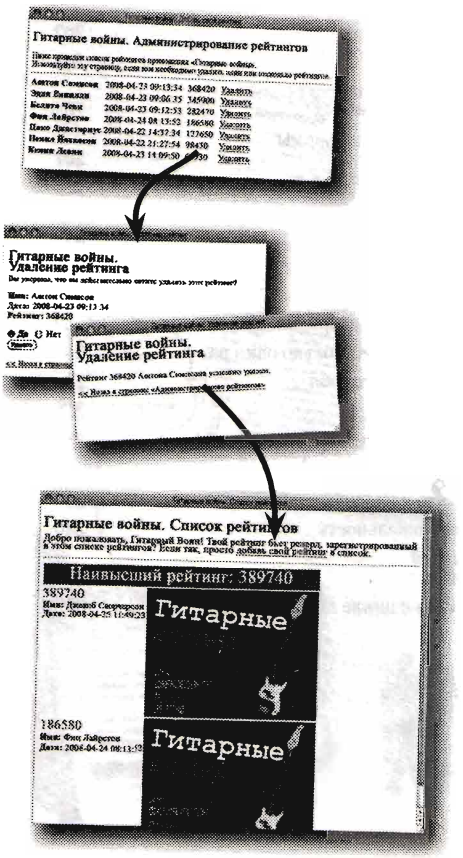
реальные и практические приложения

4 Ваше веб-приложение

Иногда надо реально смотреть на вещи и менять свои планы. Или планировать более внимательно с самого начала. После загрузки приложения на веб-сервер вы, возможно, обнаружите, что спланировали его недостаточно хорошо. Некоторые решения, которые, как вы предполагали, будут работать, в реальной жизни оказались не слишком удачными. В этой главе мы рассмотрим некоторые проблемы реальной жизни, которые могут возникнуть, после того как ваше приложение выйдет из режима тестирования и переместится в Интернет. По пути мы познакомим вас с примерами более важного кода PHP и SQL.

5 Когда просто базы данных недостаточно

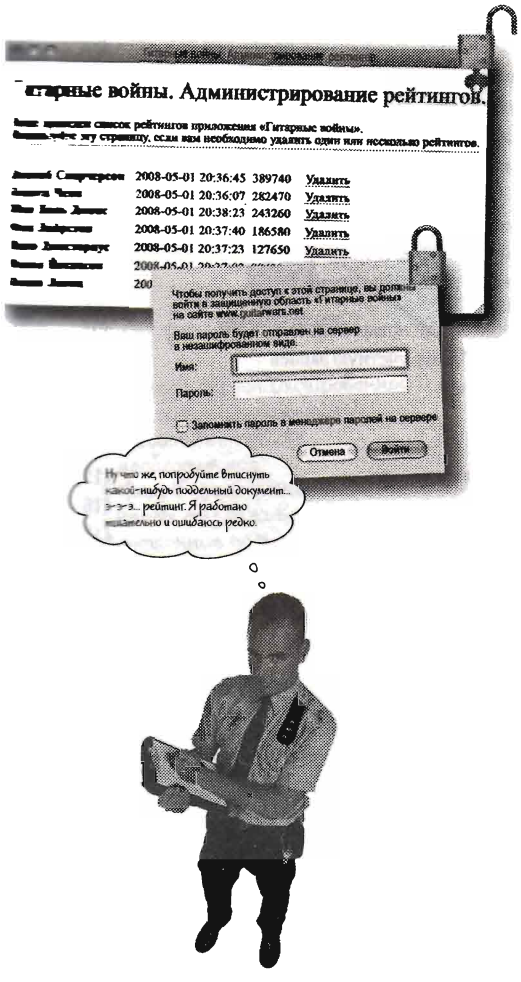
Не верьте хвалебным речам... по поводу баз данных. Конечно, они исключительно удобны для сохранения различных видов данных, включая текст, но как обстоят дела с бинарными данными? Скажем, такими как изображения в формате JPEG или документы в формате PDF. Есть ли смысл в том, чтобы хранить, например, картинки, изображающие коллекции редких гитарных медиаторов, в таблице базы данных? Чаще всего нет. Данные подобного типа обычно сохраняются в виде файлов, и будет лучше, если мы оставим их в этих файлах. В этой главе рассказывается, как совместно использовать данные, сохраненные в файлах и базах данных, для того чтобы создавать PHP-приложения, забитые под завязку бинарными данными.



защита вашего приложения

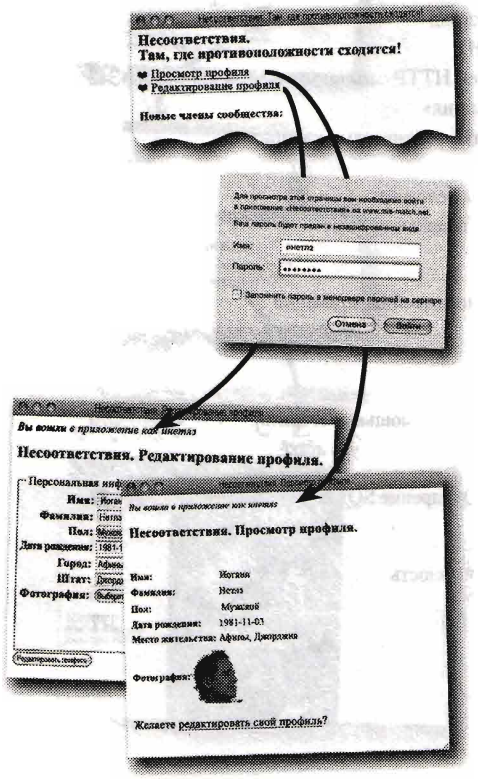
6 Считайте, что все они намерены воспользоваться вашими слабостями

Ваши родители были правы: никогда не разговаривайте с незнакомыми людьми. Или, по крайней мере, не доверяйте им. Ну и, конечно же, не давайте им ключи доступа к данным вашего приложения, полагая, что они будут делать то, что положено. Мы живем в жестоком мире, и вы не можете рассчитывать на то, что все окружающие заслуживают доверия. По сути, будучи разработчиком веб-приложения, вы должны быть частично циником, частично конспиратором-теоретиком. Да, люди в своем большинстве плохие, и все они определенно намерены воспользоваться вашими слабостями! Ладно, возможно, такой взгляд граничит с крайностью, но очень важно серьезно относиться к вопросам защиты своих приложений и разрабатывать их так, чтобы они могли противостоять любому, кто попытается нанести им вред.



7 Вы меня помните?

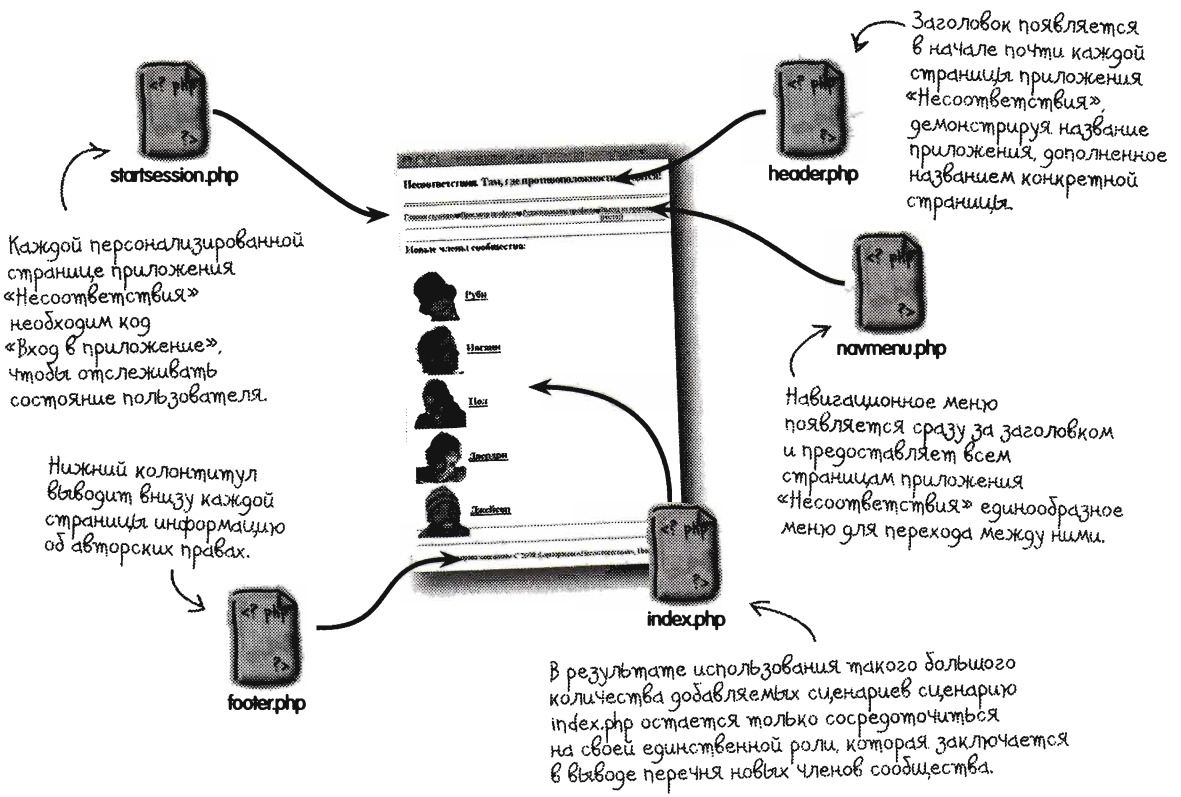
Никому не нравится, когда о нем забывают, особенно это не любят пользователи веб-приложений. Если приложение ориентировано на работу с пользователями как с членами своего рода сообщества и пользователи обмениваются с ним информацией в определенной степени личного характера, такое приложение должно помнить своих пользователей. Вам определенно не понравилось бы представляться своей семье каждый раз, появляясь на пороге дома. Вам нет необходимости делать это, потому что у всех членов вашей семьи есть такая замечательная штука, как память. Но веб-приложение не запоминает людей автоматически. Это забота толкового разработчика приложения, использующего доступные средства (может быть, PHP и MySQL?), — создавать персонализированные веб-приложения, способные запоминать пользователей.



устранение дублирования кода

7 1/2 Совместное использование значит забота

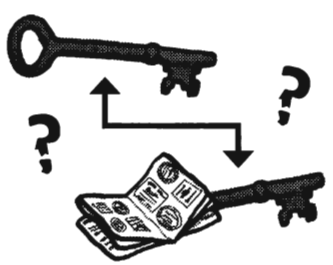
Зонтик — не единственная вещь, которой можно пользоваться совместно. В любом веб-приложении вы можете столкнуться с ситуацией, когда один и тот же фрагмент кода повторяется более чем в одном месте. И это не только неэкономно: это лишняя головная боль при технической поддержке приложения, так как в тех неизбежных ситуациях, когда вам придется вносить изменения, вы должны будете проследить, чтобы эти изменения были внесены во все места, где у вас размещен повторяющийся код. Решение этой проблемы — исключение дублирования кода путем его совместного использования. Говоря другими словами, вы размещаете в каком-то одном месте код, необходимый в нескольких местах, а затем просто ссылаетесь на него, когда у вас возникает в этом необходимость.

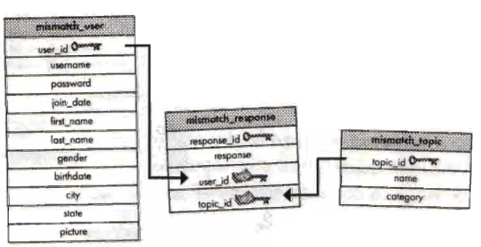


управляйте своими данными, управляйте окружающим вас миром

8 Сбор урожая данных

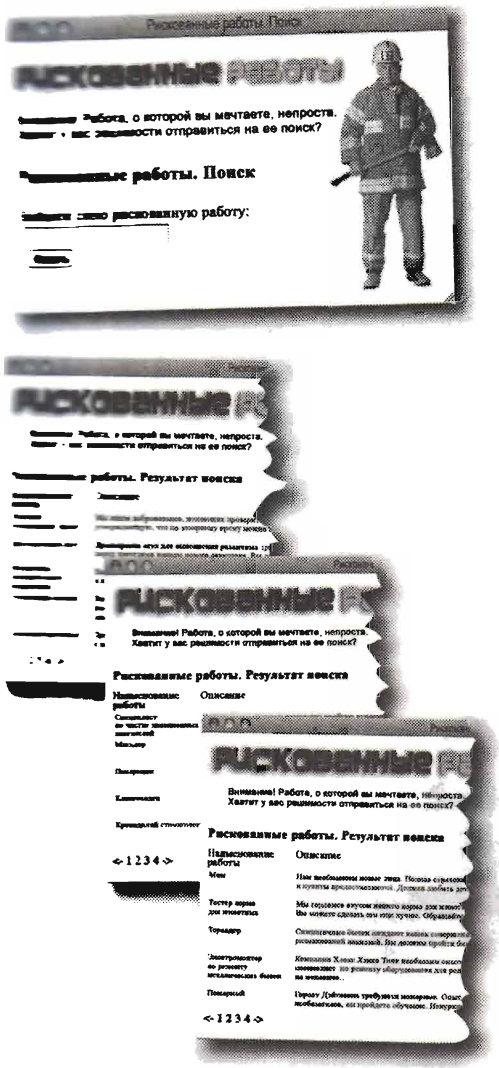
Это не что иное, как осенний сбор урожая данных. Перед вами огромное количество информации, которую нужно получить, отсортировать, сравнить, скомбинировать и вообще сделать все то, что необходимо для вашего потрясающего веб-приложения. Это выполнимо? Да. Но, как и при настоящем сборе урожая, управление данными в базе MySQL требует больших усилий и достаточно глубоких профессиональных знаний и опыта. Пользователь веб-приложения требует большего, чем устаревшие, унылые, непривлекательные данные. Он ожидает данных, которые представляют для него ценность… данных, которые призывают к действию… данных, которые обогащают. Поэтому чего вы ждете? Заводите свой MySQL-комбайн и принимайтесь за работу!





9 Функции облегчают жизнь

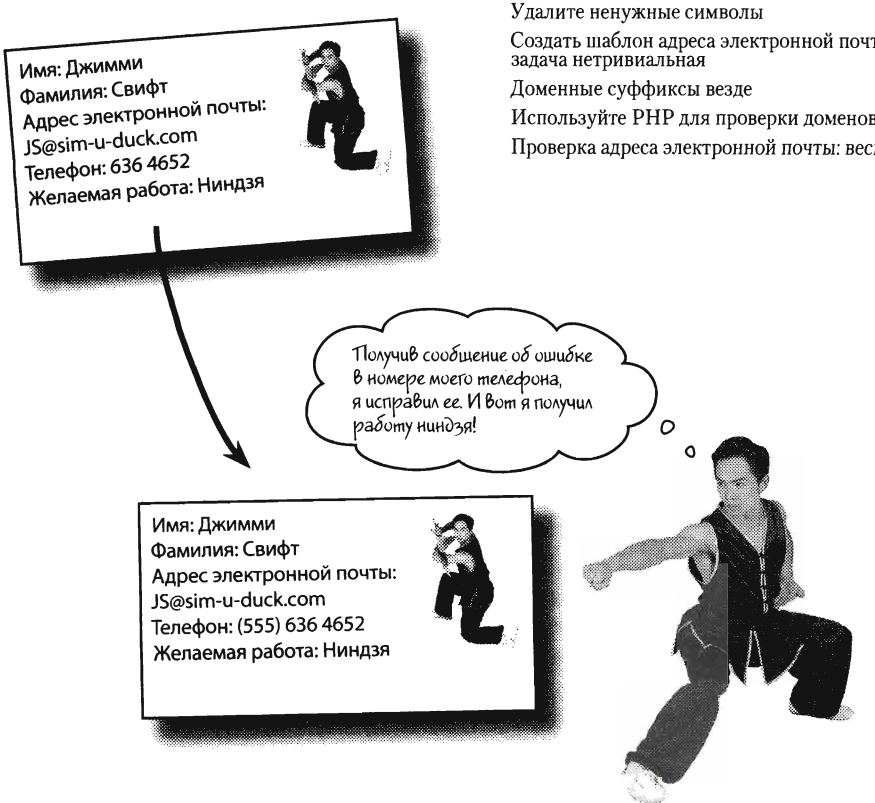
Функции переводят ваше приложение на совершенно новый уровень. Вы уже использовали встроенные PHP-функции для решения определенных задач. Наступило время рассмотреть еще несколько исключительно полезных встроенных функций. А затем вы научитесь создавать свои собственные пользовательские функции для того, чтобы продвинуться до такого уровня, что даже сложно себе представить, что это вообще возможно. Ладно, может, не до такого, на котором вы могли бы разводить лазерных акул, но, по крайней мере, пользовательские функции помогут оптимизировать ваш код и сделать его пригодным для многократного использования.



10 Регулярные выражения

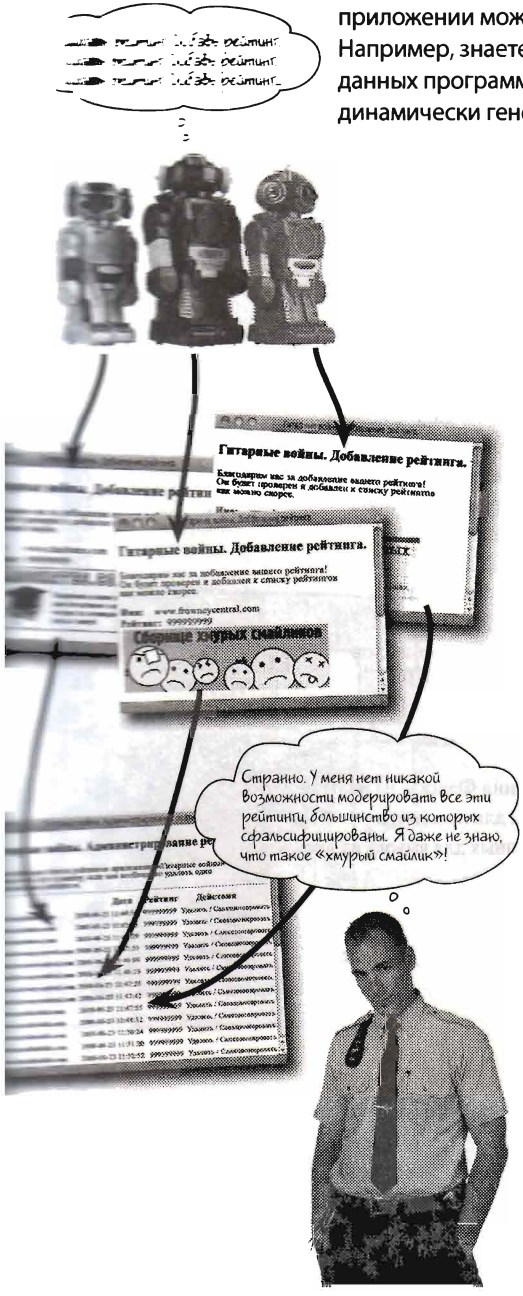
Правила замены

Функции для обработки строк — похоже, очень неплохое средство, но в то же время они достаточно ограничены в своих возможностях. Конечно, они могут определить длину вашей строки, укоротить ее и заменить отдельные символы другими. Но иногда у вас возникает необходимость делать более сложные преобразования текстовых строк. Здесь вам на помощь приходят регулярные выражения. Используя их, вы можете выполнять достаточно сложные преобразования текстовых строк, основываясь на группе правил, а не на каком-то одном критерии.



11 Динамическое создание изображений

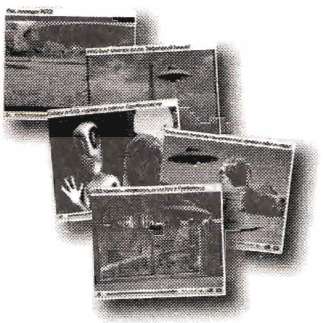
Конечно, все мы знаем широкие возможности грамотно составленного запроса, предоставляющего великолепные результаты. Но результаты запросов не всегда достаточно удобны для восприятия. Иногда полезно показать данные в другом, более наглядном свете. PHP дает возможность представлять данные вашей базы в различных графических видах: в виде секторной диаграммы, столбчатой гистограммы, диаграммах Венна, тестах Роршаха… Вы можете продолжить этот перечень. Все, что дает пользователю возможность воспринимать данные в процессе работы вашего приложения, может рассматриваться как игра. Но в качестве источника данных для построения изображений в вашем приложении может выступать не только информация, сохраненная в базе данных. Например, знаете ли вы, что вполне возможно защитить форму от занесения в нее данных программой-роботом по распространению спама (спам-ботом) с помощью динамически генерируемых изображений?

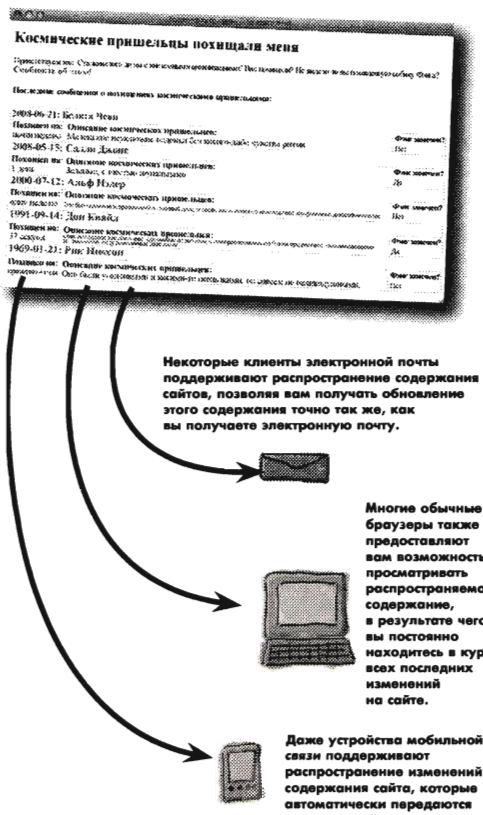


12 Распространение информации и веб-сервисы

Связь с миром

Вокруг нас огромный мир, и наше веб-приложение не может позволить себе игнорировать его. Хотя, может быть, правильнее сказать, для вас было бы лучше, если окружающий вас мир не игнорировал бы ваше веб-приложение. Один из исключительно эффективных способов привлечь внимание мира к вашему веб-приложению — это сделать ваши данные доступными для распространения, что означает предоставить пользователям возможность получать на свой компьютер новости вашего сайта, вместо того чтобы они посещали его непосредственно в надежде узнать эти новости. И не только это. Ваше приложение может взаимодействовать с другими приложениями через различные веб-сервисы и использовать информацию этих приложений.

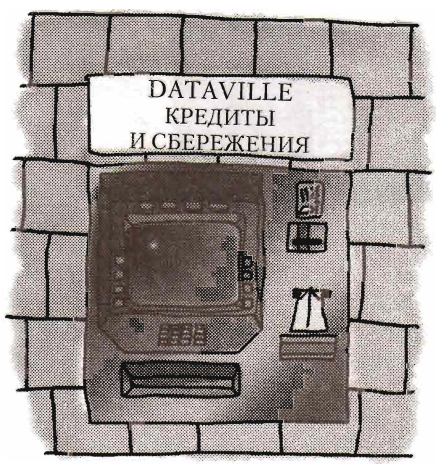




дополнения

I Десять основных тем (которые мы не затронули)

Даже после всего изложенного есть еще некоторые вещи, которые, как мы думаем, вам нужно знать. Мы были бы не правы, игнорируя их, хотя они и требуют только краткого упоминания. Поэтому перед тем, как отложить книгу в сторону, обратите внимание на эти краткие, но важные детали, касающиеся PHP и MySQL. Кроме того, раз уже вы дочитали до этого места, все, что вам осталось, — это несколько коротких дополнений… и алфавитный указатель… и, возможно, немного рекламы… и вот тогда книга действительно закончится.



Настройка среды разработки

II Место, где разворачиваются события

Вам необходимо место, где вы могли бы практиковаться в PHP и MySQL и при этом не допустить уязвимости ваших данных. Очень полезно найти безопасное место для разработки своих PHP-приложений, прежде чем предоставлять к ним доступ из Интернета. Приложение II содержит инструкции, как установить веб-, MySQL- или PHP-сервер для создания безопасного места для работы.

III расширьте возможности своего PHP

Добейтесь еще большего

Да, вы можете создавать отличные приложения с использованием PHP и MySQL. Но вы знаете, что должно быть что-то, что позволит делать это еще лучше. В этом коротком приложении мы покажем вам, как установить такие расширения языка, как mysqli и графическая библиотека GD. Затем мы упомянем еще несколько расширений PHP, которые вас, возможно, также заинтересуют. Потому что иногда хорошо хотеть большего.

Моим родителям, которые часто пользуются веб-приложениями, и всегда из-за меня.

Линн Бейли

Расмусу Лердофу, в одиночку создавшему язык, который в конечном счете превратился в то, что мы знаем теперь как PHP. Неутомимая энергия — вот что позволило этому человеку повести нас по новому, более плодотворному пути.

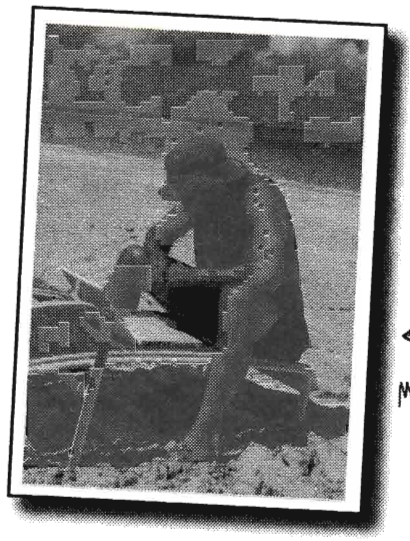
Майкл Моррисон

Автор(ы) книги «Изучаем PHP и MySQL»

Линн Бейли

Линн Бейли — писатель, сменивший область художественной литературы на сферу технической. Придя к выводу, что создание технической литературы приносит вполне реальный доход, она приняла и полюбила это занятие. После получения степени магистра вычислительной техники Линн работала под псевдонимами NRL и LANL. Затем создала свой первый бестселлер. Она переехала в Силиконовую долину как раз перед кризисом. Проработала несколько лет в Yahoo! и написала несколько книг и учебных пособий. Наконец, уступив своим творческим наклонностям писателя, она переехала в Нью-Йорк, где получила степень магистра изящных искусств как писатель. Ее магистерская диссертация Head First была очень хорошо принята преподавателями и студентами, и, получив степень, девушка закончила книгу Head First SQL и почти закончила книгу «Изучаем PHP и MySQL». Вот как!

Линн любит путешествовать, писать и составлять исключительно обоснованные истории о совершенно незнакомых людях. Она немного боится НЛО.

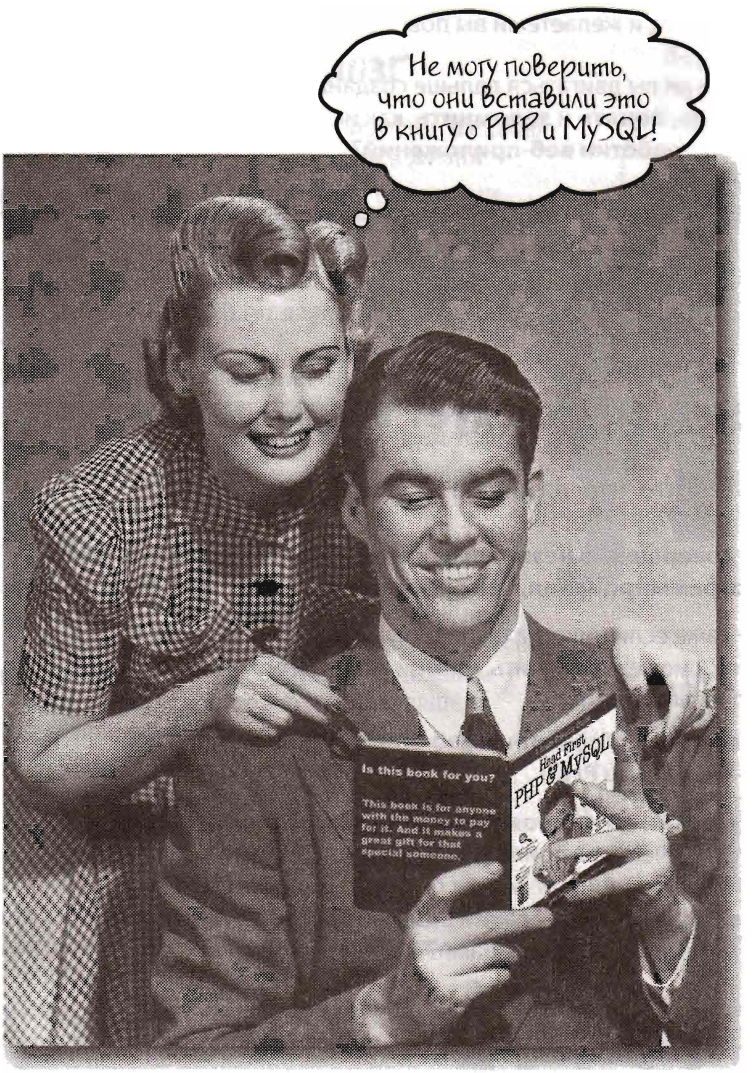


Майкл Моррисон

Майкл Моррисон стал большим энтузиастом Интернета, с тех пор как вел BBS на своем «Коммодоре 64» в те давние времена, когда быть компьютерным фанатиком еще не было так круто, как в наше компьютеризированное время. Сейчас, когда скорости передачи информации возросли на порядки, он не устает восхищаться тем, как быстро и как далеко мы продвинулись в этой области. Майкл больше не ведет BBS, но продолжает с энтузиазмом работать с современными эквивалентами этой системы. Большую часть своего «официального» времени он проводит за разработкой различных веб-технологий. Он создал в качестве автора или соавтора более пятидесяти книг на темы, простирающиеся от программирования игр для мобильных телефонов до XML. Он попал в мир серии книг Head First начиная с книги Head First JavaScript и остается в этом мире до сих пор.

Майкл является также основателем Stailfish Labs (`www.stalefishlabs.com`) — компании, специализирующейся на играх и интерактивном медиа. А еще хорошо известно, что у него есть занятия вне Интернета (не удивляйтесь!). Скейтборд, хоккей... Он любит также посидеть возле бассейна с карпами вместе со своей женой Машид. И даже время от времени ложится поспать.

Вступление



В этом разделе мы ответим на животрепещущий вопрос: «Так почему же они ВСТАВИЛИ это в книгу о PHP и MySQL?»

Для кого эта книга?

Если вы ответите «да» на все эти вопросы:

- Являетесь ли вы веб-разработчиком с опытом использования HTML или XHTML и желаете ли вы повысить уровень ваших страниц?
- Хотите ли вы двигаться дальше создания простых HTML-страниц, **изучить**, **понять** и **запомнить**, как использовать **PHP и MySQL для разработки веб-приложений**?
- Предпочитаете ли вы **стиль непринужденной беседы за обеденным столом стилю сухой и скучной академической лекции**?

тогда эта книга для вас.

Кому, скорее всего, следует отложить эту книгу в сторону?

Если вы ответите «да» на любой из этих вопросов:

- **Вы совершенно незнакомы с такими базовыми понятиями программирования, как переменные и циклы?**

  (Но даже если никогда не программировали до этого, то, скорее всего, сможете получить из этой книги необходимые вам ключевые понятия.)
- Вы опытный разработчик веб-приложений, ищущий справочник **по PHP**?
- **Боитесь** ли вы **пробовать что-нибудь отличающееся** от того, к чему вы привыкли? Считаете ли вы, что техническая книга не может быть серьезной, если в ней идет речь о создании базы данных о похищениях космическими пришельцами?

тогда эта книга не для вас.

(Примечание отдела маркетинга: эта книга только для обладателей кредитных карточек.)

Мы знаем, о чем вы думаете

«Насколько *эта* книга о PHP и MySQL серьезна?»

«Зачем вся эта графика?»

«Смогу ли я *изучить* материл, подаваемый в таком виде?»

Мы знаем, о чем думает ваш мозг

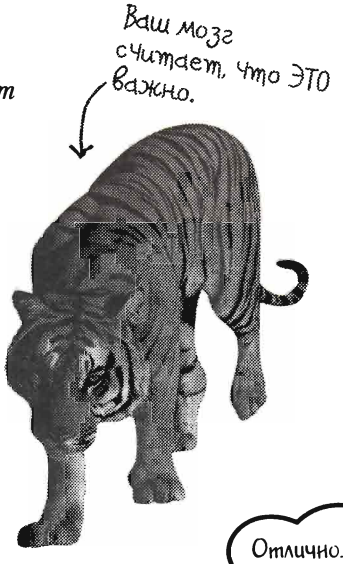
Ваш мозг жаждет нового. Он все время что-то ищет, изучает, *ожидает* чего-то необычного. Он так устроен, и это помогает вам в жизни.

Итак, что делает ваш мозг со всеми этими рутинными, обычными, нормальными вещами, с которыми вы сталкиваетесь? Все, что он может сделать, чтобы предотвратить их влияние на свою *важную* работу, — это сохранить то, что *имеет значение*. Он не утруждает себя сохранением скучных вещей, они не идут дальше фильтра под названием «это не имеет никакого значения».

Как же ваш мозг *определяет*, что имеет значение? Предположим, вы отправились на прогулку и неожиданно прямо перед вами выпрыгнул тигр. Что произойдет в вашем мозгу, да и во всем теле?

Нейроны вспыхнут. Чувства обострятся до предела. *Концентрация адреналина в крови начнет зашкаливать.*

Вот так ваш мозг и определяет...



Это очень важно! Не забудьте это!

А теперь представьте, что вы дома или в библиотеке. В этой зоне, свободной от всяких там тигров, безопасно и тепло. Вы занимаетесь. Готовитесь к экзаменам. Или пытаетесь разобраться со сложным техническим вопросом, на решение которого, по мнению вашего начальника, вам нужна неделя или, от силы, десять дней.

Вот только одна проблема. Ваш мозг пытается оказать вам большую услугу. Он старается принять все меры, чтобы все это не имеющее никакого значения содержание не забивало и без того небезграничные ресурсы. Ресурсы, которые необходимы для сохранения действительно важных вещей. Таких как тигры, как угроза пожара, как быстро спрятать окно браузера с видеозаписью приземления космических пришельцев, если вдруг появится ваш начальник.

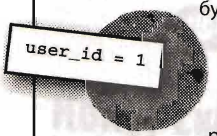
И не существует просто способа сказать вашему мозгу: «Эй, мозг, спасибо тебе, конечно, но независимо от того, насколько эта книга скучна и насколько низок сейчас мой эмоциональный уровень по шкале Рихтера, мне крайне необходимо, чтобы ты сохранил всю эту информацию».

Мы рассматриваем читателя серии книг Head First как обучающегося

Тогда что же необходимо для того, чтобы научиться чему-нибудь? Для начала материал должен *дойти* до вас, затем вам нужно постараться *не забыть* его. Недостаточно просто «набить» вашу голову фактами. Согласно последним исследованиям в области психологии обучения, теории познания и неврологии, *обучение* включает значительно больше, чем просто чтение текста. Мы знаем, как мобилизовать ваш мозг.

Вот некоторые принципы обучения, реализованные в серии Head First.

Используйте изображения. Изображения запоминаются значительно лучше простых слов и делают процесс обучения более эффективным (повышение эффективности тестов на запоминание до 89 %). Кроме того, изображения более понятны. **Помещайте слова внутри или около изображений**, к которым они относятся, а не внизу страницы или на другой странице, и обучаемому будет значительно проще разобраться в проблеме.



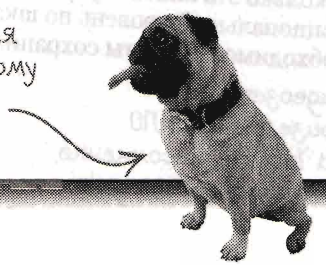
Используйте разговорный и персонализированный стиль. Как следует из последних исследований, улучшение результатов контрольных тестов учащихся достигает 40 %, если книга обращается непосредственно к читателю, материал подается в неформальном, разговорном стиле. Вместо чтения лекций рассказывайте истории. Говорите неофициальным языком. Не держитесь слишком серьезно. Что больше привлечет ваше внимание: непринужденная беседа за столом или сухая лекция?

Создавайте все условия для более глубокого осмысления материала обучаемым. Иначе говоря, пока вы активно не возбудите его нейроны, ничего не произойдет в его голове. Читатель должен быть взволнован, увлечен, удивлен и заинтересован в решении проблемы. Он должен делать выводы и получать новые знания. А чтобы достичь этого, вам необходимы задачи, упражнения, провокационные вопросы — все, что вовлекает в работу обе половины мозга и множество чувств.

Привлекайте и удерживайте внимание читателя. Все мы сталкиваемся с ситуацией «я очень хочу разобраться во всем этом, но не могу побороть сон после первой страницы». Ваш мозг обращает внимание на неординарные, интересные вещи. Вещи, которые попадают в поле зрения совершенно неожиданно. Новые технические темы совсем необязательно должны подаваться в скучном формальном стиле. Ваш мозг воспримет гораздо больше и быстрее, если это не так.

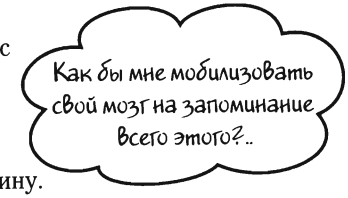
Затрагивайте эмоции. Мы знаем, что способность запоминать что-либо в значительной степени зависит от эмоционального состояния. Вы запоминаете то, что для вас важно. Вы запоминаете тогда, когда ваши чувства обострены. Мы не имеем в виду душещипательные истории о мальчике и его собаке. Мы говорим о таких чувствах, как удивление, любопытство, чувствах, вызывающих смех или вопрос: «Что это?..», чувстве глубокого удовлетворения, возникающем после решения сложной головоломки или осознания того, что многие считают слишком трудным для понимания. Что вы узнали то, чего не знает Боб из инженерного отдела, постоянно дающий вам понять: «Я технически грамотнее, чем ты».

Одно небольшое уточнение. На самом деле в этой книге есть душещипательная история о парне и его собаке. Вы будете помогать этому парню в поиске его питомца, похищенного космическими пришельцами!



Метапознание: мысли о мыслях

Если вы действительно хотите изучить что-либо и стремитесь сделать этот процесс быстрым и более глубоким, обратите внимание на то, как вы обращаете внимание на что-то. Подумайте о том, как вы думаете. Изучите то, как вы изучаете.



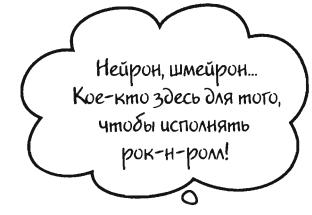
Большинство из нас в течение своей жизни не изучают метапознание, или теорию обучения. Считается, что мы это и так *знаем*, и редко когда *изучаем* это как дисциплину.

Но мы уверены, что если вы держите эту книгу в руках, то действительно желаете изучить, как с помощью PHP и MySQL создать сайт, управляемый базой данных. И вам, скорее всего, не хотелось бы тратить на это много времени. Если вы планируете использовать то, что прочитали в этой книге, вам необходимо *запомнить* все, что вы прочитали. А для этого нужно *понять* все это. Для того чтобы получить максимум от прочтения этой, любой другой книги или от обучения, необходимо управлять своим мозгом. Направить свой мозг на изучаемое содержание.

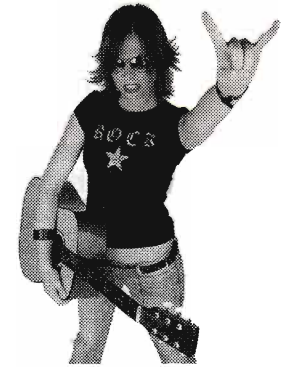
Фокус заключается в том, чтобы дать мозгу понять, что материал, который вы изучаете, действительно важен. Что он критичен для вашего существования. Имеет такое же значение, как тигр на вашем пути. В противном случае вы вынуждены будете постоянно вести борьбу со своим мозгом, прилагающим все усилия для предотвращения сохранения этой информации.

Так как, сможете вы убедить свой мозг рассматривать PHP и MySQL так же, как голодного тигра, перегородившего вам дорогу?

Имеется два пути: один медленный и нудный, другой значительно более быстрый и эффективный. Медленный путь заключается в простой зубрежке. Вы, очевидно, знаете, что можете заучить и запомнить наиглупейшее содержание, если будете долго и монотонно вбивать его в свой мозг. После достаточно длительного повторения мозг скажет себе: «Это не выглядит имеющим для него значение, но он продолжает и продолжает это повторять. Похоже, это все же важно».



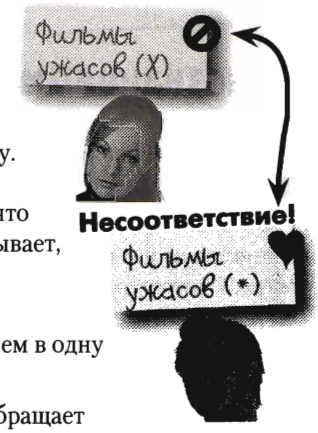
Быстрый путь использует ***все, что повышает активность мозга,*** особенно различные *виды* его активности. Перечисленное на предыдущей странице составляет значительную часть решения. Использование этих приемов поможет организовать работу мозга в соответствии с вашей необходимостью. Например, исследования показывают, что, если поместить описание изображения не где-нибудь на странице отдельно от этого изображения (как, например, заголовок текста), а непосредственно внутри него, это дает мозгу основание считать, что и изображение, и его описание — вещи тесно связанные. Следствием этого является увеличение количества возбужденных нейронов, что в свою очередь приводит к увеличению значимости для мозга рассматриваемого элемента содержания книги.



Разговорный стиль изложения материала помогает потому, что люди придают большее значение информации, когда они обмениваются ею с другими. Удивительно то, что ваш мозг совершенно необязательно рассматривает эту беседу как «беседу» с книгой! С другой стороны, если стиль книги формален и сух, ваш мозг чувствует себя так же, как чувствуете себя вы, сидя на лекции в огромной аудитории, полной пассивных слушателей. Нет особой необходимости бодрствовать.

Но использование изображений и разговорный стиль изложения материала — это только начало...

Вот что мы делаем



Мы используем ***изображения*** потому, что мозг предпочитает изображения тексту. Ваш мозг убежден, что изображение стоит тысячи слов. А когда мы используем изображения вместе с текстом, мы встраиваем этот текст в изображение потому, что мозг работает более эффективно, когда текст находится *внутри* того, что он описывает, а не в заголовке или где-либо еще на странице.

Мы используем ***избыточность***, повторяя одни и те же вещи *разными* способами и с разных *точек зрения*, чтобы увеличить шансы попадания информации более чем в одну область вашего мозга.

Мы подаем понятия и изображения в ***неожиданном*** виде потому, что ваш мозг обращает внимание на все новое. И мы стараемся придать изображениям и понятиям *определенный* ***эмоциональный*** *оттенок* потому, что ваш мозг очень чутко реагирует на биохимию эмоций. По этой причине вы *чувствуете*, что это стоит запомнить, хотя эти чувства — не более чем немного ***юмора***, ***удивления*** или ***любопытства***.

Мы используем персонализированный ***разговорный стиль*** изложения потому, что ваш мозг более восприимчив, когда он считает, что вы участвуете в беседе, чем когда вы пассивно слушаете доклад. Ваш мозг ведет себя так, даже когда вы *читаете*.

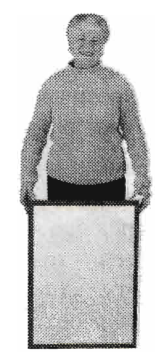
Мы включили более 80 ***практических*** упражнений потому, что ваш мозг лучше воспринимает информацию, когда вы что-то делаете, чем когда вы читаете о том, как это делается. И мы делаем упражнения хотя и сложными, но выполнимыми, потому что это мобилизует людей на их решение.

Мы используем разнообразные способы обучения потому, что вы, возможно, предпочитаете изучать материал шаг за шагом, в то время как кто-то другой хочет вначале понять общую картину, а третьему лучше просто разобрать практический пример. Но, независимо от предпочтений, *любой* человек выиграет от подачи материала разными способами.

Мы включаем информацию для ***обеих половин мозга***, так как чем большую область мозга вы задействуете, тем выше вероятность, что вы лучше поймете и запомните материал, и тем дольше вы сможете удерживать свое внимание. Так как загрузка одной половины мозга часто означает, что другая половина отдыхает, процесс изучения может быть более продуктивным в течение более продолжительного времени.

Тест-драйв

Мы включили ***истории*** и упражнения, отражающие ***более одной точки зрения***, потому что ваш мозг настроен на более глубокое изучение, если он вынужден делать вычисления и суждения.



Мы включили ***проблемные*** задачи и ***вопросы***, которые не всегда имеют прямое и простое решение, потому что ваш мозг настроен на лучшее понимание и запоминание, когда он должен *поработать* над чем-либо. Подумайте об этом. Вы не сможете обрести *фигуру* атлета, *наблюдая* за людьми в тренажерном зале. Поэтому мы приложили все усилия к тому, чтобы вы хорошо потрудились над тем, что вам действительно *необходимо*, чтобы ***вы не потратили зря ни одного дополнительного дендрита***, разбираясь с запутанным примером, с громоздким, перегруженным жаргонными выражениями или, наоборот, слишком лаконичным текстом.

Мы описывали людей в историях, упражнениях, изображениях и т. п. потому, что... ну просто потому, что вы человек. И ваш мозг уделяет больше внимания *людям*, чем *вещам*.

А вот то, что вы можете сделать, чтобы ваш мозг слушался вас

Итак, мы свое дело сделали. Остальное — за вами. Эти советы — начало пути; прислушивайтесь к своему мозгу и старайтесь понять, что он воспринимает хорошо, а что — не очень. Пробуйте разные способы.

Вырежьте это и приклейте на дверцу вашего холодильника.

Не спешите. Чем больше вы будете понимать, тем меньше вам придется запоминать.
Не ограничивайтесь только чтением. Делайте перерывы, чтобы обдумать прочитанное. Если вам встречается вопрос, не смотрите сразу ответ на него. Попробуйте представить, что кто-то действительно задает вам вопрос. Чем лучше вы заставите свой мозг думать, тем больше у вас шансов понять и запомнить материал.

Выполняйте все упражнения. Делайте свои собственные примечания.
Мы вставили примечания, но если вы не будете делать своих, это будет так, как будто кто-то другой делает вашу работу. И не ограничивайтесь простым *чтением* упражнений. **Возьмите в руки карандаш**. Имеется множество свидетельств в пользу того, что физическая активность *в процессе* обучения повышает степень восприимчивости.

Читайте разделы «Не бывает глупых вопросов».
Все они имеют большое значение. Это не просто примечания на полях. ***Это часть основного содержания!*** Не пропускайте эти разделы.

Старайтесь, чтобы чтение книги было последним вашим делом, перед тем как вы ложитесь спать. Или, по крайней мере, последним делом, требующим умственного напряжения.
Часть процесса усвоения материала (особенно передача его в долговременную память) происходит после того, как вы отложили книгу в сторону. Вашему мозгу необходимо какое-то время для обработки полученной информации. Если в течение этого процесса вы дадите ему какие-либо новые данные, процесс прервется и часть информации будет потеряна.

Пейте воду. Много воды!
Вашему мозгу труднее работать, если он испытывает недостаток в жидкости. Обезвоживание организма (о чем он сигнализирует вам возникновением чувства жажды) снижает способность мозга к усвоению информации.

Обсуждайте прочитанное вслух.
Разговор возбуждает разные области мозга. Если вы пытаетесь понять или запомнить что-либо, проговаривайте это вслух. Еще лучше, если вы попытаетесь объяснить это кому-нибудь. В этом случае усвоение материала пойдет значительно быстрее. Более того, не исключено, что вы совершенно неожиданно поймете что-то, о чем и не подозревали, когда просто читали этот материал.

Прислушивайтесь к своему мозгу.
Старайтесь понять, не перегрузился ли ваш мозг. Если вы начинаете замечать, что «скользите» по поверхности или вдруг забываете то, что вы только что прочитали, это верный сигнал, что пришло время сделать перерыв. Вы не сможете воспринять больше того, что способны. И простое механическое добавление материала сверх допустимой нормы не улучшит этот процесс, а вот повредить ему может вполне.

Привлекайте свои чувства.
Вашему мозгу необходимо знать, что это *имеет значение*. Почувствуйте себя участником историй, приведенных в книге. Придумайте новые названия к изображениям. Испытать неприятные чувства от плохой шутки все же лучше, чем не испытывать никаких чувств вообще.

Пишите много кода!
Существует единственный способ научиться программировать — **программировать, и как можно больше**. И это как раз то, что вы будете делать на протяжении чтения всей этой книги. Программирование сродни искусству, и единственный способ преуспеть в этой области заключается в обширной практике. Мы предоставим вам множество практических примеров. В каждой главе есть упражнения, имеющие проблемы в решении. Не ограничивайтесь простым чтением этих упражнений. Вы многого добьетесь в процессе их выполнения. Мы включили решение для каждого из упражнений. Не бойтесь **бросить быстрый взгляд на эти решения**, если у вас возникли затруднения! Часто бывает достаточно зацепиться за что-нибудь незначительное. Но старайтесь решать задания полностью и самостоятельно, прежде чем переходить к следующим разделам книги.

PHP и MySQL дают вам возможность создавать веб-приложения, имеющие практическое значение. Не забывайте загружать их на рабочий веб-сервер и проверять, открывая в браузере.

Read me (Прочти меня)

Эту книгу можно назвать учебным пособием, но справочником она не является. Мы умышленно удалили все стоящее на пути того, что рассматриваем на каждом из этапов развития содержания. И мы хотим подчеркнуть: начинайте сначала, потому что считается, что материал, изложенный в предыдущих главах, вам уже знаком.

Мы начинаем с изучения простых программных решений и основных приемов работы с базами данных. Затем мы рассматриваем более сложные функции PHP и запросы MySQL, а в самом конце переходим к решению наиболее сложных задач.

Хотя очень важно создавать приложения, которые позволяют пользователям добавлять в него данные и извлекать их, прежде чем вы сможете делать это, вам необходимо разобраться с синтаксисом и PHP, и MySQL. Поэтому мы даем вам выражения PHP и MySQL, которые вы можете проверить на практике. Таким образом, вы сразу же сумеете делать что-нибудь практическое, используя PHP и MySQL, и это должно заинтересовать вас. Затем мы покажем вам примеры хорошего стиля программирования приложений и структурирования баз данных. После того как вы получите твердое представление о синтаксисе, мы сфокусируемся на изучении общих вопросов.

Мы не рассматриваем все выражения, функции, ключевые слова и запросы PHP и MySQL.

Хотя мы смогли бы включить все выражения, функции, ключевые слова и запросы PHP и MySQL, мы подумали, что вы предпочтете иметь относительно подъемную книгу, которая научит вас использованию наиболее важных выражений, функций, ключевых слов и запросов. Мы даем те из них, которые вам необходимо знать и которые вы будете использовать 95 % времени. А когда вы закончите изучать эту книгу, сможете вполне уверенно найти любую функцию, необходимую для включения в ваше сногсшибательное приложение, к которому вы приступили.

Мы поддерживаем PHP 5 и MySQL 5.0.

Так как многие продолжают пользоваться PHP 4 и 5, мы избегали применения любого специфического для PHP 4, 5 и 6 кода, где это возможно. Мы рекомендуем вам использовать PHP 5 или 6 и MySQL 5 или 6 при работе с этой книгой. При этом вы должны учитывать, что наш код совместим с более ранними версиями этого программного обеспечения.

Мы можем использовать PHP4 при изучении этой книги, внеся небольшие изменения в приведенный в ней код. Обратитесь к разделу #1 приложения

Вам необходим веб-сервер, поддерживающий PHP.

Код PHP выполняется на веб-сервере. Вам необходимо установить Apache или какой-либо другой веб-сервер на вашем или на любом другом компьютере, к которому у вас есть доступ. Вы также должны установить сервер MySQL или получить доступ к нему. Обратитесь к приложениям II и III за инструкциями по установке и настройке веб-сервера с поддержкой PHP и сервера MySQL.

Мы используем MySQL.

Хотя существует стандартный SQL (Structured Query Language — язык структурированных запросов), в этой книге мы сосредоточились на конкретной его реализации для MySQL. С самыми

незначительными изменениями код этой книги может быть использован в запросах к Oracle, MS SQL Server, PostgreSQL, DB2 и другим RDBMS (Relational Database Management System — системы управления реляционными базами данных). Если у вас возникнет необходимость соединения с одной из этих баз данных, вы сможете найти соответствующие функции и описание синтаксиса PHP. Если бы мы поставили себе задачу описать все варианты синтаксиса для работы со всеми базами данных, объем этой книги был бы намного больше. Поэтому мы сосредоточились только на MySQL.

Практические задания НЕ являются дополнительным материалом.

Упражнения и другие практические задания не являются каким-то дополнительным материалом. Это часть содержания книги. Одни из них даны, чтобы улучшить запоминание, другие — чтобы улучшить понимание, третьи помогут применить то, что вы изучили. Не пропускайте практические задания. Единственное, что вы не должны делать, — это разгадывать кроссворды, но их решение даст вашему мозгу возможность хорошо поразмыслить о значениях слов и терминов, которые вы изучаете в разных контекстах.

Избыточность не случайна и важна.

Одна из особенностей серии книг Head First заключается в том, что мы поставили себе задачу, чтобы книга «дошла» до вас. И мы очень надеемся на то, что после прочтения этой книги вы будете помнить все изученное в ней. Большинство справочников не рассматривают в качестве своей цели запоминание изложенного в них материала, но это не справочник — это книга для обучения, поэтому вы увидите, что некоторые понятия возникают на ее страницах чаще одного раза.

Примеры, приведенные в книге, настолько малы по объему, насколько это возможно.

Наши читатели говорят, что их смущает необходимость пробираться через 200 строк кода в поисках двух строк, которые им нужно понять. Большинство примеров в этой книге настолько малы по объему, насколько это возможно для того, чтобы та часть, которую вам необходимо понять, была выражена как можно яснее и проще. Не ждите, что все примеры будут законченными и начнут работать исключительно эффективно. Они приводятся для изучения, а не для непосредственного использования в готовых приложениях.

Некоторые из примеров являются вполне завершенными веб-приложениями, способными выполнять довольно сложные задачи.

Мы поместили коды всех примеров и приложений на сайт, чтобы вы имели возможность скопировать фрагменты из них в файлы своих сценариев, используя текстовый редактор, или выполнить запросы в MySQL-терминале. Вы также можете загрузить их на свой веб-сервер для тестирования. Вы найдете их в Интернете по адресу:

```
http://www.headfirstlabs.com/books/hfphp
```

Упражнения под названием «Сила мысли» оставлены нами без ответа.

Для некоторых из этих упражнений правильного ответа просто не существует, а для других часть процесса обучения заключается в том, чтобы вы сами определили, в каком случае ваш ответ является правильным. В некоторых случаях в этих упражнениях содержатся намеки на правильное направление.

Команда технических рецензентов

Технические рецензенты:

Джереми Ален является веб-разработчиком высокого уровня и обладает опытом создания веб-приложений с использованием самых современных технологий. Более девяти лет он использует в своих разработках PHP, MySQL, а также множество других языков программирования, программных интерфейсов, средств разработки и операционных систем.

Давид Бриггс — автор технической литературы и специалист по локализации программного обеспечения. Живет в Бирмингеме, Англия. Когда он свободен от тщательного, педантичного разъяснения пользователям особенно сложных программистских приемов, для него нет ничего лучше, чем погулять по местному парку вместе с женой Паулеттой и их собакой Клео.

Уил Харрис проводит свой день, управляя отделом информационных технологий, предоставляющим услуги 11 компаниям на 4 континентах. Он является вице-президентом филиала ассоциации PASS (Professional Association for SQL Server — Профессиональная ассоциация пользователей SQL-сервера), Лас-Вегас. Он любит проводить время с женой Хитер, своими прелестными дочками Марой и Элли, а также собакой Свайпером.

Стефани Лиз — инструктор и веб-разработчик в Сакраменто, Калифорния. Когда она не расхваливает достоинства кода, разработанного в соответствии с требованиями стандартов, или не отлаживает компоновку CSS, вы сможете найти ее за занятием йогой.

Если **Стив Милано** не пишет код для компании The Day Job™ или не исполняет панк-рок со своей группой Onion Flavored Rings («Цветущие кольца лука») в каком-нибудь душном подвале, он, скорее всего, проводит время дома со своим ноутбуком, не уделяя никакого внимания кошачьей компании в лице Ральфа и человечьей — в лице Бьянки.

Харви Квимен променял карьеру компьютерного программиста на академическую жизнь с постоянными перелетами в окружении толп папарацци. Он сейчас адъюнкт-профессор университета Альберты, где читает курсы лекций по киберкультуре, литературе XX века и веб-разработке, включая PHP и MySQL.

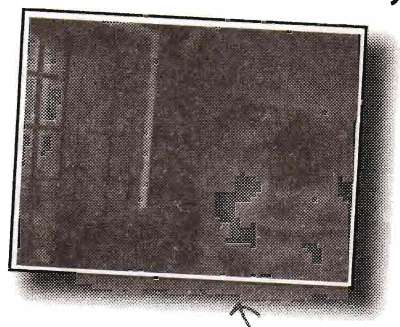
Крис Шифлет — технический директор компании OmniTI, где он является ведущим специалистом в вопросах безопасности. Крис — признанный лидер в области безопасности PHP- и веб-приложений, известный блоггер на сайте `shiftlett.org`, популярный докладчик на различных конференциях и основатель PHP Security Consortium. Он автор книг Essential PHP Security (O'Рейли) и HTTP Developer's Handbook.

Благодарность

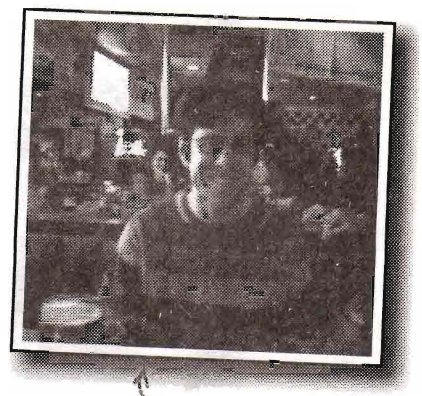
Наши редакторы:

Множество благодарностей **Бретту Маклафлину** за его бесценные архивные материалы, которые направили нас на правильный путь, и за его жесткую приверженность к когнитивному обучению.

Эта книга не увидела бы свет, если бы не героизм, терпение и постоянство **Сандерса Клейнфельда**. Мы исключительно благодарны ему за то, что ему всегда удавалось подхватывать мячи (или это были кошки?), которыми мы жонглировали, при случайном падении одного из них (а то и трех!). Мы надеемся, у него появился шанс положить ноги на стол на пару дней, перед тем как он возьмется за другой проект, не менее сложный, чем этот.



Бретт Маклафлин



Сандерс Клейнфельд

Команда О'Рейли:

Лу Барр

Множество благодарностей **Лу Барр** за ее феноменальные дизайнерские способности, которые придали этой книге такой вид.

Мы благодарим также **Британи Смит** за ее тяжелую работу в самую последнюю минуту и **Катрин Маккаллоф** за установку и настройку сайтов. И **Лори Петрики** за ее веру в то, что мы сможем написать еще одну замечательную книгу из этой серии.

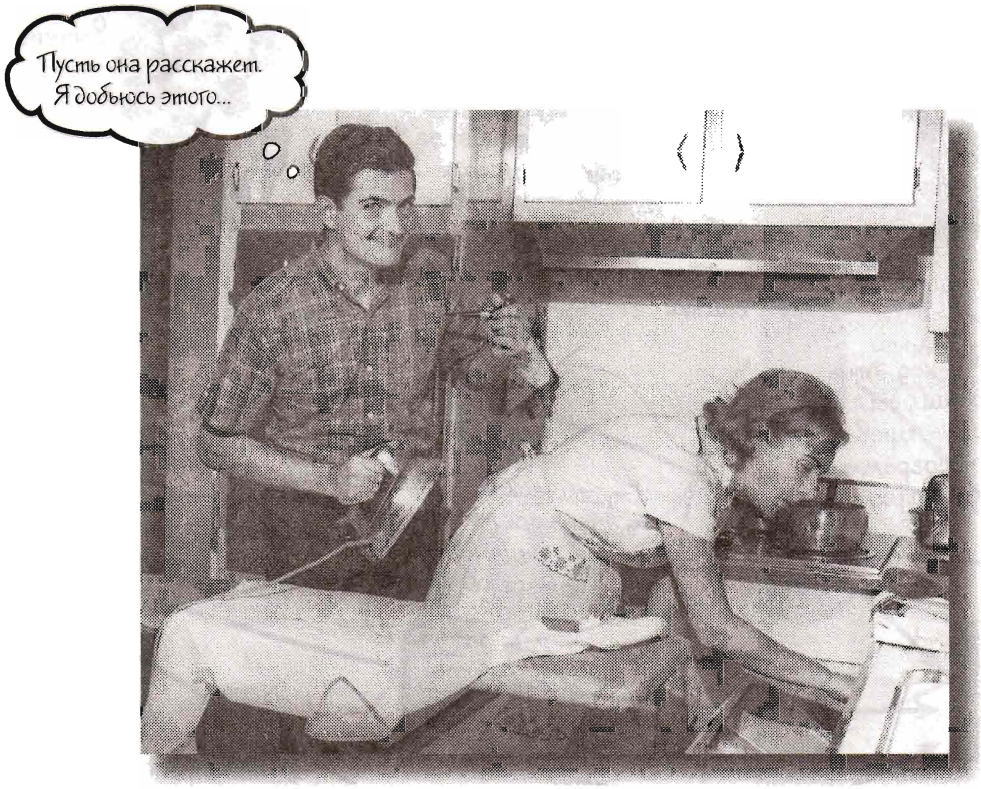
И еще:

Наконец, огромная благодарность **Элвису Уилсону** за сбор и приведение в порядок всех видеозаписей о похищении космическими пришельцами на YouTube для главы 12. Великолепная работа! Особенно если принять во внимание, что он самый обыкновенный, пещерный главный художник.

Элвису Уилсону

1 Вдохните жизнь в ваши статичные страницы

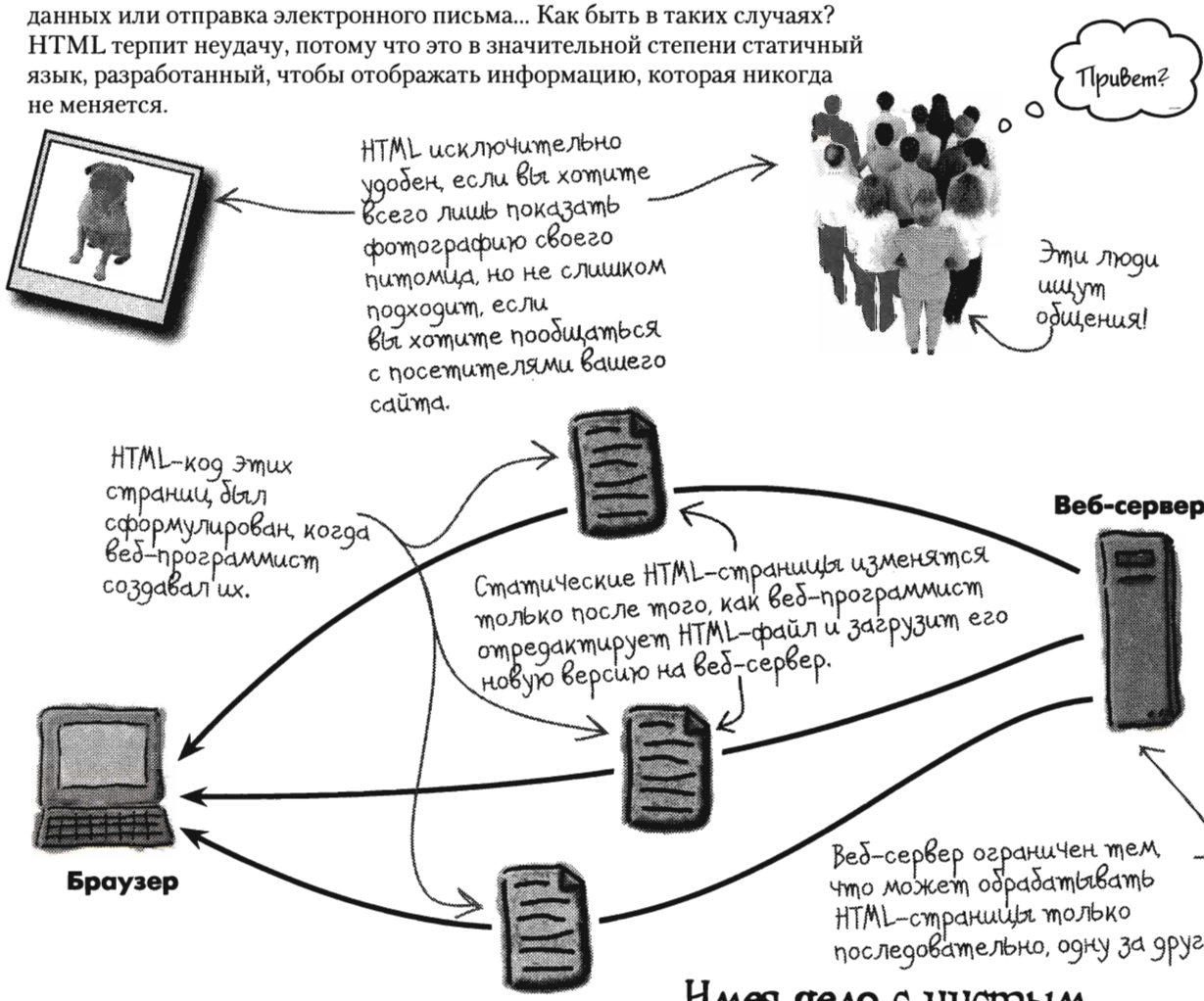
Она жива



Вы создали замечательную веб-страницу, используя HTML, и украсили ее с помощью CSS, но заметили, что внешний вид вашего сайта не идет дальше того, чтобы пассивно показывать свое содержимое. Информация передается только в одну сторону, и вам хотелось бы изменить такое положение вещей. По сути, вам хотелось бы **знать, о чем думает ваша аудитория**. Но для этого вы должны предоставить посетителям возможность **вносить данные в веб-форму**. Вам также необходимо, чтобы **эта информация была обработана и передана вам**. Похоже, вам нужно больше, чем просто HTML, чтобы поднять ваш сайт на уровень выше.

HTML статичен и скучен

Как мы хорошо знаем, HTML исключительно удобен для создания веб-страниц. Но как быть, если нам необходимы не просто веб-страницы, а такие, которые делают что-то? Предположим, вам необходим поиск в базе данных или отправка электронного письма... Как быть в таких случаях? HTML терпит неудачу, потому что это в значительной степени статичный язык, разработанный, чтобы отображать информацию, которая никогда не меняется.



Во многом проблема веб-сервера при обработке статичных HTML-страниц заключается в том, что в этом случае он выступает в роли простого передаточного механизма, и не более того. Браузер запрашивает страницу, сервер предоставляет ее, и на этом все заканчивается. Чтобы превратить сайт в диалоговое веб-приложение, веб-сервер должен выступить в новой, более динамичной роли, которую ему может обеспечить PHP.

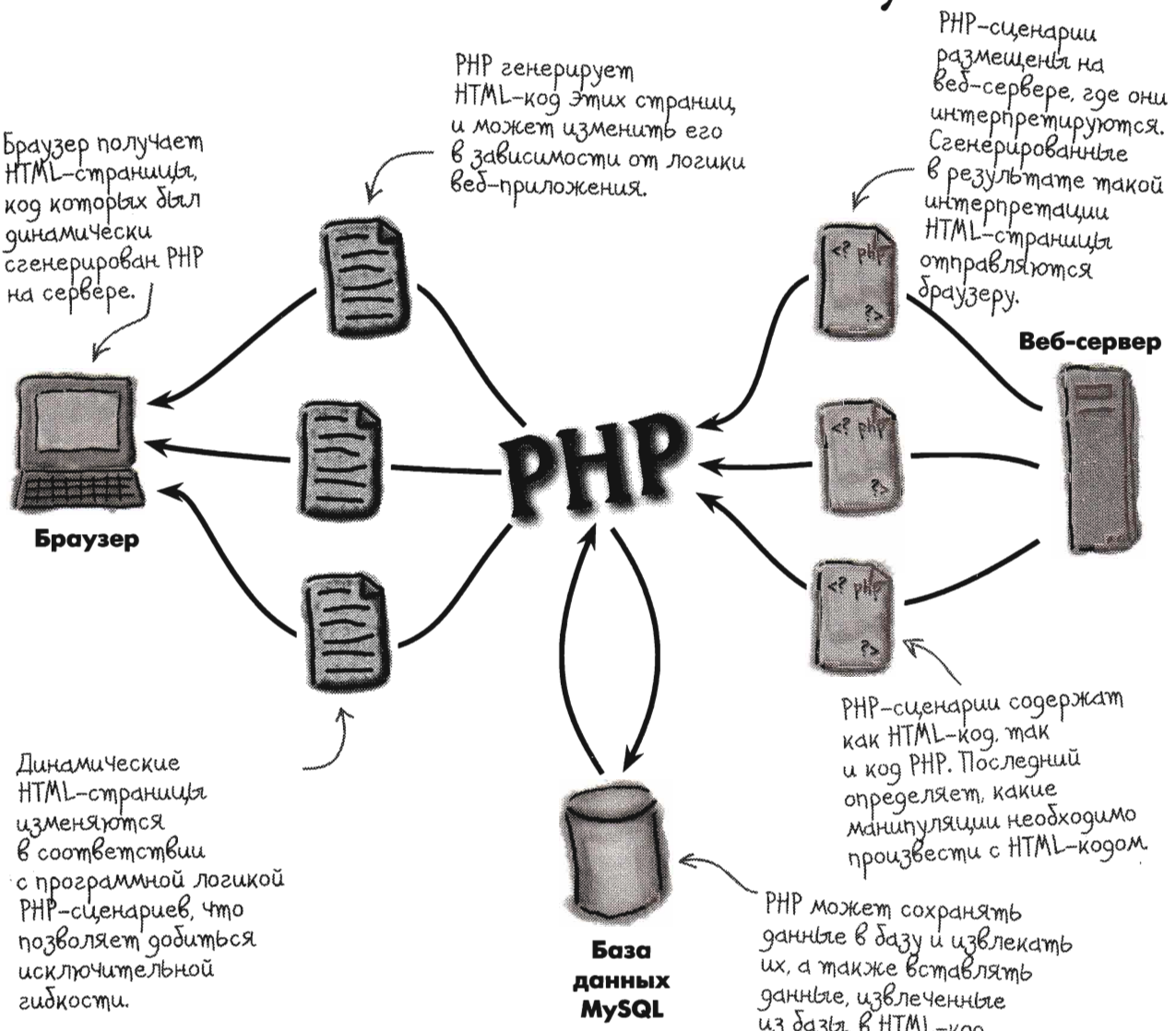
Имея дело с чистым HTML, сервер просто обрабатывает статичную HTML-страницу и, таким образом, только отображает ее содержание.

PHP оживляет веб-страницы

С небольшой помощью сервера.

PHP позволяет вам манипулировать содержанием веб-страницы на сервере непосредственно перед тем, как она будет отправлена браузеру. Это происходит следующим образом: в процессе интерпретации PHP-сценария сервер может изменить или создать любой HTML-код. После этого HTML-страница отправляется браузеру, который не знает и не придает значения тому, что в модификации кода HTML на сервере принимал участие PHP.

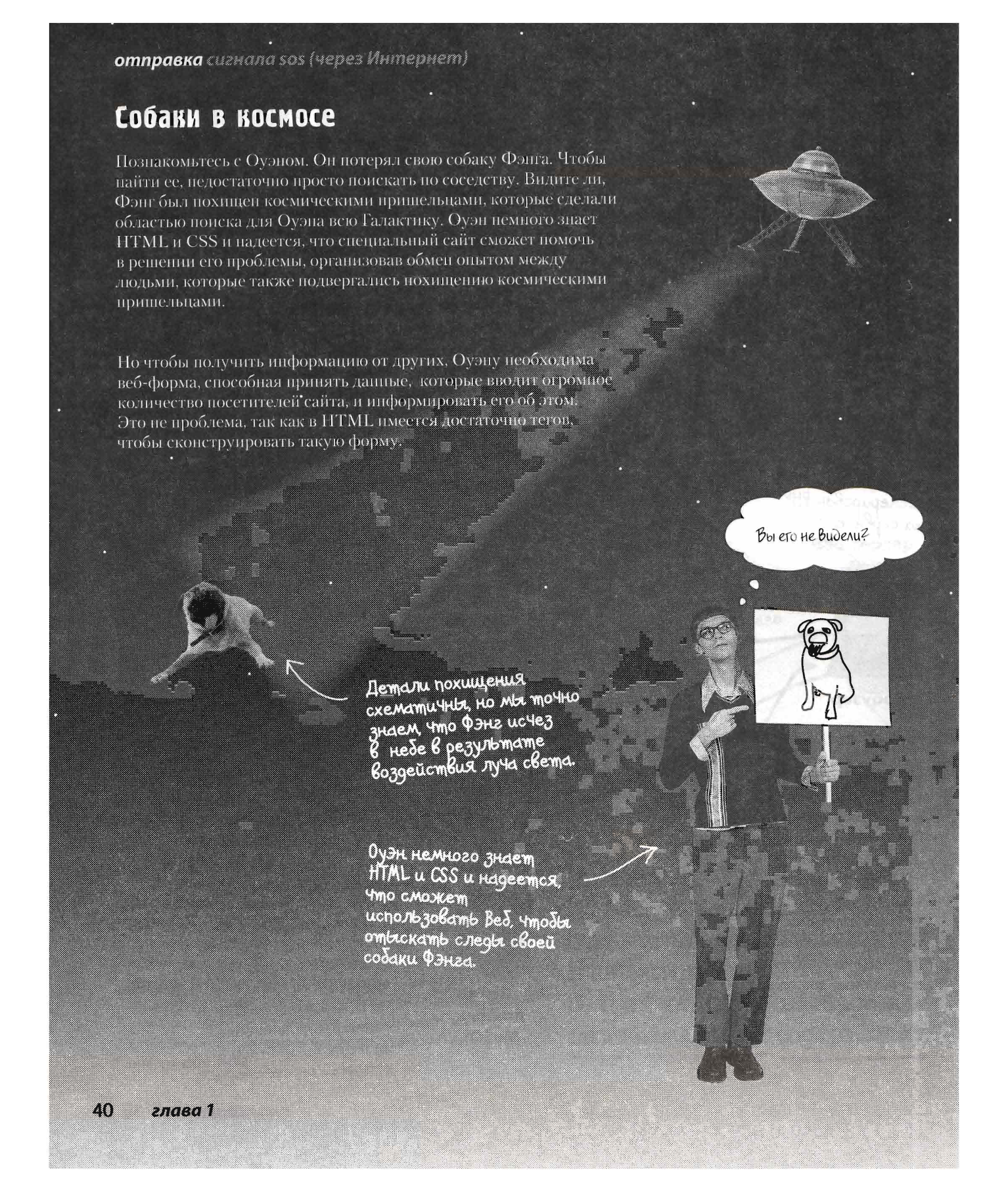
Вместе с PHP веб-сервер способен динамически создавать HTML-страницы на лету.



Собаки в космосе

Познакомьтесь с Оуэном. Он потерял свою собаку Фэнга. Чтобы найти ее, недостаточно просто поискать по соседству. Видите ли, Фэнг был похищен космическими пришельцами, которые сделали областью поиска для Оуэна всю Галактику. Оуэн немного знает HTML и CSS и надеется, что специальный сайт сможет помочь в решении его проблемы, организовав обмен опытом между людьми, которые также подвергались похищению космическими пришельцами.

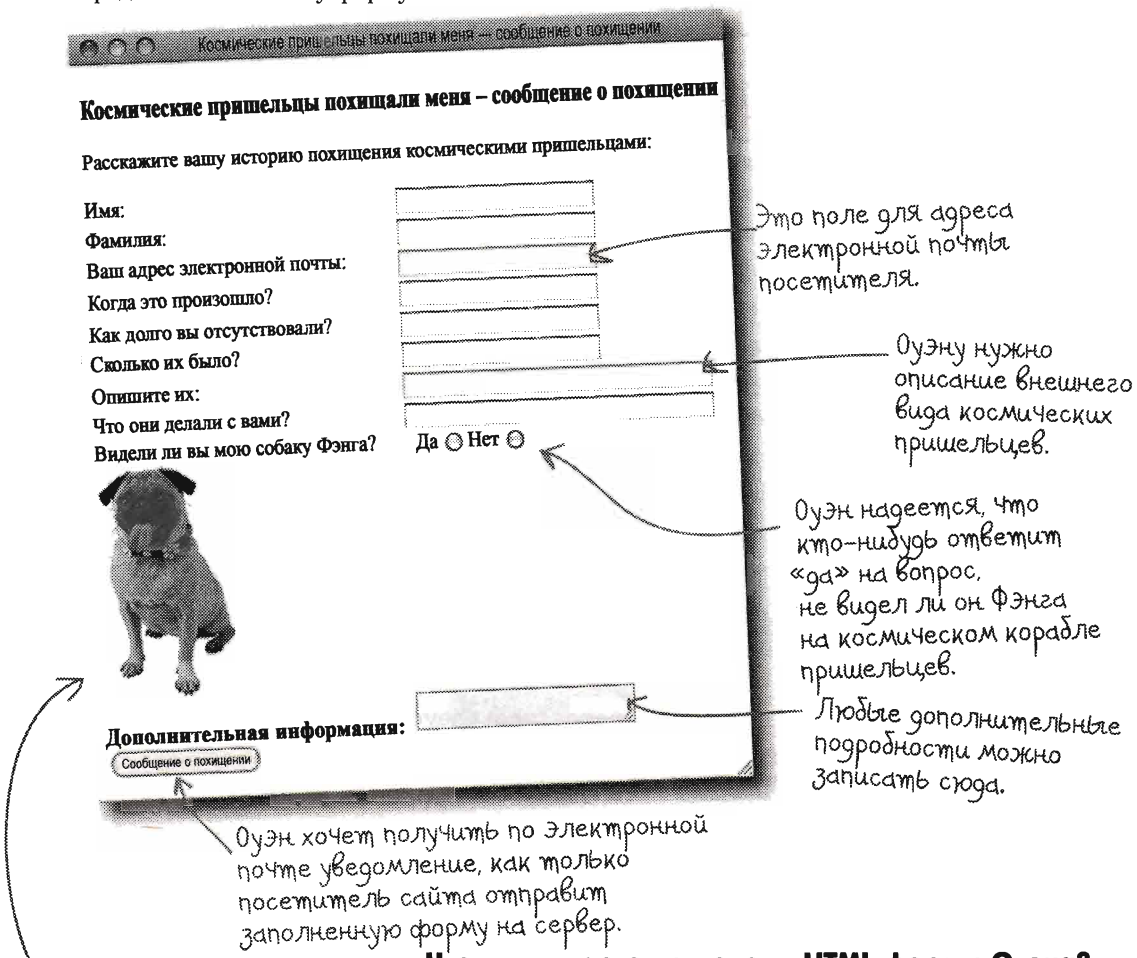
Но чтобы получить информацию от других, Оуэну необходима веб-форма, способная принять данные, которые вводит огромное количество посетителей сайта, и информировать его об этом. Это не проблема, так как в HTML имеется достаточно тегов, чтобы сконструировать такую форму.



Форма помогает Оуэну разобраться в этой истории

Цель нового сайта Оуэна `AliensAbductedMe.com` (`ПришельцыПохищалиМеня.com`) — связать Оуэна с теми, кого похищали пришельцы и кто мог бы пролить свет на исчезновение Фэнга. Оуэн знает, что ему необходима HTML-форма, чтобы запрашивать у посетителей сайта истории их похищений и через которую он смог бы узнать, не сталкивались ли они с Фэнгом во время их межзвездных путешествий. Но ему необходима ваша помощь, чтобы подготовить ее и запустить в работу.

Вот как он представляет себе эту форму.



Что вы думаете по поводу HTML-формы Оуэна?

Как вы думаете, может ли Оуэн столкнуться с какими-либо проблемами, пытаясь получить данные о похищениях космическими пришельцами, используя эту форму? Действуйте, изложите свои соображения.

Формы пишутся на HTML

Форма Оуэна «Сообщение о похищении» создана с использованием исключительно одних тегов и атрибутов HTML. В ней имеются текстовые поля для большинства вопросов, кнопки с зависимой фиксацией, чтобы узнать, видел ли Фэнга посетитель сайта, и область ввода текста для дополнительных подробностей. Форма также предусматривает отправку данных на адрес электронной почты Оуэна.

Если вам необходимо вспомнить, как создаются HTML-формы, обратитесь к главе 14 книги Head First HTML with CSS & XHTML.

Оуэн получит данные, введенные в форму, на этот адрес электронной почты. Измените его на свой, когда будете тестировать форму.

«mailto» — это протокол, позволяющий отправлять по электронной почте данные, введенные в форму.

Значение этого параметра говорит серверу, как отправлять данные. Он может принимать значения «post» или «get». Мы объясним разницу между ними немного позднее.

Теги <input> сообщают форме, что здесь будут вводиться данные.

Атрибут type сообщает форме, что здесь будет введен текст (при его значении «text»).

```
<p>Расскажите историю о похищении вас космическими пришельцами</p>
<form method="post" action="mailto"owen@aliensubductedme.com">
   <label for="firstname">Имя:</label>
   <input type="text" id="firstname" name="firstname" /><br />
   <label for="lastname">Фамилия:</label>
   <input type="text" id=" lastname" name=" lastname" /><br />
   <label for="email">Ваш адрес электронной почты:</label>
   <input type="text" id=" email" name=" email" /><br />
   <label for="whenithappend">Когда это произошло?</label>
   <input type="text" id=" whenithappend" name=" whenithappend" /><br />
   <label for="howlong">Как долго вы отсутствовали?</label>
   <input type="text" id=" howlong" name=" howlong" /><br />
   <label for="howmany">Сколько их было?</label>
   <input type="text" id=" howmany" name=" howmany" /><br />
   <label for=" aliendescription">Опишите их:</label>
   <input type="text" id=" aliendescription" name=" aliendescription" size="32" /><br />
   <label for="whattheydid">Что они делали с вами?</label>
   <input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
   <label for="fangspotted">Видели ли вы мою собаку Фэнга?</label>
   Да <input id=" fangspotted" name=" fangspotted" type="radio" value="yes"/>
   Нет <input id=" fangspotted" name=" fangspotted" type="radio" value="no"/><br />
   <img src="fang.jpg" width="100" height="175"
      alt="Моя похищенная собака Фэнг.">
   <label for="other">Дополнительная информация:</label>
   <textarea id=" other" name=" other" /></textarea><br />
   <input type="sabmit" value="Сообщение о похищении" name="submit" />
</form>
```

Форма ограничена открывающим и закрывающим тегами <form>.

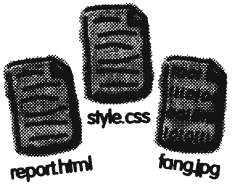
Никаких неожиданностей — это 100%-й код HTML!

Кнопка «Сообщение о похищении» сообщает форме, что необходимо обработать и отправить данные на сервер.

Тест-драйв

Испытайте форму «Сообщение о похищении».

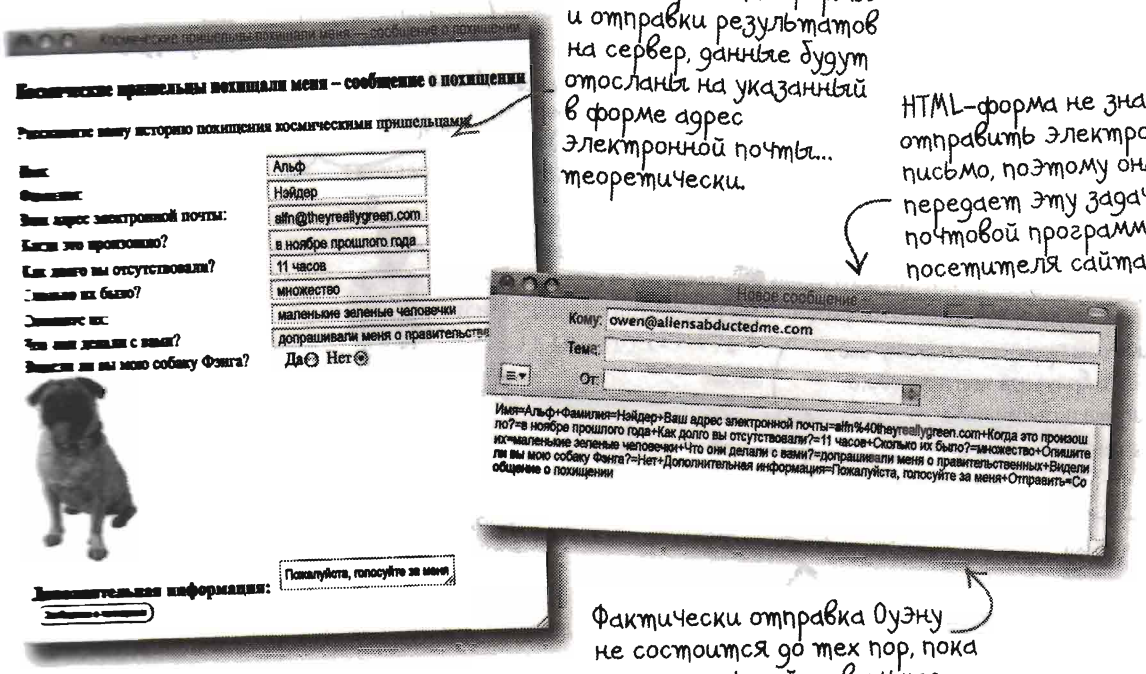
Загрузите страницы «Сообщение о похищении» с сайта по адресу `www.headfirstlabs.com/books/hfphp`. Этот код находится в каталоге `chapter01`. Этот же каталог содержит файл `report.html` с веб-формой Оуэна, а также каскадные таблицы стилей (`style.css`) и изображение Фэнга (`fang.jpg`).



Откройте файл `report.html` в текстовом редакторе и измените адрес электронной почты Оуэна на свой. Затем откройте страницу в браузере, введите в форму какую-нибудь информацию о похищении космическими пришельцами и нажмите кнопку «Сообщение о похищении».

После обработки формы и отправки результатов на сервер, данные будут отосланы на указанный в форме адрес электронной почты... теоретически.

HTML-форма не знает, как отправить электронное письмо, поэтому она передает эту задачу почтовой программе посетителя сайта.

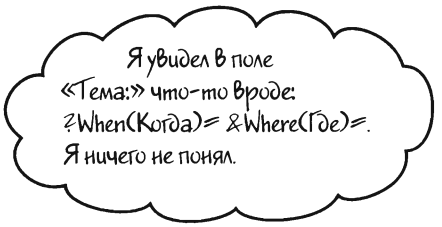


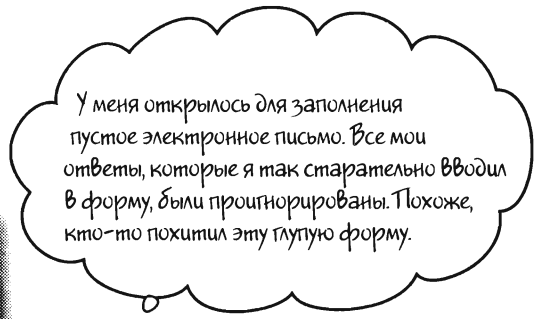
Фактически отправка Оуэну не состоится до тех пор, пока посетитель сайта вручную не отправит эти довольно странно выглядящие данные.

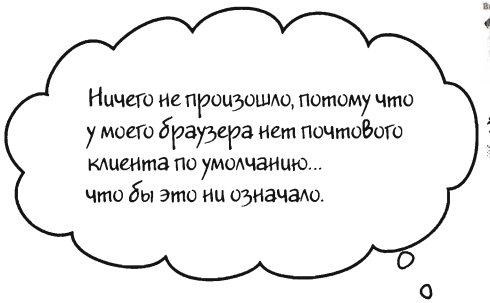
Итак, как вы думаете, придут ли данные обработки формы в виде электронного письма на ваш почтовый ящик?

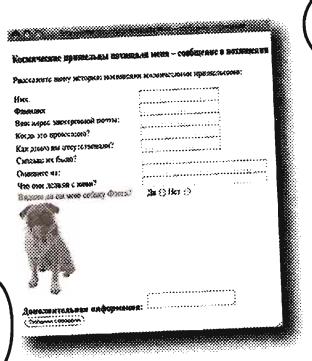
У HTML-формы имеются проблемы

Форма Оуэна «Сообщение о похищении» выполняет свою задачу, но он не получает достаточно информации от посетителей сайта. Может быть, похищение Фэнга является исключительным случаем... или что-то не так с формой? Давайте посмотрим, что говорят посетители сайта.









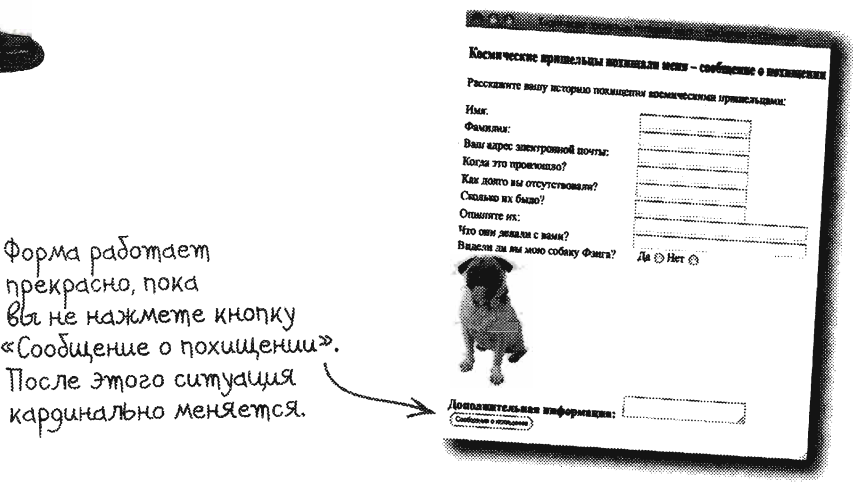
Форма Оуэна больше сбивает посетителей его сайта с толку, чем предоставляет им информацию.

Что происходит? Есть ли у вас какие-нибудь соображения о том, как исправить форму?

Да. Код HTML-формы безукоризнен, но mailto не является удачным способом передачи данных формы.

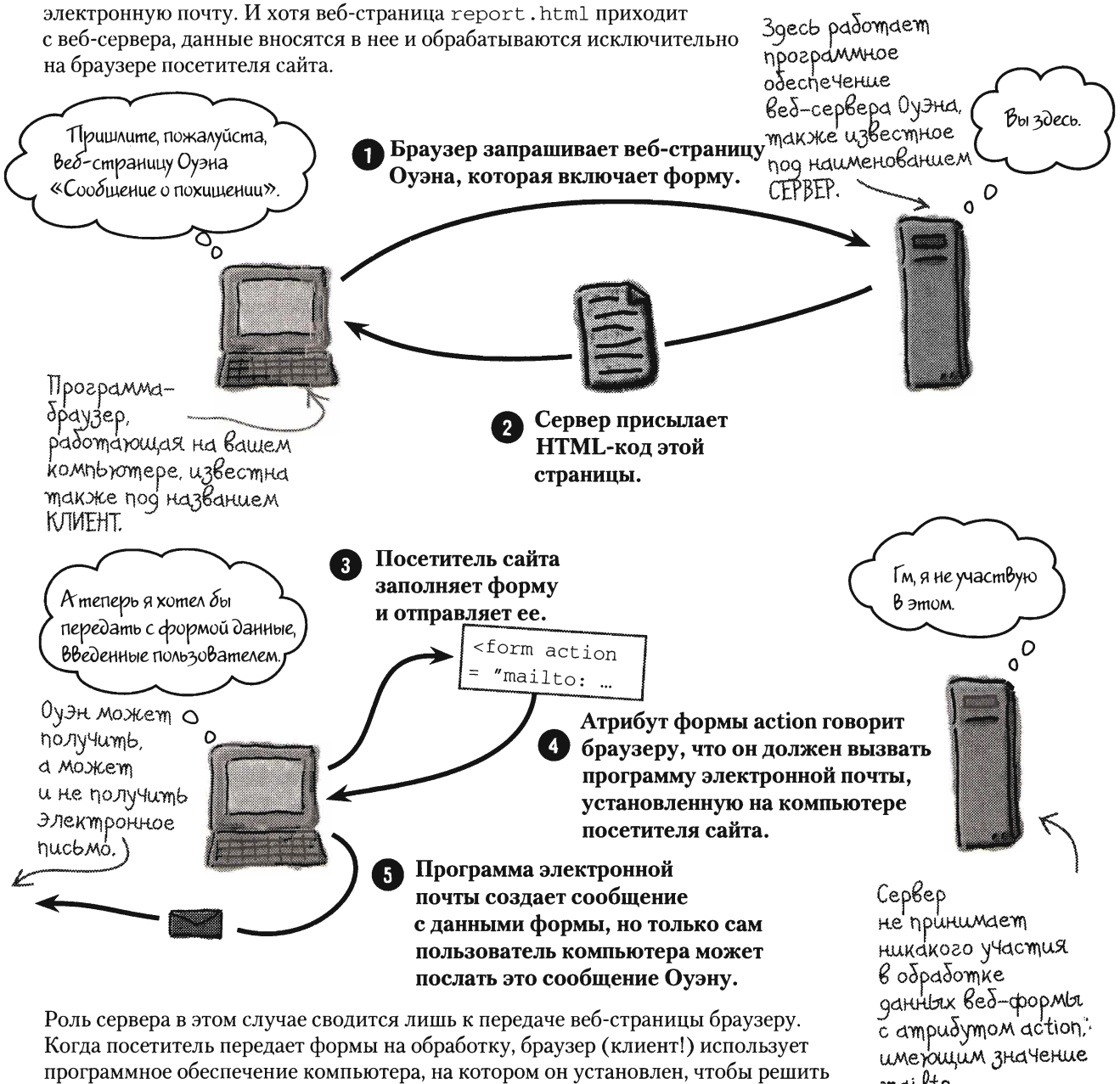
Форма Оуэна абсолютно совершенна до тех пор, пока посетитель сайта не нажмет кнопку «Сообщение о похищении». С этого момента вы полагаетесь на `mailto` в вопросе включения данных формы в электронное письмо. Но это сообщение не будет послано автоматически; вместо этого оно создается в почтовой программе по умолчанию, установленной на компьютере посетителя сайта. И уж совсем неожиданный поворот событий... Посетитель должен сам отправить это письмо, чтобы данные попали к вам! Таким образом, вы не в состоянии управлять процессом отправки электронной почты, в результате чего данные могут успешно пройти свой путь из вашей формы через браузер посетителя в его программу электронной почты и назад к вам в виде электронного письма, а могут и потерпеть в этом неудачу. Не слишком-то хорошо.

Вам необходимо управлять процессом передачи данных, введенных в веб-форму. А конкретней, вам нужно, чтобы PHP упаковал данные в электронное письмо и подтвердил факт их отправки. Это требует переключения вашего внимания от клиентской программы (HTML, `mailto` и т. п.) к серверу (PHP).



HTML-код интерпретируется на клиентской программе

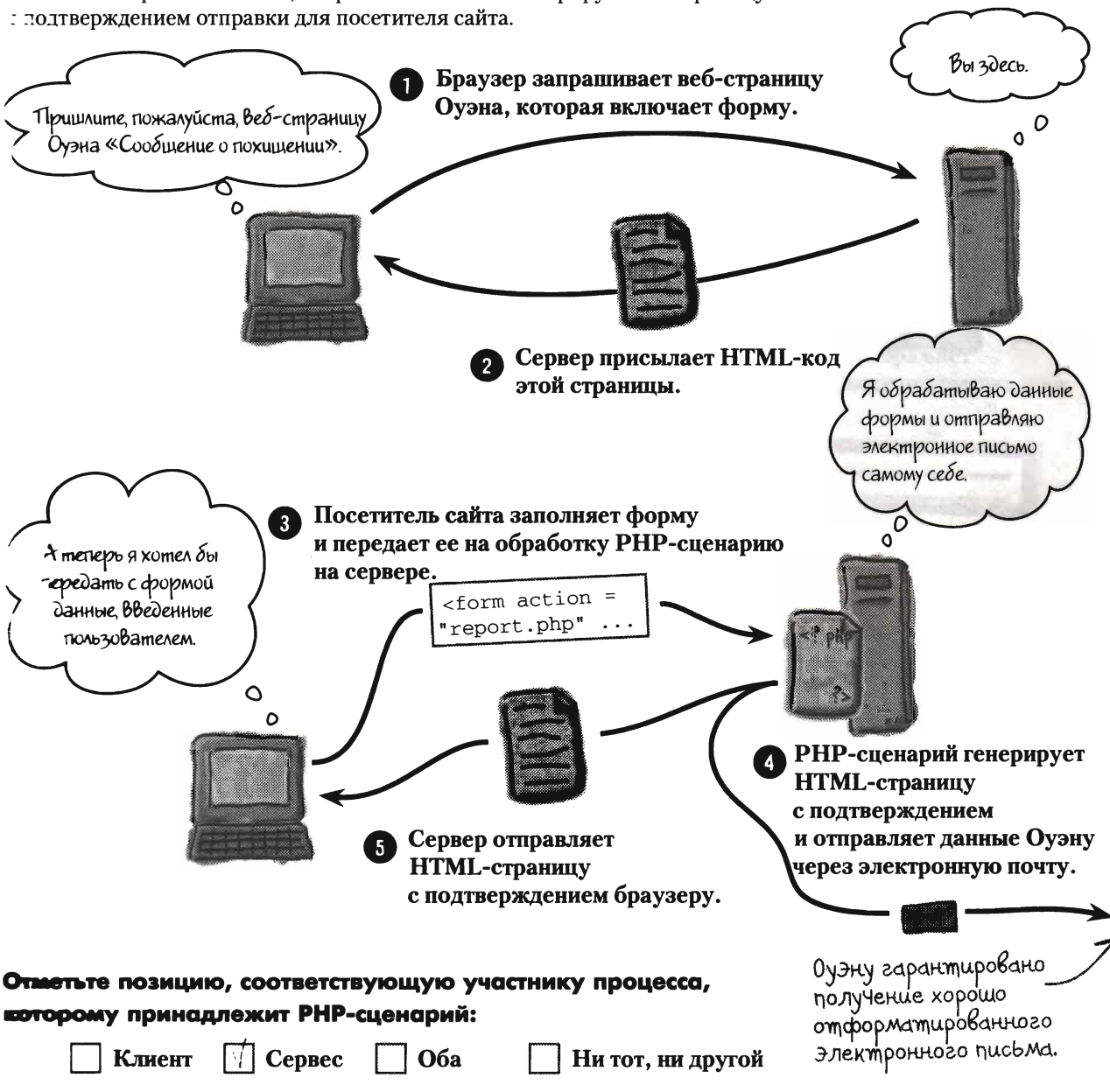
Форма Оуэна написана на чистом HTML с атрибутом `action`, имеющим значение `mailto`, в результате чего после обработки формы осуществляется попытка отправить данные через электронную почту. И хотя веб-страница `report.html` приходит с веб-сервера, данные вносятся в нее и обрабатываются исключительно на браузере посетителя сайта.



Роль сервера в этом случае сводится лишь к передаче веб-страницы браузеру. Когда посетитель передает формы на обработку, браузер (клиент!) использует программное обеспечение компьютера, на котором он установлен, чтобы решить вопрос, как отослать данные через электронную почту. Сам клиент не имеет возможности пересылать данные — это работа сервера.

PHP действует на сервере

PHP дает возможность манипулировать данными, введенными в форму посетителем сайта, отправляя их вам через электронную почту совершенно прозрачно. Посетитель сайта вводит историю своего похищения в форму, нажимает кнопку «Сообщение о похищении» — и все готово! PHP-код создает электронное письмо, отправляет его вам и генерирует веб-страницу с подтверждением отправки для посетителя сайта.



Отметьте позицию, соответствующую участнику процесса, которому принадлежит PHP-сценарий:

☐ Клиент ☑ Сервер ☐ Оба ☐ Ни тот, ни другой

PHP-сценарии интерпретируются на сервере

PHP-код интерпретируется на сервере. Он сохраняется в PHP-сценариях, которые чаще всего имеют расширение имени файла PHP. Эти сценарии часто выглядят так же, как и обычный HTML- и CSS-код. Фактически результатом процесса интерпретации сервером PHP-кода является чистый HTML- и CSS-код. Поэтому каждый PHP-сценарий в конечном счете после окончания исполнения его на сервере преобразуется в HTML и CSS.

Давайте посмотрим внимательнее, как PHP-сценарий изменяет порядок обработки веб-формы Оуэна.

PHP — это серверный язык программирования. Сценарии, написанные на нем, интерпретируются на веб-сервере.

5 **Сервер присылает HTML-страницу, сгенерированную PHP-сценарием.**

PHP-сценарий интерпретируется на сервере!

report.php

Хотя имя страницы показывается в браузере с расширением PHP, на данном этапе это чистый HTML.

6 **Браузер отображает веб-страницу подтверждения.**

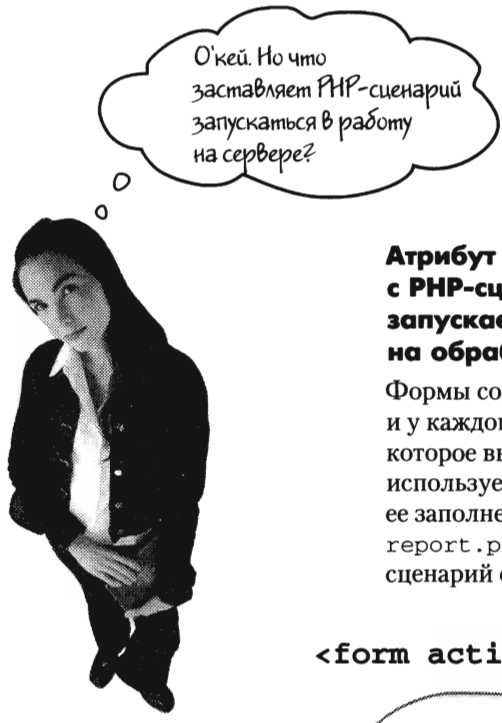
report.php

Посетитель сайта видит веб-страницу подтверждения.

4 **Сервер интерпретирует PHP-сценарий, который отправляет электронное письмо и генерирует HTML-страницу с подтверждением.**

Оуэн получает электронное письмо.

7 **Электронное письмо поступает в почтовый ящик Оуэна.**



Атрибут формы action обеспечивает связь формы с PHP-сценарием, являясь тем элементом, который запускает сценарий, когда форма передается на обработку.

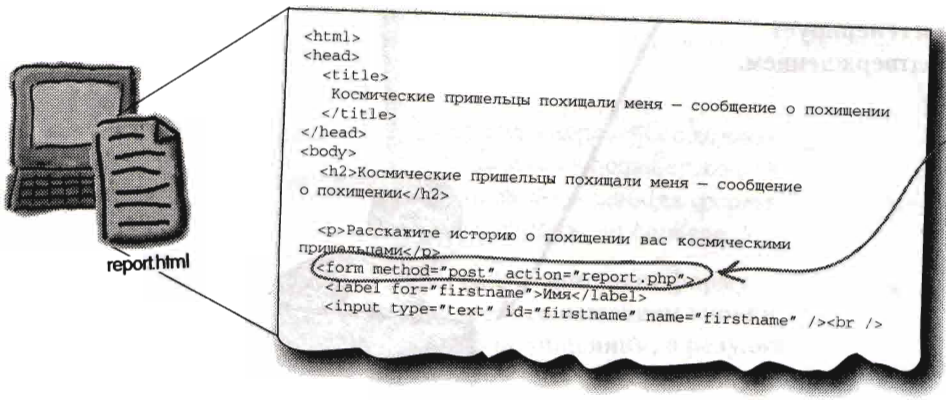
Формы создаются с использованием HTML-тега `<form>`, и у каждого тега `<form>` имеется атрибут `action`. Имя файла, которое вы присваиваете в качестве значения атрибута `action`, используется сервером для обработки формы после подтверждения ее заполнения. Поэтому если имя PHP-сценария Оуэна `report.php`, то атрибут `action`-тега `<form>`, который связывает сценарий с формой, будет выглядеть так:

```
<form action = "report.php" method = "post">
```

Это имя файла вашего PHP-сценария.

Когда посетитель сайта нажмет на форме кнопку «Сообщение о похищении», установленное значение атрибута формы `action` приведет к тому, что файл `report.php` будет запущен на сервере для обработки данных.

Значение атрибута формы action позволяет серверу найти и загрузить файл, который является PHP-сценарием, предназначенным для обработки заполненной формы.



Не бывает глупых вопросов

В: Что означает PHP?

О: PHP — это аббревиатура, которая вначале означала **P**ersonal **H**ome **P**ages (персональные домашние страницы). Позднее значением этой аббревиатуры стало PHP: **H**ypertext **Prep**rocessor (PHP: обработчик гипертекста). Последняя аббревиатура рассматривается как рекурсивная, так как она ссылается на саму себя — аббревиатура в аббревиатуре. Умно? Запутано? Решайте сами!

В: Даже несмотря на то, что мой браузер показывает, что имя веб-страницы оканчивается на .php, она все равно является HTML-страницей. Как такое может быть?

О: Это возможно потому, что страница первоначально создана как PHP-код на сервере, но **преобразована** в HTML-код перед тем, как была отправлена на браузер. Таким образом, сервер интерпретирует PHP-код и конвертирует его в HTML-код перед тем, как отправить его браузеру для просмотра. Это означает, что хотя .PHP-файл содержит PHP-код, браузер никогда не видит этого; он видит только HTML-код, который появляется в результате исполнения PHP-кода на сервере.

В: Но разве не все веб-страницы создаются на сервере, даже страницы на чистом HTML в HTML-файлах?

О: Все файлы сайта сохранены на серверах HTML-, CSS-, PHP-серверах и т. п. Но не все они **интерпретируются** на этих серверах. HTML- и CSS-файлы, так же как файлы изображений, сервер пересылает браузеру непосредственно, не обращая ни малейшего внимания на то, что в действительности в них содержится. PHP-файлы отличаются тем, что они содержат код, который **интерпретируется** на сервере. Браузеру отправляется не сам PHP-код, а **результат** его исполнения, который является чистым HTML и CSS.

Используйте PHP для доступа к данным формы

Итак, Оуэну необходим PHP-сценарий, чтобы процесс получения данных формы о похищении пришельцами был более надежным, чем при использовании `mailto`. Давайте создадим его. Не беспокойтесь, если что-то вам пока не понятно. Мы разберемся.

Это вполне допустимо для PHP-сценария — включать обычные HTML-теги и атрибуты.

PHP-код часто начинается точно так же, как HTML-страница.

```
<html>
<head>
  <title>Космические пришельцы похищали меня — сообщение
о похищении</title>
</head>
<body>
  <h2>Космические пришельцы похищали меня — сообщение
о похищении</h2>
```

Весь этот блок сценария — это PHP-код... Все остальное — обычный HTML.

Вот здесь начинаются интересные вещи: это начало PHP-кода.

```
<?php
  $when_it_happened = $_POST['whenithappened'];
  $how_long = $_POST['howlong'];
  $alien_description = $_POST['description'];
  $fang_spotted = $_POST['fangspotted'];
  $email = $_POST['email'];

  echo 'Спасибо за заполнение формы.<br />';
  echo 'Вы были похищены ' . $when_it_happened;
  echo ' и отсутствовали в течение ' . $how_long . '<br />';
  echo 'Опишите их: ' . $alien_description . '<br />';
  echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
  echo 'Ваш адрес электронной почты: ' . $email;
?>
```

В этом фрагменте PHP-кода считываются данные формы, чтобы они могли быть выведены на экран как часть страницы подтверждения.

Здесь мы используем PHP, чтобы сгенерировать HTML-код из данных формы.

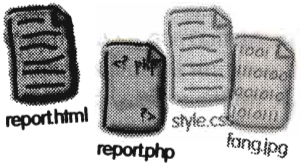
```
</body>
</html>
```

Так же как и обычная веб-страница, этот PHP-сценарий заканчивается закрытием открытых HTML-тегов.

Тест-драйв

Внесите изменения в форму Оуэна с использованием PHP для обработки данных формы.

Создайте новый текстовый файл с именем `report.php` и введите весь текст предыдущей страницы. Это сценарий, который будет обрабатывать веб-форму Оуэна.

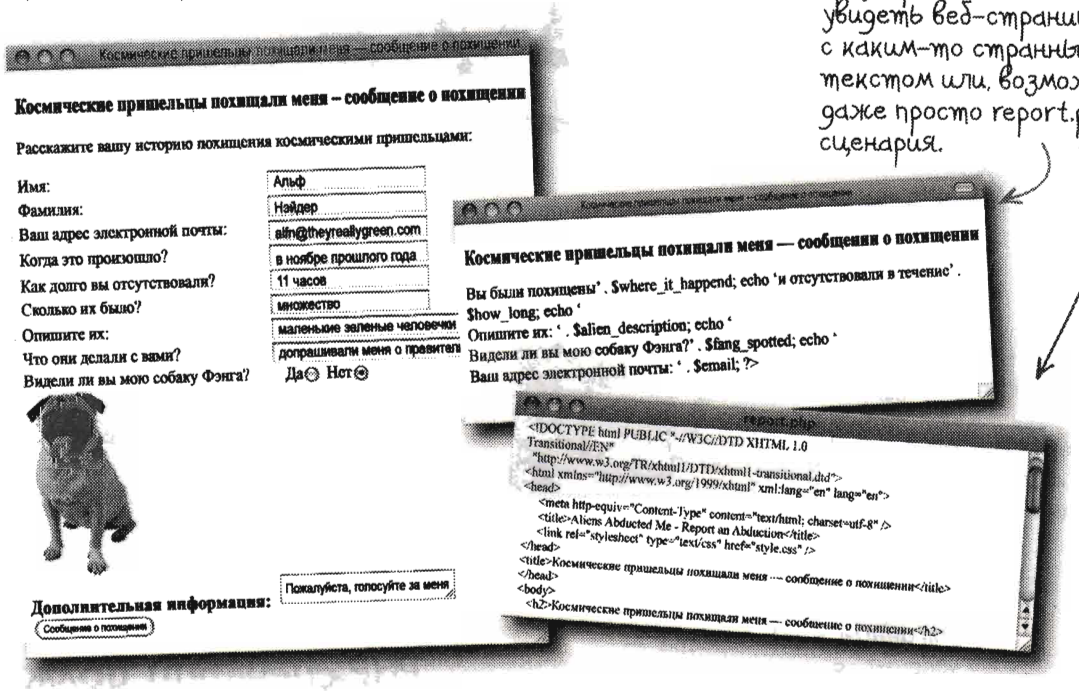


PHP-сценарий пока не подключен к форме, поэтому откройте файл `report.html` в текстовом редакторе и измените атрибут формы `action` с `mailto` на `report.php`.

```
<form action = "report.php" method = "post">
```

Откройте страницу `report.html` в браузере, введите в форму какие-нибудь данные о похищении пришельцами и нажмите кнопку «Сообщение о похищении».

В зависимости от вашего браузера вы можете увидеть веб-страницу с каким-то странным текстом или, возможно, даже просто report.php сценария.



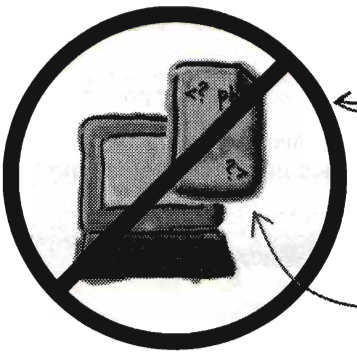
Как вы думаете, так и должен интерпретироваться PHP-сценарий? Напишите, почему да или почему нет, и что, по-вашему, происходит.

PHP-сценарий должен размещаться на сервере

Если на вашем локальном компьютере нет работающего веб-сервера, сценарий `report.php` не будет исполняться, когда вы нажмете кнопку «Сообщение о похищении» на форме. Не забывайте: PHP — это язык программирования, и ему необходимо окружение, в котором он будет интерпретироваться. Этим окружением является веб-сервер с поддержкой PHP. Для того чтобы PHP-сценарий делал то, для чего он предназначен, он и веб-страницы, которые используют его, должны размещаться на реальном веб-сервере. Непосредственное открытие сценария в вашей локальной файловой системе в отсутствие веб-сервера не приведет к ожидаемому результату.

Если на вашем локальном компьютере установлен веб-сервер и он поддерживает PHP, то вы можете протестировать PHP-сценарий.

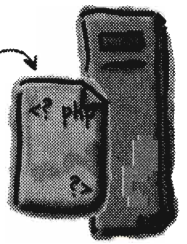
В отличие от HTML-страниц, которые могут быть открыты локально в браузере, PHP-сценарий всегда должен «открываться» через URL на сервере.



Браузер ничего не знает о PHP и поэтому не имеет возможности интерпретировать PHP-сценарии.

Для браузера этот PHP-сценарий — всего лишь набор бессмысленного кода.

Веб-сервер понимает этот PHP-код и может выполнить сценарий.



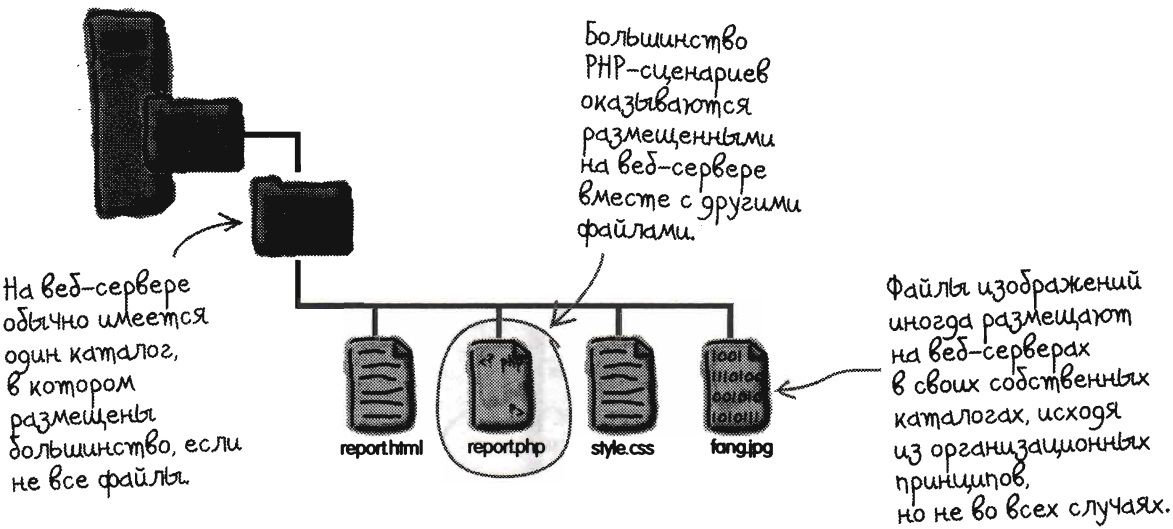
Веб-серверы с поддержкой PHP могут интерпретировать PHP-сценарии и преобразовывать их в HTML-страницы, понятные для браузера.

Самый быстрый способ определить, получена ли веб-страница с веб-сервера, — это убедиться, что ее URL начинается с «http:». URL веб-страницы, открытой как локальный файл, всегда начинается с «file:».

PHP-сценарии должны интерпретироваться на веб-сервере, иначе они не будут выполнять свою функцию.

Поместите свои PHP-сценарии на сервер

Совершенно нормально, если вы создаете и редактируете свои PHP-сценарии на локальном компьютере. Но вам необходимо переместить файлы на веб-сервер, где они будут интерпретироваться. PHP-файлы часто размещают на сервере вместе с HTML-файлами, и в этом нет ничего таинственного. Для загрузки файлов на веб-сервер требуется соответствующая утилита, например FTP.



Одной только загрузки ваших PHP-сценариев недостаточно. Этот веб-сервер должен также поддерживать PHP. Некоторые серверы включают такую поддержку по умолчанию, некоторые — нет.

Не бывает глупых вопросов

В: Как я узнаю, установлен PHP на моем сервере или нет?

О: Вы можете спросить веб-администратора — своего или хостинговой компании либо провести тест самостоятельно. Создайте текстовый файл с именем `test.php` и введите в него следующий код:

```
<?php
  phpinfo();
?>
```

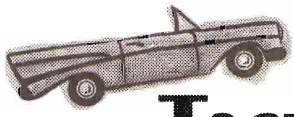
Этот код предлагает PHP отобразить информацию о себе.

А теперь загрузите файл `test.php` на свой веб-сервер и введите его URL в адресную строку вашего браузера. Если на вашем сервере установлен PHP, вы увидите большое количество детальной информации о PHP, включая его версию. Вы сделали это!

Не забудьте удалить сценарий php.info() по окончании вашего тестирования, чтобы никто другой не смог воспользоваться этой информацией.

Если PHP не установлен на вашем веб-сервере, обратитесь к Приложению II.

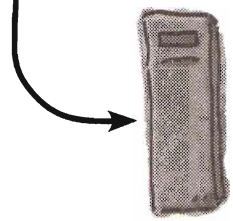
В нем вы найдете инструкции, как установить и запустить PHP на вашем веб-сервере.

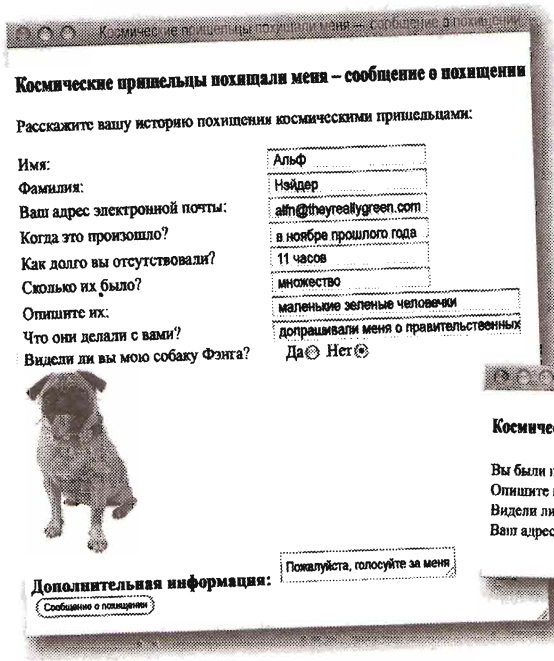


Тест-драйв

Загрузите файлы «Сообщение о похищении» на веб-сервер и попытайтесь открыть форму... опять.

Загрузите `report.html`, `report.php`, `style.css` и `fang.jpg` на веб-сервер, который поддерживает PHP. Введите URL страницы `report.html` в адресную строку своего браузера, занесите в форму данные о похищении космическими пришельцами и нажмите кнопку «Сообщение о похищении».





PHP-сценарий работает! Он выводит данные формы на веб-странице подтверждения.

Круто! Теперь все, что вам нужно, — это только добавить немного PHP-кода, который позаботится о пересылке данных формы по электронной почте.

Правильно. В сценарии report.php все еще отсутствует код пересылки по электронной почте Оуэну данных о похищении космическими пришельцами.

Но это не проблема, так как PHP предлагает функцию (встроенный в язык фрагмент многократно используемого кода), которую вы можете использовать для отправки электронного письма. Все, что вам необходимо, — это определить, что это сообщение должно содержать, а затем создать и отправить его с помощью PHP.

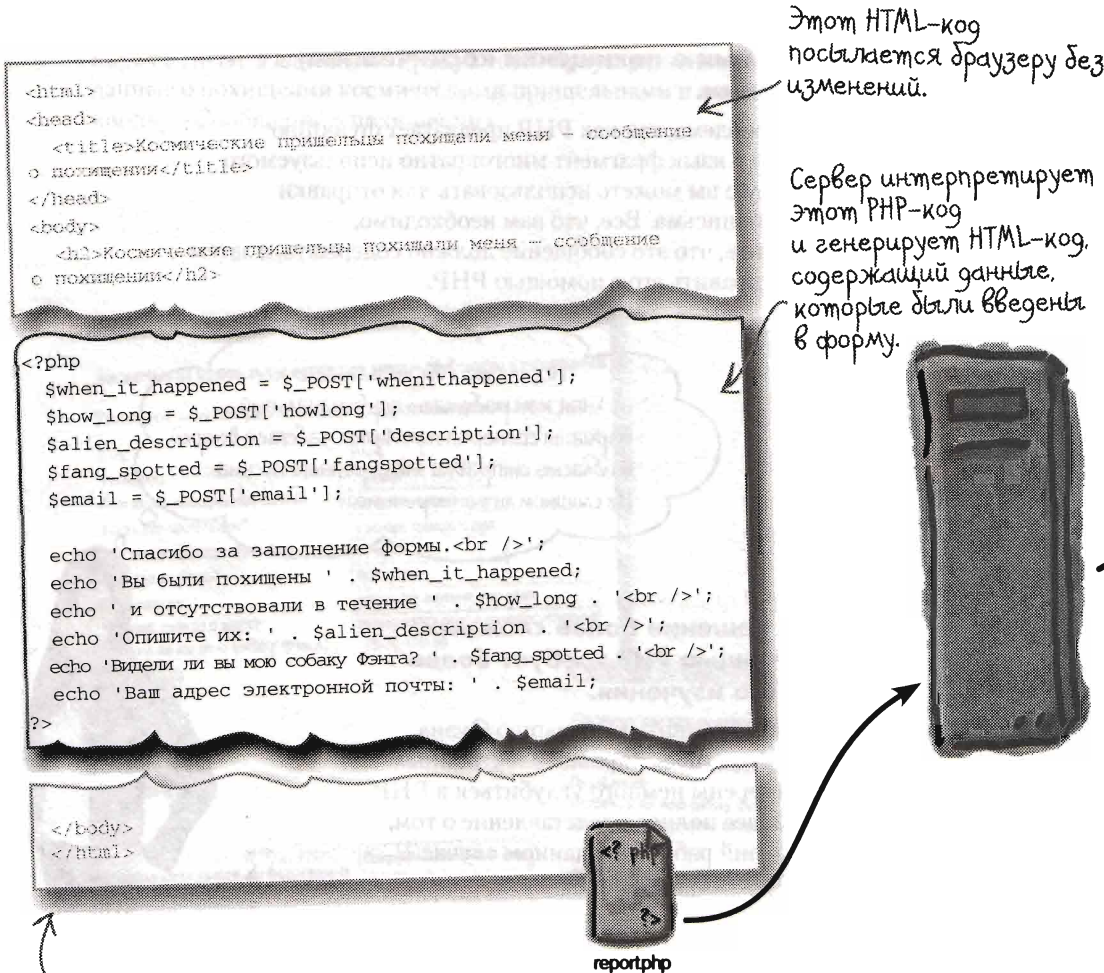
Постойте! Мы, толком еще не зная, как работает первоначальный вариант сценария report.php, забегаем вперед, в область отправки электронного письма. Не слишком ли мы торопимся?

Это верно. Решение более сложных задач с помощью PHP требует более глубокого его изучения.

Поэтому, чтобы добавить к сценарию Оуэна `report.php` функцию отправки электронных писем, мы намерены немного углубиться в PHP и составить более полное представление о том, как этот сценарий работает в данном случае.

Сервер преобразует PHP в HTML

Для того чтобы взять в толк, как работает PHP-сценарий, нужно понять, что с ним происходит, когда он интерпретируется на сервере. Большинство PHP-сценариев содержат как PHP-, так и HTML-код. PHP интерпретируется и преобразуется в HTML перед тем, как сервер отправляет законченную работу (в HTML) браузеру. В сценарии Оуэна `report.php` PHP-код генерирует большую часть HTML-контента в теге `<body>` страницы подтверждения. HTML-код вне этого тега остается без изменения.



Этот HTML-код создан на лету PHP-сценарием, который позволяет решать достаточно сложные задачи, например извлекать данные, только что введенные в форму, и комбинировать их с другими.

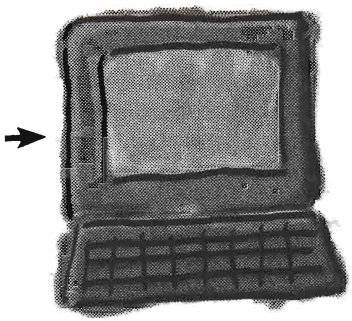
Статичный код — он не изменяется.

Динамичный код — изменяется каждый раз, когда кто-нибудь внесет данные в форму!

```
<html>
<head>
  <title>Космические пришельцы похищали меня — сообщение о
похищении</title>
</head>
<body>
  <h2>Космические пришельцы похищали меня — сообщение о похищении
  </h2>
```

```
Спасибо за заполнение формы.<br />
Вы были похищены в ноябре прошлого года и отсутствовали в течение
11 часов. .<br />
Опишите их: .<br />
Видели ли вы мою собаку Фэнга? Нет.<br />
Ваш  адрес электронной почты:  alfn@theyreallygreen.com
```

```
</body>
</html>
```



report.php

Конечный результат интерпретации PHP — это чистая HTML-страница, динамически генерируемая на сервере.

Разбор PHP-сценария Оуэна

report.php

Сценарий `report.php` запускается в работу при нажатии кнопки «Сообщение о похищении», и его задача (на текущий момент) — прочитать данные формы и сгенерировать веб-страницу подтверждения. Давайте посмотрим, как это происходит.

Первый фрагмент кода — это чистый HTML. Он просто конфигурирует страницу, которую мы создаем, и включает несколько HTML-тегов, необходимых для любой веб-страницы.

```
<html>
<head>
  <title>Космические пришельцы похищали меня — сообщение
о похищении</title>
</head>
<body>
  <h2>Космические пришельцы похищали меня — сообщение
о похищении</h2>
```

Да, этот HTML-код неполон: в идеале здесь должны присутствовать теги DOCTYPE, <meta> и т. п., но мы не приводим их для простоты.

Дальше становится интереснее. Здесь мы прерываем HTML-код и входим в PHP. Тег `<?php` открывает раздел сценария, содержащий PHP-код: все, что следует за этим тегом, является чистым PHP-кодом.

```
<?php
```

Начиная с этого момента мы имеем дело с PHP-кодом... по крайней мере, пока мы не достигнем закрывающего тега ?>.

Этот код читает данные формы и присваивает их значения индивидуальным переменным. Появляется простая возможность получить доступ к значениям этих переменных в дальнейшем. PHP-**переменные** позволяют вам сохранять данные любого типа, будь то числа, текст или что-нибудь иное.

```
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$alien_description = $_POST['description'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
```

Каждая строка PHP-кода присваивает новым переменным данные, введенные в поля формы.

А теперь мы сообщаем посетителю сайта о результатах обработки формы. В этом месте переменные, которые мы только что создали, используются в работе. Мы вставляем их значения в динамически генерируемый HTML-код. Выводом команды `echo` является HTML-код, который непосредственно передается браузеру.

```
echo 'Спасибо за заполнение формы.<br />';
echo 'Вы были похищены ' . $when_it_happened;
echo ' и отсутствовали в течение ' . $how_long . '<br />';
echo 'Опишите их: ' . $alien_description . '<br />';
echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
echo 'Ваш адрес электронной почты: ' . $email;
```

Этот PHP-код вставляет значения переменных в HTML-код, который передается браузеру.

Тег `?>`, соответствующий тегу `<?php`, закрывает раздел сценария, содержащий PHP-код. С этого момента мы возвращаемся в HTML-раздел.

```
?>
```

Здесь заканчивается PHP-код: с этого момента мы возвращаемся к обычному HTML.

А теперь закончим страницу, закрыв HTML-теги, которые мы открыли ранее.

```
</body>
</html>
```

Не забывайте, что мы создаем HTML-страницу: необходимо закрыть ранее открытые HTML-теги.

Несколько PHP-правил ~~кодирования~~ существования

Сценарий Оуэна `report.php` демонстрирует несколько фундаментальных правил языка PHP, которые относятся ко всем PHP-сценариям. Давайте рассмотрим их подробнее.

☑ PHP-код всегда заключен между тегами <?php и ?>.

Здесь должен быть ваш PHP-код.

```
<?php
...
?>
```

Большей частью PHP-сценарии — это просто HTML-страницы со вставленным в них PHP-кодом. Эти теги говорят серверу, какую часть общего кода он должен рассматривать как PHP-код.

☑ Каждое предложение PHP должно заканчиваться точкой с запятой (;).

```
echo 'Спасибо за заполнение формы.<br />';
```

Если выполнение вашего кода заканчивается ошибкой, убедитесь в том, что не забыли про точку с запятой. Это случается гораздо чаще, чем вы думаете.

Точка с запятой дает понять PHP, что это конец PHP-предложения.

☑ Если на веб-странице имеется любое количество PHP-кода, лучше использовать в имени файла на веб-сервере расширение PHP, а не HTML.

Хорошая мысль — давать PHP-сценариям имена с расширением PHP. И это не просто более короткое расширение,

☑ Имена переменных PHP должны начинаться со знака доллара ($).

```
$email = $_POST['email'];
```

Знак доллара однозначно идентифицирует PHP-переменную, которая содержит информацию внутри сценария PHP.

Рассматривая переменные, использованные в сценарии report.php, обнаружили ли вы еще какие-нибудь правила PHP, имеющие отношение к переменным? Запишите их!

Поиск допустимых имен для переменных

Переменная — это контейнер, в который вы можете заносить данные, и каждая переменная должна иметь уникальное имя.

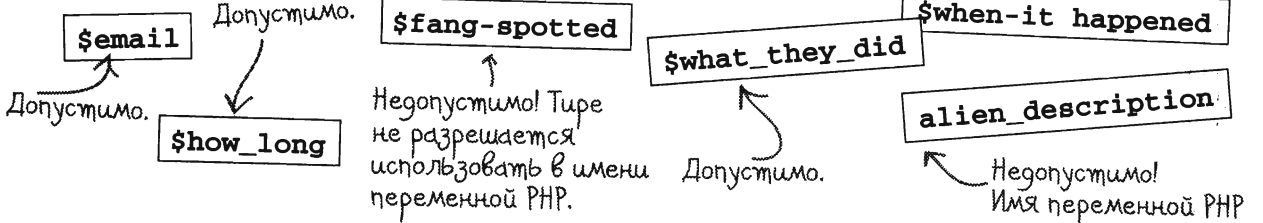
В дополнение к тому, что имя переменной PHP должно начинаться с символа $, оно зависит от регистра клавиатуры (строчная и прописная буквы рассматриваются как разные). Но и это не все. Имеются другие важные правила, регламентирующие выбор имени переменной. Некоторые из этих правил относятся к синтаксическим: интерпретация кода будет прервана в случае их игнорирования. Другие относятся к хорошему стилю программирования, полученному нами в наследство от добрых PHP-программистов.

Давайте начнем с официальных правил, которые, безусловно, приведут к возникновению проблем в случаях, если вы будете их игнорировать. Следуйте этим правилам для создания **допустимых** имен переменных.

- ☑ **Первый символ должен быть знаком доллара ($).**
- ☑ **Имя переменной должно состоять хотя бы из одного символа.**
- ☑ **Символ, следующий непосредственно за символом доллара, должен быть либо буквой английского алфавита, либо символом подчеркивания (_), при этом все последующие символы могут быть буквами, цифрами или символами подчеркивания.**
- ☑ **Пробелы и специальные символы, кроме _ и $, не допускаются в любой части имени переменной.**

Вот он! Не считая символа $, который должен быть в имени каждой переменной.

Недопустимо! Тире и пробел не разрешается использовать в именах переменных PHP.



Несоблюдение этих правил приведет к тому, что ваш код станет невыполнимым. Но кроме них есть еще правила, которые следует рассматривать большей частью как языковые соглашения. Они помогут сделать ваш код более совместимым и легким для прочтения и понимания.

- ☑ **Используйте строчные буквы для имен переменных.**
- ☑ **Отделяйте слова символом подчеркивания в многословных именах переменных.**

Несоблюдение двух последних правил не прервет процесс выполнения кода. В своей практике вы наверняка столкнетесь с примерами их игнорирования, причем такой код будет исполняться без проблем, потому что эти правила являются скорее стилистическими соглашениями. Но они сослужат вам хорошую службу, когда вы начнете давать имена своим собственным переменным.

Имена PHP-переменных должны начинаться со знака доллара и не могут включать пробелов.

Не бывает глупых вопросов

В: Имеет ли значение, ввожу я команды PHP прописными или строчными буквами?

О: И да и нет. В большей своей части PHP не зависит от регистра клавиатуры, поэтому вы можете смешивать регистры при вводе большинства команд. Например, вы можете использовать `echo`, `ECHO` или `EchO`, когда вы хотите вывести какие-либо данные. Тем не менее учитывая вопросы совместимости, считается хорошим стилем придерживаться сложившихся в программистском сообществе соглашений. Большинство программистов PHP предпочитают использовать строчные символы при написании команд, поэтому вы увидите `echo` во всех примерах кода этой книги.

В: Значит, хоть это и плохой стиль программирования, я могу смешивать и совмещать PHP-код, написанный в разных регистрах?

О: Нет, не во всех случаях. Исключением из этого правила является использование регистров клавиатуры при определении имен переменных, которые вы создаете для сохранения данных. Возьмем, например, переменную `$email`, используемую в сценарии «Сообщение о похищениях». Имя этой переменной является зависимым от регистра клавиатуры, поэтому вы не можете обращаться к ней как к `$EMAIL` или `$eMail`. Все имена переменных в PHP зависимы от регистра клавиатуры, поэтому необходимо быть очень внимательным при их выборе и ссылках на них в своем коде. Дополнительно об именах переменных чуть позднее.

В: Это действительно допустимо — использовать и PHP-, и HTML-код в одном и том же файле?

О: Абсолютно. Во многих случаях это просто необходимо.

В: Почему у меня может возникнуть в этом необходимость?

О: Потому что основная задача, возложенная на веб-сервер, заключается в том, что он должен предоставлять браузеру HTML-страницы. PHP не меняет этого принципа. Что PHP позволяет вам делать, так это менять HTML-контент на лету, используя, например, сегодняшнюю дату, информацию, запрошенную в базе данных, или даже результаты вычисления нескольких данных, например суммарную стоимость всех товаров в вашей покупательской корзине. Таким образом, PHP позволяет вам манипулировать HTML-кодом, который формирует веб-страницу, чего невозможно добиться при использовании статического кода, то есть разработанного во время создания такой страницы.

В: Должен ли PHP-код, вставляемый в HTML-файл, находиться в своей собственной строке или я могу вставлять его в HTML-строки, например, как часть атрибута HTML-тега?

О: Кроме требования размещения PHP-кода между тегами `<?php` и `?>` не существует никаких ограничений на встраивание его в HTML-код. Фактически очень часто необходимо «вклинить» фрагмент PHP-кода в середину HTML-кода, как, например, в случаях, когда вы устанавливаете значение атрибута HTML-тега. Это является совершенно допустимым использованием PHP.

В: Я видел PHP-код, ограниченный `<?` в качестве стартового тега вместо `<?php`. Это правильно?

О: Не совсем. Говоря технически, это допустимо, но не рекомендуется. В принципе, сервер можно сконфигурировать на допустимость короткого стартового тега (`<?`), в то время как обычный стартовый тег (`<?php`) работает всегда, поэтому лучше использовать его и быть уверенным, что он будет работать при любых обстоятельствах.

В: Если веб-сервер всегда отправляет браузеру чистый HTML-код, почему URL ссылается на имя PHP-сценария, скажем, webpage.php?

О: Не забывайте, что каждая веб-страница — это результат двухстороннего обмена информацией, включающего запрос от браузера и ответ от веб-сервера. URL — это основа запроса, в то время как контент, полученный от сервера, — это ответ. PHP-сценарии запрашиваются как обычные HTML-страницы через URL, введенный в адресную строку браузера, или гиперссылку с другой страницы, или как результат передачи формы на обработку. Это объясняет, почему URL для PHP-страницы ссылается на имя PHP-сценария.

Вторая часть этого диалога — это ответ сервера, который генерируется в результате интерпретации PHP-сценария. Большинство PHP-сценариев генерируют HTML-код, но только этот код и имеет смысл для браузера. Поэтому невозможен случай, при котором сервер, получив запрос с URL, ссылающимся на .PHP-файл, отправил бы браузеру содержащийся в этом файле PHP-код вместо HTML-кода, полученного в результате его интерпретации.

В: Могут ли переменные PHP содержать другие типы данных?

О: Да. Вы можете использовать переменные, содержащие булевы (true/false) данные. А числовые данные могут быть как целыми числами, так и числами с плавающей запятой (десятичные дроби). Существуют также массивы, содержащие множество данных с кодом, позволяющим манипулировать ими. Массивы рассматриваются немного дальше в этой главе, а объекты — в главе 12. Имеется также специальный тип данных `NULL`, который представляет собой полное отсутствие какого-либо значения. Например, переменная, которой не было присвоено никакого значения, рассматривается как переменная со значением `NULL`.

Или память PHP не настолько хороша, или что-то не так со сценарием... Некоторые данные потеряны.

Космические пришельцы похищали меня — сообщение о похищении

Космические пришельцы похищали меня – сообщение о похищении

Расскажите вашу историю похищения космическими пришельцами:

Имя:	Альф
Фамилия:	Нэйдер
Ваш адрес электронной почты:	alfn@theyreallygreen.com
Когда это происошло?	в ноябре прошлого года
Как долго вы отсутствовали?	11 часов
Сколько их было?	множество
Опишите их:	маленькие зеленые человечки
Что они делали с вами?	допрашивали меня о правительственных
Видели ли вы мою собаку Фэнга?	Да ○ Нет ◉

Дополнительная информация: Пожалуйста, голосуйте за меня

Сообщение о похищении

Ясно видно, что описание космических пришельцев введено в форму...

...но это описание отсутствует на странице подтверждения.

Космические пришельцы похищали меня — сообщение о похищении

Космические пришельцы похищали меня — сообщение о похищении

Вы были похищены в ноябре прошлого года и отсутствовали в течение 11 часов.
Опишите их:
Видели ли вы мою собаку Фэнга? Нет
Ваш адрес электронной почты: alfn@theyreallygreen.com

Время поработать

В сценарии Оуэна `report.php` имеется проблема с данными формы «Описание космических пришельцев». Просмотрите все строки кода, которые, на ваш взгляд, имеют отношение к этой проблеме, и напишите, что они делают. Как вы думаете, в чем проблема?

```
<html>

<head>

  <title>Космические пришельцы похищали меня – сообщение
о похищении</title>

</head>

<body>

  <h2>Космические пришельцы похищали меня – сообщение
о похищении</h2>

<?php

   $when_it_happened = $_POST['whenithappened'];

   $how_long = $_POST['howlong'];

   $alien_description = $_POST['description'];

   $fang_spotted = $_POST['fangspotted'];

   $email = $_POST['email'];

echo 'Спасибо за заполнение формы.<br />';

echo 'Вы были похищены ' . $when_it_happened;

echo ' и отсутствовали в течение ' . $how_long . '<br />';

echo 'Опишите их: ' . $alien_description . '<br />';

echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';

echo 'Ваш адрес электронной почты: ' . $email;

?>


</body>

</html>
```

reportphp

Решение задачи

В сценарии Оуэна `report.php` имеется проблема с данными формы «Описание космических пришельцев». Просмотрите все строки кода, которые, на ваш взгляд, имеют отношение к этой проблеме, и напишите, что они делают. Как вы думаете, в чем проблема?

```
<html>

<head>

  <title>Космические пришельцы похищали меня — сообщение
о похищении</title>

</head>

<body>

  <h2>Космические пришельцы похищали меня — сообщение
о похищении</h2>

<?php

   $when_it_happened = $_POST['whenithappened'];

   $how_long = $_POST['howlong'];

   $alien_description = $_POST['description'];

   $fang_spotted = $_POST['fangspotted'];

   $email = $_POST['email'];


echo 'Спасибо за заполнение формы.<br />';

echo 'Вы были похищены ' . $when_it_happened;

echo ' и отсутствовали в течение ' . $how_long . '<br />';

echo 'Опишите их: ' . $alien_description . '<br />';

echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';

echo 'Ваш адрес электронной почты: ' . $email;

?>


</body>

</html>
```

Код этой строки считывает значение поля «Описание космических пришельцев» формы и присваивает его переменной $alien_description.

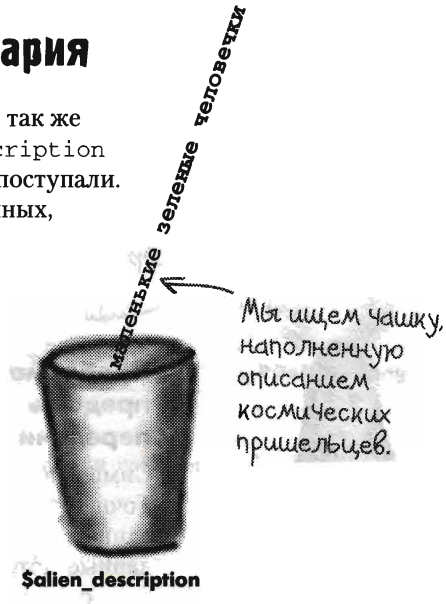
Код, объединяющий описание космических пришельцев с другим текстом и HTML-кодом и передающий результат браузеру.

По каким-то причинам переменная $alien_description оказалась пустой. Это плохо.

reportphp

Данные содержатся в переменных сценария

PHP-переменные — это контейнеры, в которых содержатся данные, так же как напиток содержится в чашке. Так как переменная `$alien_description` не содержит данных, мы знаем, что данные формы никогда туда не поступали. Таким образом, переменная `$alien_description` не содержит данных, несмотря на наши попытки присвоить ей значение.



Единственный способ исправить сценарий — это присвоить конкретное, ожидаемое значение переменной `$alien_description`, вот так:

```
$alien_description = 'маленькие зеленые человечки';
```

Знак равенства говорит PHP установить значение правой части для переменной в левой части.

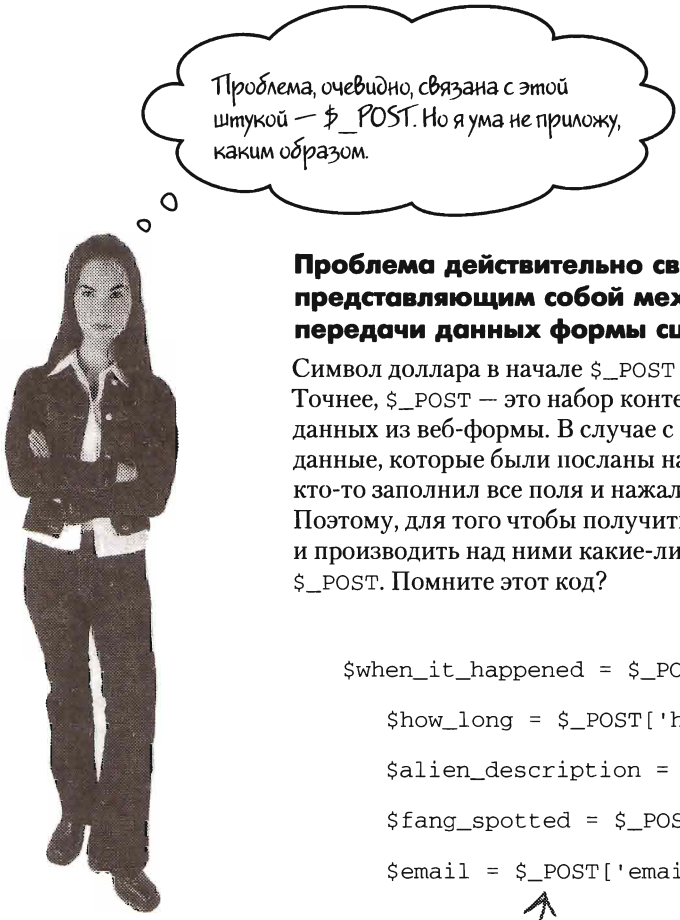
Фрагменты текста в PHP, также называемые строками, должны быть всегда ограничены кавычками — одинарными или двойными.

Этот код работает, так как он действительно присваивает текст `маленькие зеленые человечки'` переменной `$alien_description`. Но решив одну проблему, мы создали другую: использование такого кода приводит к тому, что описание космических пришельцев всегда будет одинаковым независимо от того, что посетитель сайта введет в форму.

Как-то получилось, что попытка присвоения переменной `$alien_description` значения описания космических пришельцев, взятого из данных формы, не удалась.

```
$alien_description = $_POST['description'];
```

Как вы думаете, что в этом коде неправильно?



Проблема действительно связана с $_POST, представляющим собой механизм, используемый для передачи данных формы сценарию.

Символ доллара в начале `$_POST` дает ключ... `$_POST` — это контейнер! Точнее, `$_POST` — это набор контейнеров, используемых для сбора данных из веб-формы. В случае с Оуэном эта переменная содержит все данные, которые были посланы нашему сценарию `report.php`, когда кто-то заполнил все поля и нажал кнопку «Сообщения о похищении». Поэтому, для того чтобы получить доступ ко всем данным формы и производить над ними какие-либо действия, мы должны пройти через `$_POST`. Помните этот код?

```
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$alien_description = $_POST['description'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
```

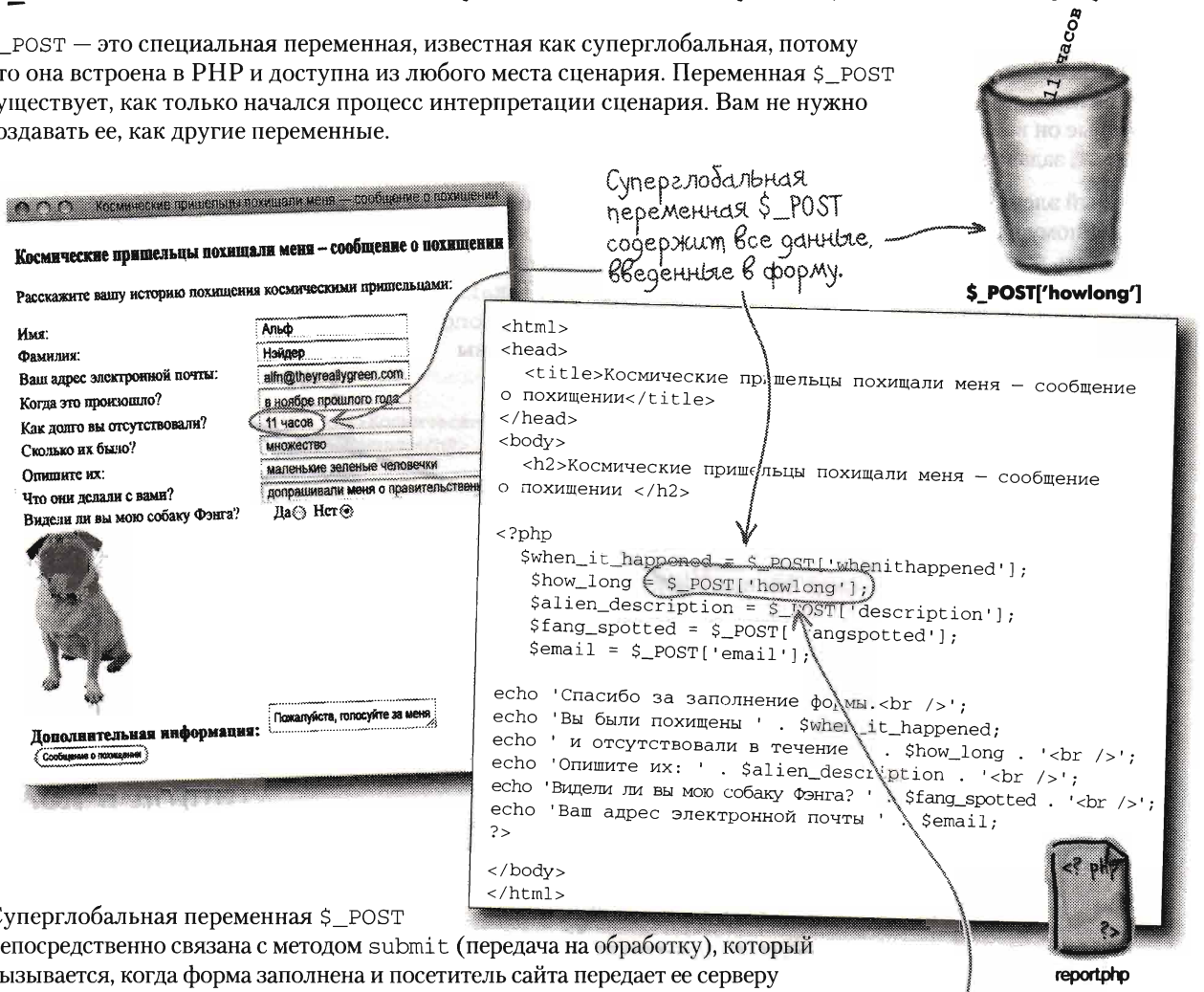
Часть данных формы, содержащая длительность похищения, присвоена переменной $how_long.

Аналогичная операция, только в этом случае данные формы, содержащие адрес электронной почты, были извлечены и присвоены переменной $email.

Таким образом, данные каждого поля формы «Сообщение о похищении» доступны благодаря `$_POST`. А что такое `$_POST` в действительности... переменная?

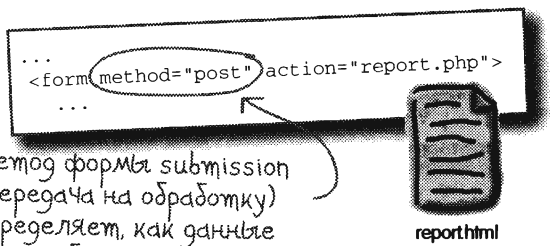
$_POST — специальная переменная, содержащая данные формы

$_POST — это специальная переменная, известная как суперглобальная, потому что она встроена в PHP и доступна из любого места сценария. Переменная $_POST существует, как только начался процесс интерпретации сценария. Вам не нужно создавать ее, как другие переменные.



Суперглобальная переменная $_POST непосредственно связана с методом submit (передача на обработку), который вызывается, когда форма заполнена и посетитель сайта передает ее серверу на обработку, нажав соответствующую кнопку. Если атрибуту method тега <form> установлено значение post, то все данные формы «упаковываются» в суперглобальную переменную $_POST, из которой они при необходимости могут быть извлечены для использования.

Имя «howlong» — это имя атрибута в теге <input> для этого поля формы.



Метод формы submission (передача на обработку) определяет, как данные формы будут переданы PHP-сценарию для обработки.

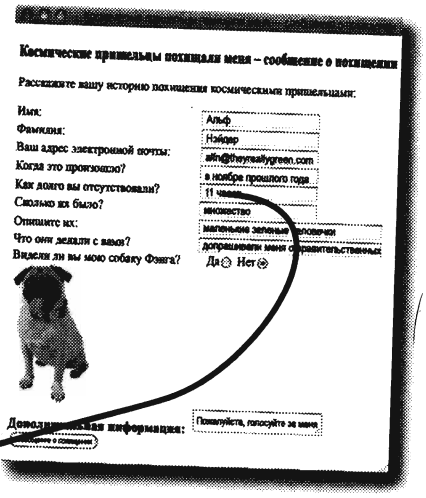
СИЛА МЫСЛИ

Что вы думаете по поводу того, как работает суперглобальная переменная $_POST? Каким образом она содержит множество значений, взятых из всех этих текстовых полей в форме Оуэна?

$_POST передает данные формы вашему сценарию

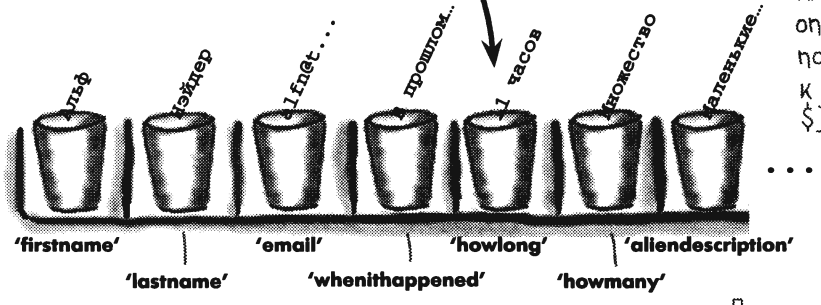
$_POST — это специальный вид PHP-контейнера, известный как **массив**, который содержит множество переменных под одним именем. Когда кто-нибудь передает форму Оуэна на обработку, все данные, которые он ввел накануне в поля формы, сохраняются в массиве $_POST, задача которого заключается в их передаче PHP-сценарию.

Каждый элемент массива $_POST соответствует значению, введенному в соответствующее поле формы. Для того чтобы получить доступ к данным определенного поля формы, необходимо использовать имя поля как указатель в массиве $_POST. Так, продолжительность похищения содержится в $_POST['howlong']. HTML-код формы Оуэна показывает, как имена полей связаны с данными, содержащимися в массиве $_POST.




```
<p>Расскажите вашу историю похищения космическими пришельцами:</p>
<form method="post" action="report.php">
   <label for="firstname">Имя:</label>
   <input type="text" id="firstname" name="firstname" /><br />
   <label for="lastname">Фамилия:</label>
   <input type="text" id=" lastname" name=" lastname" /><br />
   <label for="email">Ваш адрес электронной почты:</label>
   <input type="text" id=" email" name=" email" /><br />
   <label for="whenithappend">Когда это произошло?</label>
   <input type="text" id=" whenithappend" name=" whenithappend" /><br />
   <label for="howlong">Как долго вы отсутствовали?</label>
   <input type="text" id=" howlong" name=" howlong" /><br />
   <label for="howmany">Сколько их было?</label>
   <input type="text" id=" howmany" name=" howmany" /><br />
   <label for=" aliendescription">Опишите их:</label>
   <input type="text" id=" aliendescription" name=" aliendescription" size="32" /><br />
   <label for="whattheydid">Что они делали с вами?</label>
   <input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
   <label for="fangspotted">Видели ли вы мою собаку Фэнга?</label>
   Да <input id=" fangspotted" name=" fangspotted" type="radio" value="yes"/>
   Нет <input id=" fangspotted" name=" fangspotted" type="radio" value="no"/><br />
   <img src="fang.jpg" width="100" height="175"
      alt="Моя похищенная собака Фэнг.">
   <label for="other"> Дополнительная информация:</label>
   <textarea id=" other" name=" other" /></textarea><br />
   <input type="sabmit" value="Сообщение о похищении" name="submit" />
</form>
```

Массив $_POST заполнен данными, введенными посетителем сайта в поля формы.



Имя поля формы определяет, как можно получить доступ к данным в массиве $_POST.

Доступ ко всем данным формы осуществляется через массив $_POST.

Время поработать

Просмотрите внимательно сценарий Оуэна `report.php` и найдите код, который приводит к тому, что описание космических пришельцев отображается в виде пустой строки. Напишите, как исправить это.

Совет: используйте HTML-код с предыдущей страницы.

```
<html>
<head>
   <title>Космические пришельцы похищали меня — сообщение
о похищении</title>
</head>
<body>
   <h2>Космические пришельцы похищали меня — сообщение
о похищении</h2>

<?php
   $when_it_happened = $_POST['whenithappened'];
   $how_long = $_POST['howlong'];
   $alien_description = $_POST['description'];
   $fang_spotted = $_POST['fangspotted'];
   $email = $_POST['email'];

echo 'Спасибо за заполнение формы.<br />';
echo 'Вы были похищены ' . $when_it_happened;
echo ' и отсутствовали в течение ' . $how_long . '<br />';
echo 'Опишите их: ' . $alien_description . '<br />';
echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
echo 'Ваш адрес электронной почты ' . $email;
?>

</body>
</html>
```

Вспомните: ранее мы решили проблему для этих двух строк кода.

report.php

Решение задачи

Просмотрите внимательно сценарий Оуэна `report.php` и найдите код, который приводит к тому, что описание космических пришельцев отображается в виде пустой строки. Напишите, как исправить это.

Совет: используйте HTML-код с предыдущей страницы.

```
...
<input type="text" id="aliendescription" name="aliendescription" size="32" />
...
```

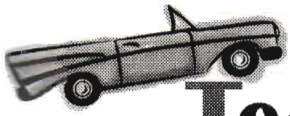
report.html

```
</head>
<body>
    <h2>Космические пришельцы похищали меня – сообщение
о похищении</h2>

<?php
    $when_it_happened = $_POST['whenithappened'];
    $how_long = $_POST['howlong'];
    $alien_description = $_POST['description'];
    $fang_spotted = $_POST['fangspotted'];
    $email = $_POST['email'];

echo 'Спасибо за заполнение формы.<br />';
echo 'Вы были похищены ' . $when_it_happened;
echo ' и отсутствовали в течение ' . $how_long . '<br />';
echo 'Опишите их: ' . $alien_description . '<br />';
echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
echo 'Ваш адрес электронной почты: ' . $email;
?>

</body>
</html>
```

report.php

Имя поля формы в report.html – «aliendescription», что не соответствует имени, используемому в $_POST.

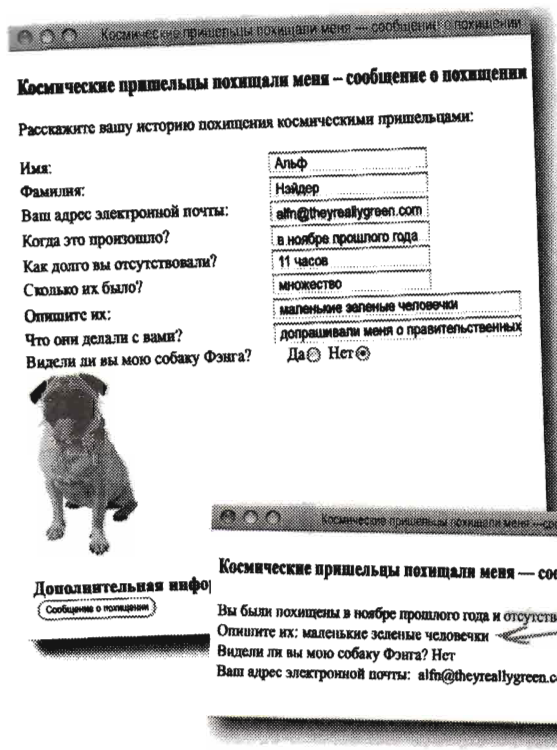
Нам необходимо изменить $_POST так, чтобы имя стало правильным: 'aliendescription'.



Тест-драйв

Исправьте сценарий и проверьте, как он работает.

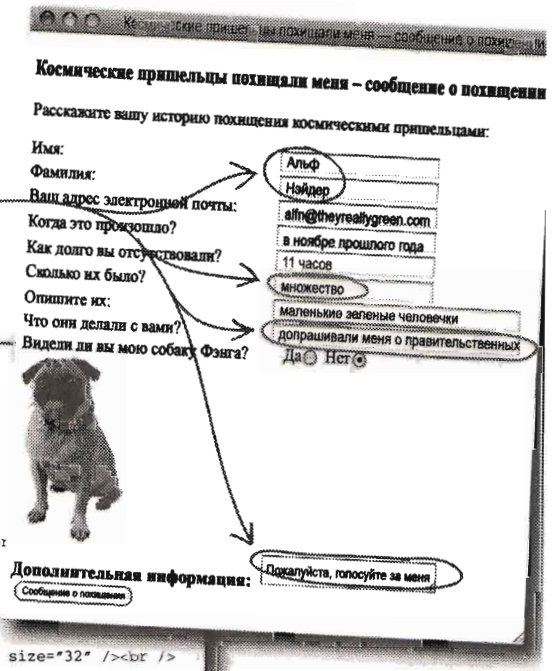
Измените ошибочную строку кода в файле `report.php` и загрузите его на ваш веб-сервер. Откройте веб-страницу `report.html` в вашем браузере, заполните поля формы информацией о похищении космическими пришельцами и нажмите кнопку «Сообщения о похищении», чтобы отправить форму на обработку.



Время поработать

В форме Оуэна «Сообщение о похищении» имеются некоторые данные, которые мы пока не использовали. Не забывайте, что они содержат жизненно важную информацию о похищении космическими пришельцами, которая может привести Оуэна к его потерянной собаке Фэнгу. Поэтому мы должны получить все данные о похищении и сохранить их в переменной `$_POST`.

В настоящий момент сценарий report.php игнорирует пять различных данных формы. Это никуда не годится!




```
<form method="post" action="report.php">
 <label for="firstname">Имя:</label>
 <input type="text" id="firstname" name="firstname" /><br />
 <label for="lastname">Фамилия:</label>
 <input type="text" id=" lastname" name=" lastname" /><br />
 <label for="email">Ваш адрес электронной почты:</label>
 <input type="text" id=" email" name=" email" /><br />
 <label for="whenithappend">Когда это произошло?</label>
 <input type="text" id=" whenithappend" name=" whenithappend" /><br
 <label for="howlong">Как долго вы отсутствовали?</label>
 <input type="text" id=" howlong" name=" howlong" /><br />
 <label for="howmany">Сколько их было?</label>
 <input type="text" id=" howmany" name=" howmany" /><br />
 <label for=" aliendescription">Опишите их:</label>
 <input type="text" id=" aliendescription" name=" aliendescription" size="32" /><br />
 <label for="whattheydid">Что они делали с вами?</label>
 <input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
 <label for="fangspotted">Видели ли вы мою собаку Фэнга?</label>
 Да <input id=" fangspotted" name=" fangspotted" type="radio" value="yes"/>
 Нет <input id=" fangspotted" name=" fangspotted" type="radio" value="no"/><br />
 <img src="fang.jpg" width="100" height="175"
    alt="Моя похищенная собака Фэнг.">
 <label for="other"> Дополнительная информация:</label>
 <textarea id=" other" name=" other" /></textarea><br />
 <input type="submit" value="Сообщение о похищении" name="submit" />
 </form>
</body>
</html>
```

Тег `<input>` для каждого поля данных формы содержит ключ для доступа к этим данным из PHP.

report.html

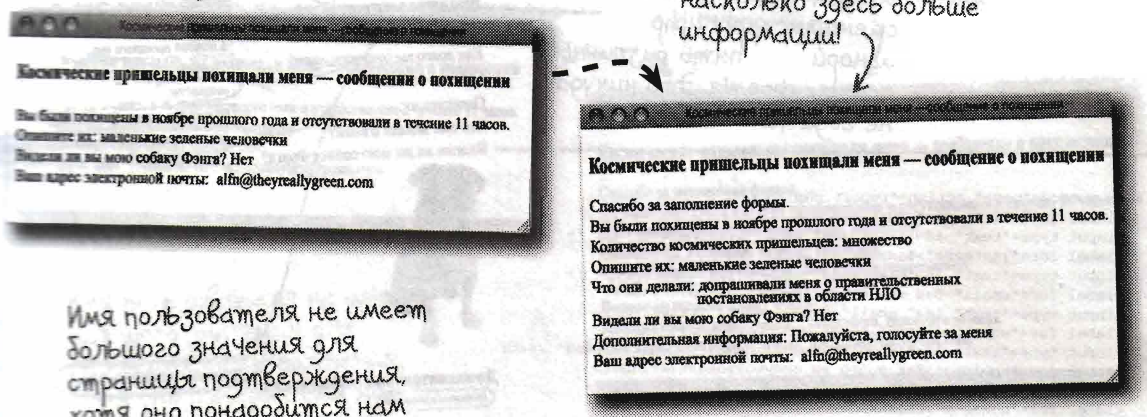
Напишите PHP-код, создающий четыре новые переменные, которые будут содержать недостающие данные формы: `$name`, `$how_many`, `$what_they_did` и `$other`.

Совет: создайте переменную `$name` так, чтобы она содержала имя и фамилию посетителя сайта.

Ваша работа еще не закончена. Веб-странице подтверждения, генерируемой PHP-сценарием, необходимы эти новые переменные, чтобы воспроизвести дополнительную информацию о похищении космическими пришельцами.

Нам нужно перейти от этого...

...к этому! Видите, насколько здесь больше информации!



Имя пользователя не имеет большого значения для страницы подтверждения, хотя оно понадобится нам позднее, когда мы будем посылать Оуэну сообщение по электронной почте.

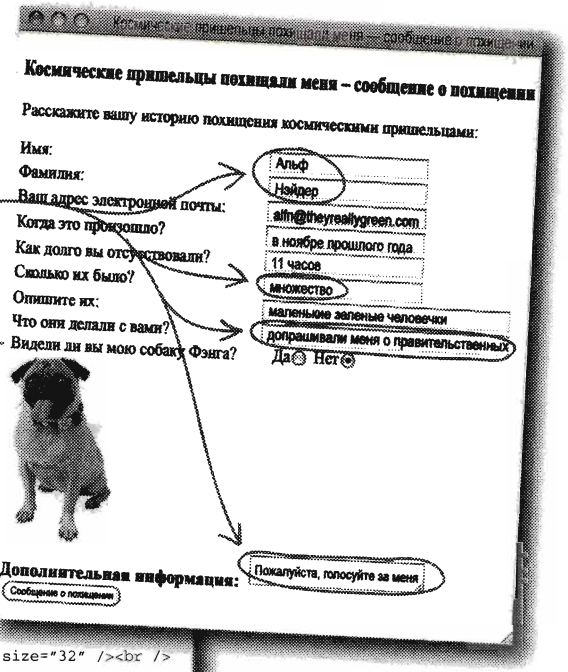
Используя все переменные, которые мы только что создали, за исключением `$name`, допишите ниже пропущенный код, который генерирует более информативную веб-страницу подтверждения.

```
echo 'Спасибо за заполнение формы.<br />';
echo 'Вы были похищены ' . $when_it_happened;
echo ' и отсутствовали в течение ' . $how_long . '<br />';
  .....
echo 'Опишите их: ' . $alien_description . '<br />';
                  .....    .....                    .
echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
                        ...                          ..
echo 'Ваш адрес электронной почты: ' . $email;
```

Решение задачи

В форме Оуэна «Сообщение о похищении» имеются некоторые данные, которые мы пока не использовали. Не забывайте, что они содержат жизненно важную информацию о похищении космическими пришельцами, которая может привести Оуэна к его потерянной собаке Фэнгу. Поэтому мы должны получить все данные о похищении и сохранить их в переменной `$_POST`.

В настоящий момент сценарий report.php игнорирует пять различных данных формы. Это никуда не годится!




```
...
<form method="post" action="report.php">
    <label for="firstname">Имя:</label>
    <input type="text" id="firstname" name="firstname" /><br />
    <label for="lastname">Фамилия:</label>
    <input type="text" id=" lastname" name=" lastname" /><br />
    <label for="email">Ваш адрес электронной почты:</label>
    <input type="text" id=" email" name=" email" /><br />
    <label for="whenithappend">Когда это произошло?</label>
    <input type="text" id=" whenithappend" name=" whenithappend" /><br />
    <label for="howlong">Как долго вы отсутствовали?</label>
    <input type="text" id=" howlong" name=" howlong" /><br />
    <label for="howmany">Сколько их было?</label>
    <input type="text" id=" howmany" name=" howmany" /><br />
    <label for=" aliendescription">Опишите их:</label>
    <input type="text" id=" aliendescription" name=" aliendescription" size="32" /><br />
    <label for="whattheydid">Что они делали с вами?</label>
    <input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
    <label for="fangspotted">Видели ли вы мою собаку Фэнга?</label>
    Да <input id=" fangspotted" name=" fangspotted" type="radio" value="yes"/>
    Нет <input id=" fangspotted" name=" fangspotted" type="radio" value="no"/><br />
    <img src="fang.jpg" width="100" height="175"
       alt="Моя похищенная собака Фэнг.">
    <label for="other"> Дополнительная информация:</label>
    <textarea id=" other" name=" other" /></textarea><br />
    <input type="submit" value="Сообщение о похищении" name="submit" />
</form>
</body>
</html>
```

Тег `<input>` для каждого поля данных формы содержит ключ для доступа к этим данным из PHP.

report.html

Напишите PHP-код, создающий четыре новые переменные, которые будут содержать недостающие данные формы: `$name`, `$how_many`, `$what_they_did` и `$other`.

Совет: Создайте переменную `$name` так, чтобы она содержала имя и фамилию посетителя сайта.

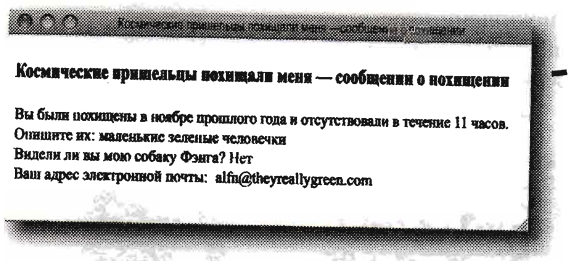
Этот пробел отделяет имя от фамилии.

Оператор «точка» позволяет вам соединять между собой несколько строк текста. Такая операция называется конкатенацией.

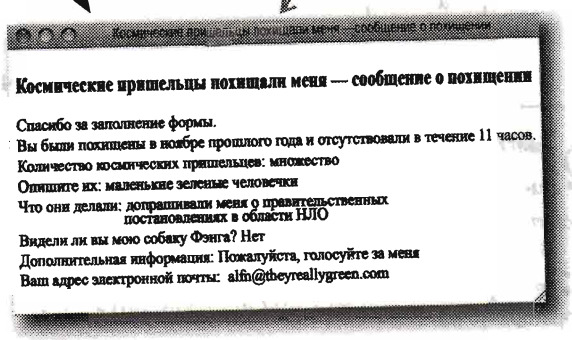
```
$name = $_POST['firstname'] . ' ' . $_POST['lastname'];
$how_many = $_POST['howmany'];
$what_they_did = $_POST['whattheydid'];
$other = $_POST['other'];
```

Ваша работа еще не закончена. Веб-странице подтверждения, генерируемой PHP-сценарием, необходимы эти новые переменные, чтобы воспроизвести дополнительную информацию о похищении космическими пришельцами.

Нам нужно перейти от этого...



...к этому! Видите, насколько здесь больше информации!



Имя пользователя не имеет большого значения для страницы подтверждения, хотя оно понадобится нам позднее, когда мы будем посылать Оуэну сообщение по электронной почте.

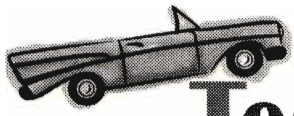
Используя все переменные, которые мы только что создали, за исключением `$name`, допишите ниже пропущенный код, который генерирует более информативную веб-страницу подтверждения.

Тег `<br />` помогает в форматировании информации — не забывайте, что мы используем PHP для создания HTML.

Команда echo используется для передачи браузеру дополнительной информации как HTML-контента.

Снова оператор «точка» используется для соединения вместе строк текста и переменных.

```
echo 'Спасибо за заполнение формы.<br />';
echo 'Вы были похищены ' . $when_it_happened;
echo ' и отсутствовали в течение ' . $how_long . '<br />';
echo 'Сколько их было?' . $how_many . '<br />';
echo 'Опишите их: ' . $alien_description . '<br />';
echo 'Что они делали с вами?' . $what_they_did . '<br />';
echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
echo 'Дополнительная информация:' . $other . '<br />';
echo 'Ваш адрес электронной почты: ' . $email;
```



Тест-драйв

Исправьте сценарий Оуэна и проверьте, как он работает.

Добавьте код для новых переменных в файл `report.php`, а также код, который передает значения этих переменных в виде отформатированного HTML. Затем загрузите сценарий на свой веб-сервер, откройте страницу `report.html` в браузере и занесите в форму информацию о похищении космическими пришельцами. Теперь нажмите кнопку «Сообщения о похищении» для передачи формы на обработку и наблюдайте результат.

не бывает глупых вопросов

В: Что фактически происходит, когда я объединяю вместе несколько строк текста, используя точки?

О: Результатом объединения нескольких строк с помощью оператора «точка» (.) является одна совершенно новая строка. При этом не имеет значения, сколько отдельных строк первоначально принимали участия в такой операции. Поэтому, когда объединяете несколько строк как часть команды `echo`, PHP вначале соединяет все строки в одну, а затем передает ее браузеру.

В: Когда я объединяю с помощью оператора «точка» (конкатенирую) переменную со строкой текста, должна ли переменная содержать текст?

О: Нет. Хотя результатом конкатенации всегда является строка текста, переменные не обязательно должны содержать текст для выполнения этой операции. Если, скажем, переменная содержит число, PHP сначала конвертирует это число в строку, а затем произведет саму конкатенацию.

В: Что происходит с PHP-кодом в браузере?

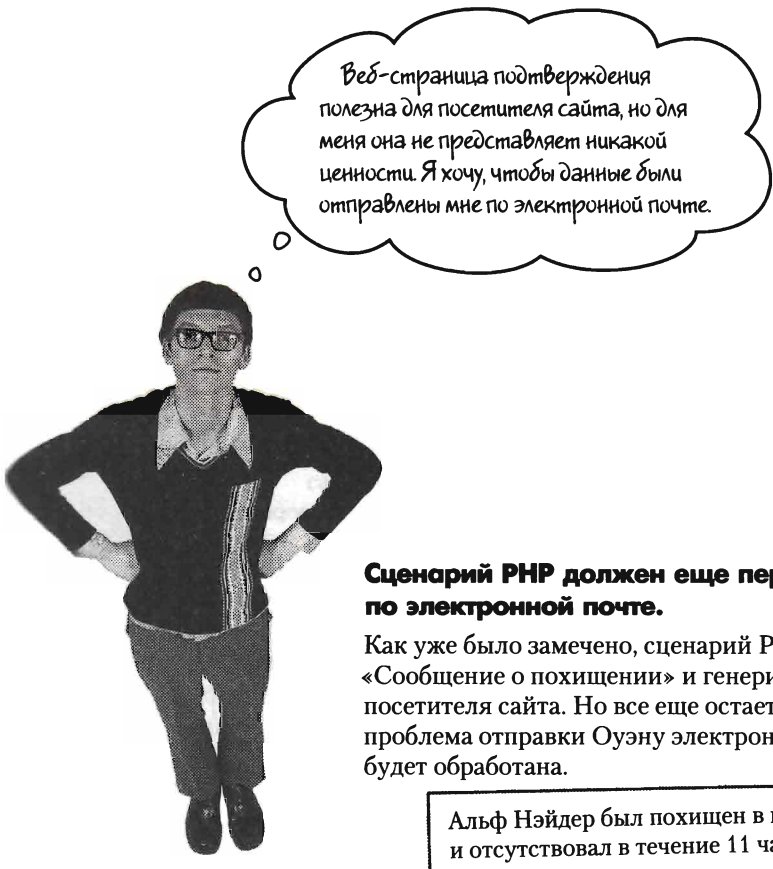
О: Ничего, потому что браузер не понимает PHP-кода. PHP-код интерпретируется на сервере и преобразуется в HTML-код, который и передается браузеру. Поэтому браузер совершенно не знает о существовании PHP. Веб-страницы поступают на него в виде чистого HTML- и CSS-кода.

В: Хорошо, но как сервер преобразует PHP-код в HTML и CSS?

О: Во-первых, не забывайте, что по умолчанию действует принцип, согласно которому код PHP-сценария должен в конечном счете стать HTML-кодом. Вы отделяете PHP-код от остального текста в сценарии, размещая его между тегами `<?php` и `?>`. Сервер видит эти теги и интерпретирует код внутри них как PHP, а весь код за пределами этих тегов передает браузеру как HTML, без изменения.

В: Хорошо, но это все еще не объясняет, как именно PHP-код преобразуется в HTML/CSS-код. Что происходит?

О: Вот здесь как раз и вступает в игру команда `echo`. Вы можете рассматривать ее как средство вывода информации за пределы границ, определенных тегами `<?php` и `?>`. Поэтому команда `echo` играет ключевую роль в возможности PHP динамически генерировать HTML/CSS-код. Объединяя строки текста с переменными PHP, вы можете конструировать HTML-код на лету, а затем использовать команду `echo`, чтобы передавать его браузеру как часть результирующей веб-страницы. Хорошим примером этого является сценарий Оуэна `report.php`, где тег `<br />` добавлен к концу текста, чтобы генерировать перевод строки в HTML.



Сценарий PHP должен еще переслать Оуэну данные формы по электронной почте.

Как уже было замечено, сценарий PHP считывает данные из формы «Сообщение о похищении» и генерирует страницу подтверждения для посетителя сайта. Но все еще остается нерешенной первоначальная проблема отправки Оуэну электронного письма, после того как форма будет обработана.

Аналогично веб-странице подтверждения, это электронное письмо содержит статичный текст и данные формы.

Альф Нэйдер был похищен в ноябре прошлого года и отсутствовал в течение 11 часов.

Количество космических пришельцев: множество.

Описание космических пришельцев: маленькие зеленые человечки.

Что они делали? Допрашивали меня о правительственных постановлениях в области НЛО.

Фэнг замечен? Нет.

Дополнительная информация: Пожалуйста, голосуйте за меня.

Это электронное письмо может быть сгенерировано PHP-кодом путем объединения строк статического текста, таких как «`Дополнительная информация:`», с данными формы, содержащимися в переменных.

Напишите, как бы вы объединили строки статического текста с PHP-переменными для составления электронного письма.

Построение содержания электронного письма в PHP

Вы уже видели, как оператор «точка» может быть использован для конкатенации (объединения) нескольких строк текста в одну. Сейчас нам необходимо использовать конкатенацию вновь, чтобы создать содержательную часть электронного письма, расставляя значения переменных в различные места статичного текста.

Большинство текстовых редакторов при достижении текстом правого края автоматически переводят его на новую строку, даже если вы не вводите символ перевода строки (клавиша «Ввод», или «Enter»).

С помощью оператора «точка» значения переменных и статичный текст объединяются в строку содержательной части электронного письма.

```
$msg = $name . ' был похищен ' . $when_it_happened . ' и отсутствовал в течение ' . $how_long . '.'
'Количество космических пришельцев: ' . $how_many . 'Описание космических пришельцев: ' .
$alien_description . 'Что они делали?' .
$what_they_did . 'Фэнг замечен? ' . $fang_spotted . 'Дополнительная информация: ' . $other;
```

Не забывайте: каждая переменная содержит строку текста, извлеченного из формы «Сообщение о похищении».

Одна из проблем заключается в том, что требуется составить длинную строку PHP-кода, который трудно читать и понимать. Вы можете распределить PHP-код между несколькими строками, чтобы сделать его более легким для прочтения и понимания. Только следите за тем, чтобы разбивать строку на фрагменты в тех местах, где ввод дополнительных разделителей (в данном случае перевод строки) не имеет значения (игнорируется PHP), например **между** двумя соединяемыми строками текста, а не в середине строки. Поставьте точку с запятой в конце последней строки, чтобы завершить PHP-предложение.

Эта длинная строка кода распределена между несколькими короткими.

```
$msg = $name . ' был похищен ' . $when_it_happened . ' и отсутствовал в течение ' . $how_long . '.'
  'Количество космических пришельцев: ' . $how_many .
  'Описание космических пришельцев: ' . $alien_description .
  'Что они делали?' . $what_they_did .
  'Фэнг замечен? ' . $fang_spotted .
  'Дополнительная информация: ' . $other;
```

Необходимо следить за тем, чтобы при распределении длинной строки кода между несколькими короткими перевод на новую строку не производился в середине фразы.

Нужно поставить точку с запятой в конце всего PHP-предложения.

Когда длинная строка PHP-кода распределяется между несколькими короткими, принято смещать все последующие строки относительно первой, чтобы было лучше видно, какие короткие строки принадлежат одной длинной.

Длинная строка PHP-кода может быть разделена между несколькими более короткими, но необходимо соблюдать осторожность при определении мест, куда нужно установить дополнительные разделители.

Этот PHP-код действительно очень хорошо выглядит. Но без надлежащего форматирования не будет ли электронное письмо выглядеть беспорядочным нагромождением текста?

Да. Но то, что PHP-код хорошо упорядочен, совершенно не значит, что результат его интерпретации будет так же хорошо упорядочен автоматически.

Организация PHP-кода с целью сделать его более легким для прочтения и понимания **для вас** — совершенно не то же самое, что форматирование результата его интерпретации, который будут наблюдать посетители вашего сайта. Обычно для такого форматирования вы применяете теги HTML, так как в большинстве случаев PHP используется для динамического генерирования веб-страниц. Но это другой случай.

Здесь мы генерируем электронное письмо, которое является простым текстом, а не HTML. Мы должны принять во внимание тот факт, что пока это письмо будет выглядеть так:

Альф Нэйдер был похищен в ноябре прошлого года и отсутствовал в течение 11 часов.Количество космических пришельцев: множествоОписание космических пришельцев: маленькие зеленые человечкиЧто они делали? допрашивали меня о правительственных постановлениях в области НЛО Фэнг замечен? нетДополнительная информация: Пожалуйста, голосуйте за меня.

Ой! Но это совсем НЕ ТО, что Оуэн предполагал получить в качестве электронного письма о похищении космическими пришельцами!

Не бывает глупых вопросов

В: Существует ли способ использовать форматирование HTML в электронных письмах, которые вы посылаете из PHP-сценария?

О: Существует. Но этот метод требует следующего шага, который касается установки поля `content type` (`тип контента`) в заголовке электронного письма. Вы узнаете больше об этих заголовках в главе 6 и тогда сможете вернуться к вопросам, связанным с электронной почтой в HTML-формате.

Сила мысли

Как бы вы переделали электронное письмо в формате простого текста, чтобы оно выглядело удобнее для прочтения?

Даже простой текст может быть отформатирован... немного

Раз Оуэн отправляет электронное письмо в виде простого текста без HTML-форматирования, он не может использовать тег `<br />` для перевода строки в местах, где возникает необходимость разбить длинную строку на несколько коротких. Но он может использовать символ новой строки в форме Escape-последовательности (экранированной последовательности) `'\n'`. Если в электронном письме встретится этот символ, весь текст после него будет выводиться с новой строки. Вот новый код электронного письма с добавленными символами новой строки:

Escape-последовательности в PHP начинаются с символа «обратная наклонная черта» (\).

`'\n'` используется для того, чтобы вставить символ новой строки в текст электронного письма.

```
$msg = $name . ' был похищен ' . $when_it_happened . ' и отсутствовал в течение ' . $how_long . '.\n'
'Количество космических пришельцев: ' . $how_many . '\n' .
'Описание космических пришельцев: ' . $alien_description . '\n'
'Что они делали?' . $what_they_did . '\n'
'Фэнг замечен? ' . $fang_spotted . '\n' .
'Дополнительная информация: ' . $other;
```

Символ новой строки выглядит заманчиво. Жаль только, что код с его использованием не работает.

Альф Нэйдер был похищен в ноябре прошлого года и отсутствовал в течение 11 часов\nКоличество космических пришельцев: множество\nОписание космических пришельцев: маленькие зеленые человечки\nЧто они делали? допрашивали меня о правительственных постановлениях в области НЛО\nФэнг замечен?нет\nДополнительная информация: Пожалуйста, голосуйте за меня.

Вместо того чтобы переводить строку, `'\n'` появляется в тексте как обычная последовательность символов.

Не бывает глупых вопросов

В: Что в действительности представляет собой Escape-последовательность?

О: Escape-последовательность (экранированная последовательность) позволяет добавить символ, который трудно ввести (клавиша с таким символом отсутствует на клавиатуре) или интерпретация которого может вызвать конфликт с другим кодом PHP. Вы, возможно, уже знакомы с Escape-последовательностями в HTML, где они кодируются несколько иначе, например: `&#169;` или `&copy;` для символа копирайта. В PHP набор Escape-последовательностей невелик: например, символ одинарной кавычки (`\'`), двойной кавычки (`\"`) и, конечно, новой строки (`\n`).

Символы новой строки необходимо помещать в строки, ограниченные двойными кавычками

Проблема с кодом Оуэна заключается в том, что PHP обрабатывает строки по-разному, в зависимости от того, заключены они в одинарные или двойные кавычки. Символ новой строки (`\n`) будет рассматриваться как экранированная последовательность, если он помещен в строку, ограниченную двойными кавычками. Поэтому, для того чтобы символ новой строки интерпретировался именно как символ новой строки, а не как последовательность символов обратной наклонной черты (`\`) и английской буквы `n`, текст сообщения о похищениях, пересылаемого по электронной почте, должен создаваться с использованием строк текста, заключенных в двойные кавычки.

Но это еще не вся разница между одинарными и двойными кавычками. Строка, ограниченная одинарными кавычками, рассматривается PHP как «сырой» текст, в то время как в строках, ограниченных двойными кавычкам, и переменные рассматриваются особенным образом. Когда PHP встречает переменную в строке, ограниченной двойными кавычками, он заменяет имя этой переменной на ее значение. Поэтому использование двойных кавычек не только заставляет работать символ перевода строки, но и упрощает код, позволяя вставлять имена переменных непосредственно в строки текста.

```
$msg = "$name был похищен $when_it_happened и отсутствовал в течение $how_long.\n" .
"Количество космических пришельцев: $how_many\n" .
"Описание космических пришельцев: $alien_description\n" .
"Что они делали? $what_they_did\n" .
" Фэнг замечен? $fang_spotted\n" .
"Дополнительная информация: $other";
```

Необходимость в операции конкатенации отпадает, так как переменные могут быть размещены непосредственно в тексте, ограниченном двойными кавычками.

Символ новой строки теперь интерпретируется правильно благодаря двойным кавычкам.

Но нам все равно необходимо распределить сообщение на несколько соединенных между собой строк, чтобы код было удобнее читать.

В самом конце нет необходимости ставить символ новой строки, так как это последняя строка электронного письма.

Не бывает глупых вопросов

В: Если двойные кавычки такие удобные, почему мы до сих пор использовали в основном одинарные?

О: Примите во внимание тот факт, что строки, заключенные в одинарные кавычки, никак не обрабатываются PHP, что делает их идеальными для использования с простым текстом, то есть не содержащим переменных. Поэтому мы продолжим использовать одинарные кавычки на протяжении всей книги, если только не возникнет серьезных причин использовать двойные. Наиболее важным соображением относительно использования одинарных или двойных кавычек является стремление быть настолько последовательным, насколько это возможно.

В: Итак, одинарные кавычки поддерживают `\'`, но не поддерживают `\n`. Как мне определить, какую экранирующую последовательность я могу использовать в строке текста, ограниченного одинарными кавычками?

О: Одинарные кавычки разрешают только экранирующие последовательности `\'` и `\\`. Все остальные могут быть использованы лишь в строках текста, ограниченных двойными кавычками.

Скомпонуйте электронное письмо для Оуэна

С содержанием письма, сгенерированным в виде строки текста, вы можете двигаться дальше в компоновке остальных частей электронного письма для Оуэна. Сообщение электронной почты больше, чем просто содержание: имеется еще несколько его составляющих. Хотя некоторые из них не являются обязательными, следующая информация используется практически во всех электронных письмах.

Уже сделано!

1. ~~Содержание электронного письма.~~
2. **Тема электронного письма.**
3. **Адрес электронной почты отправителя (ОТ КОГО).**
4. **Адрес электронной почты получателя (КОМУ).**

Все, что вы посчитаете нужным поместить сюда, Оуэн увидит в графе «Тема» своей почтовой программы.

Адрес электронной почты посетителя сайта.

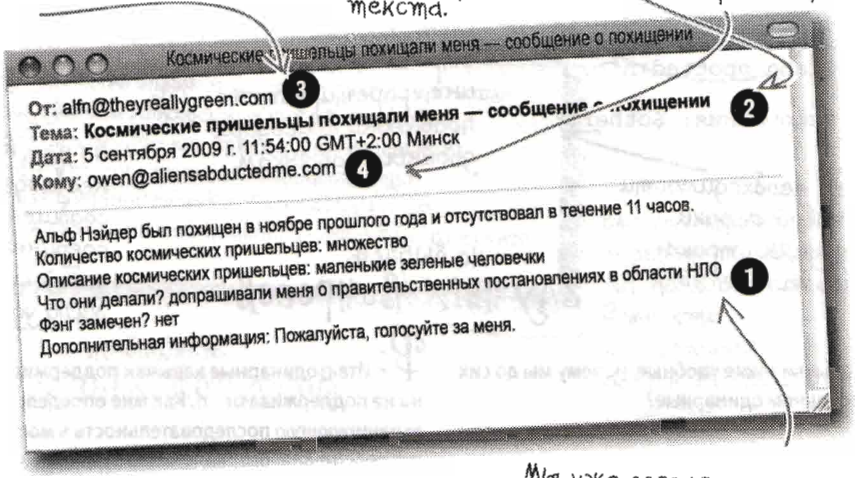
Адрес электронной почты Оуэна.

Электронное письмо примерно такого вида Оуэн надеется получить, после того как кто-нибудь отправит форму на сервер.

Это адрес электронной почты посетителя сайта, который сохранен в переменной $email.

Это может быть строка простого текста.

Это адрес электронной почты Оуэна, который также может быть строкой простого текста.

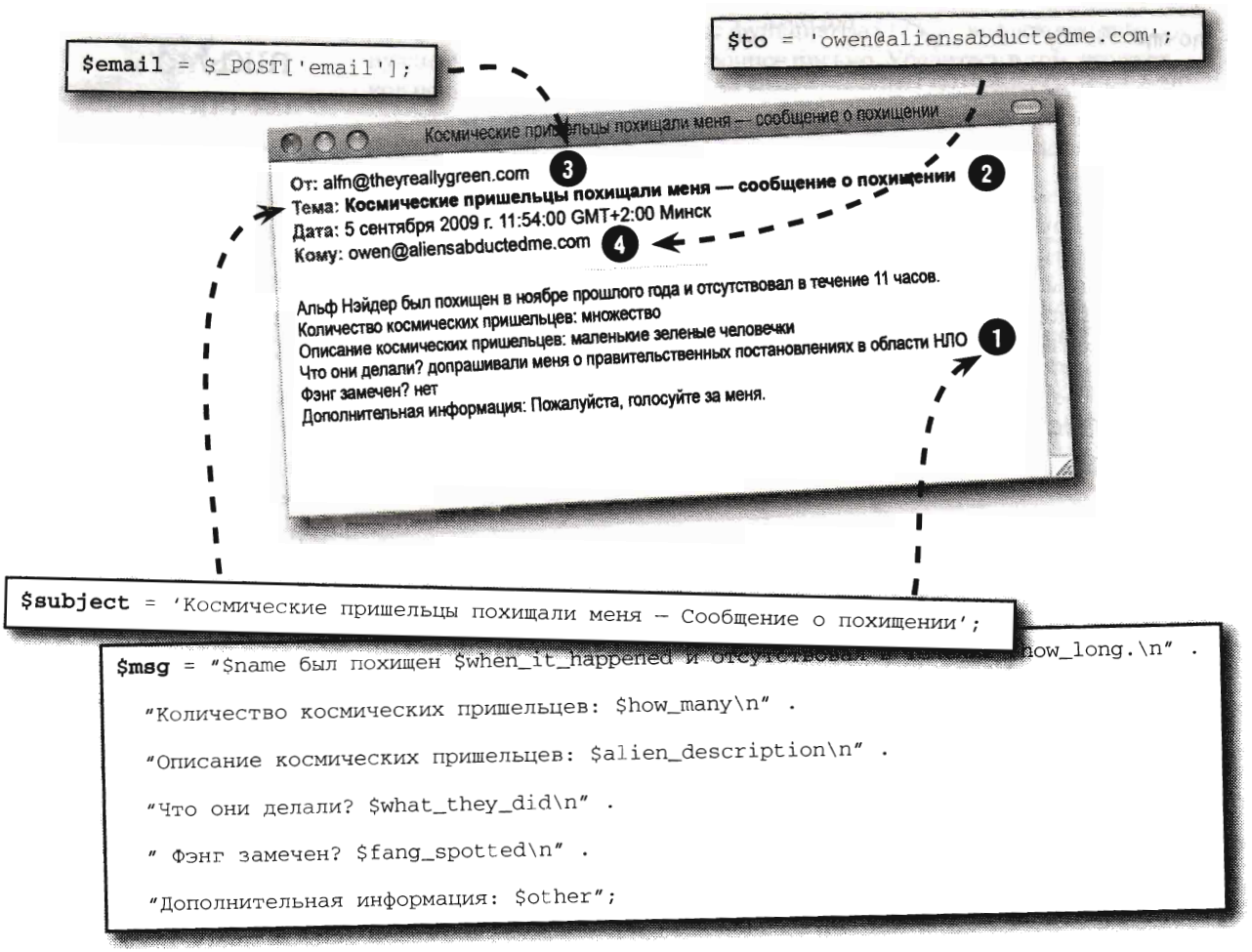


Мы уже создали строку с содержанием электронного письма, которая присвоена переменной $msg.

Этот пример электронного письма показывает, что большая часть контента находится в его содержании, которое мы уже закончили. Все, что осталось, — это добавить тему, адреса электронной почты «кому» и «от кого»... и, конечно, как-то отправить его, используя PHP!

Разделы электронного письма содержатся в переменных

У нас уже есть содержание электронного письма в переменной `$msg`, но все еще нет темы сообщения и адресов электронной почты «кому» и «от кого». Тема сообщения и адрес электронной почты «кому» могут быть присвоены новым переменным как строки простого текста, в то время как адрес электронной почты посетителя сайта «от кого» уже сохранен в переменной `$email` благодаря коду обработки формы, который мы написали ранее, читая эту главу.



Вся информация для электронного письма собрана и готова к отправке!

1. ~~Содержание электронного письма.~~
2. ~~Тема электронного письма.~~
3. ~~Адрес электронной почты отправителя (ОТ КОГО).~~
4. ~~Адрес электронной почты получателя (КОМУ).~~

Отправка электронного письма с помощью PHP

Итак, вы готовы написать PHP-код, который *отправит* электронное письмо Оуэну. Для этого необходима встроенная PHP-функция `mail()`, отправляющая электронное письмо, основываясь на информации, которую вы ей предоставляете.

PHP-функция mail() отправляет электронное письмо из сценария.

Адрес электронной почты «кому».

Тема электронного письма.

```
mail($to, $subject, $msg);
```

Содержание электронного письма.

Эти три фрагмента информации являются обязательными при вызове функции `mail()`, поэтому вы всегда должны их предоставлять. Адрес электронной почты «от кого» не является обязательным, но его рекомендуется включать. При вызове функции `mail()` с указанием этого адреса аргумент, передаваемый функции `mail()`, обязательно должен иметь в своем начале слово «`From: `».

Адресу электронной почты отправителя, передаваемому в качестве аргумента функции mail(), обязательно должно предшествовать слово 'From: '.

```
mail($to, $subject, $msg, 'From:' . $email);
```

Оператор «точка» опять используется для конкатенации слова 'From: ' и адреса электронной почты Оуэна.

```
$to = 'owen@aliensabductedme.com';
$subject = 'Космические пришельцы похищали меня – Сообщение о похищении';
$msg = "$name был похищен $when_it_happened и отсутствовал в течение $how_long.\n" .
"Количество космических пришельцев: $how_many\n" .
"Описание космических пришельцев: $alien_description\n" .
"Что они делали? $what_they_did\n" .
"Фэнг замечен? $fang_spotted\n" .
"Дополнительная информация: $other";
$email = $_POST['email'];
```

Каждая часть сообщения электронной почты передается функции mail() как переменная.

Не бывает глупых вопросов

В: Имеется ли еще какая-нибудь информация, которая может быть определена как часть электронного письма, кроме адреса электронной почты «от кого»?

О: Да. Вы можете определить поля «копия» и «слепая копия» таким же способом, как и в случае с адресом электронной почты «от кого». Используйте просто `'Cc: '` или `'Bcc: '` вместо `'From: '`. Если вы хотите определить как поле «от кого», так и поле «копия», вы должны отделить их друг от друга комбинацией символов «возврат каретки» и «новая строка», как показано ниже:

Экранировать символы один за другим — это правильный способ.

```
"From: " . $from  "\r\nCc  " . $cc
```

Мы используем двойные кавычки, так как применяем экранированные последовательности \r и \n.

Просто добавьте в свой сценарий код, который вызывает mail().

Строка кода, который вызывает функцию `mail()`, — вот все, что вам нужно, чтобы отправить электронное письмо. Убедитесь в том, что этот код появляется в сценарии после кода, который создает все переменные, связанные с электронной почтой, и все будет в порядке. Ниже приведен полный код сценария Оуэна `report.php`, включающий вызов функции `mail()`.

```
<html>
<head>
   <title>Космические пришельцы похищали меня – сообщение о похищении</title>
</head>
<body>
   <h2>Космические пришельцы похищали меня – сообщение
о похищении</h2>

<?php
   $name = $_POST['firstname'] . ' ' . $_POST['lastname'];
   $when_it_happened = $_POST['whenithappened'];
   $how_long = $_POST['howlong'];
   $how_many = $_POST['howmany'];
   $alien_description = $_POST['aliendescription'];
   $what_they_did = $_POST['whattheydid'];
   $fang_spotted = $_POST['fangspotted'];
   $email = $_POST['email'];
   $what_they_did = $_POST['whattheydid'];
   $other = $_POST['other'];

   $to = 'owen@aliensabductedme.com';
   $subject = 'Космические пришельцы похищали меня – сообщение о похищении';
   $msg = "$name был похищен $when_it_happened и отсутствовал в течение $how_long.\n"
    "Количество космических пришельцев: $how_many\n" .
    "Описание космических пришельцев: $alien_description\n" .
    "Что они делали? $what_they_did\n" .
    "Фэнг замечен? $fang_spotted\n" .
    "Дополнительная информация: $other";
   mail($to, $subject, $msg, 'From: ' . $email);

   echo 'Спасибо за заполнение формы.<br />';
   echo 'Вы были похищены ' . $when_it_happened;
   echo ' и отсутствовали в течение ' . $how_long . '<br />';
   echo 'Количество космических пришельцев: ' . $how_many . '<br />';
   echo 'Опишите их: ' . $alien_description . '<br />';
   echo 'Что они делали? ' . $what_they_did . '<br />';
   echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
   echo 'Дополнительная информация: ' . $other . '<br />';
   echo 'Ваш адрес электронной почты: ' . $email;
?>

</body>
</html>
```

Извлекает все данные формы из массива $_POST и присваивает их индивидуальным переменным.

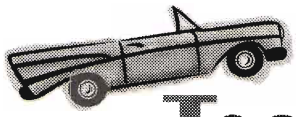
Убедитесь в том, что вы заменили адрес электронной почты на свой, перед тем как тестировать этот сценарий.

Отправка электронного письма.

Определяет отдельные части электронного письма перед тем, как отправить его Оуэну.

Создание на лету HTML-страницы, которая подтверждает успешную отправку формы на сервер.

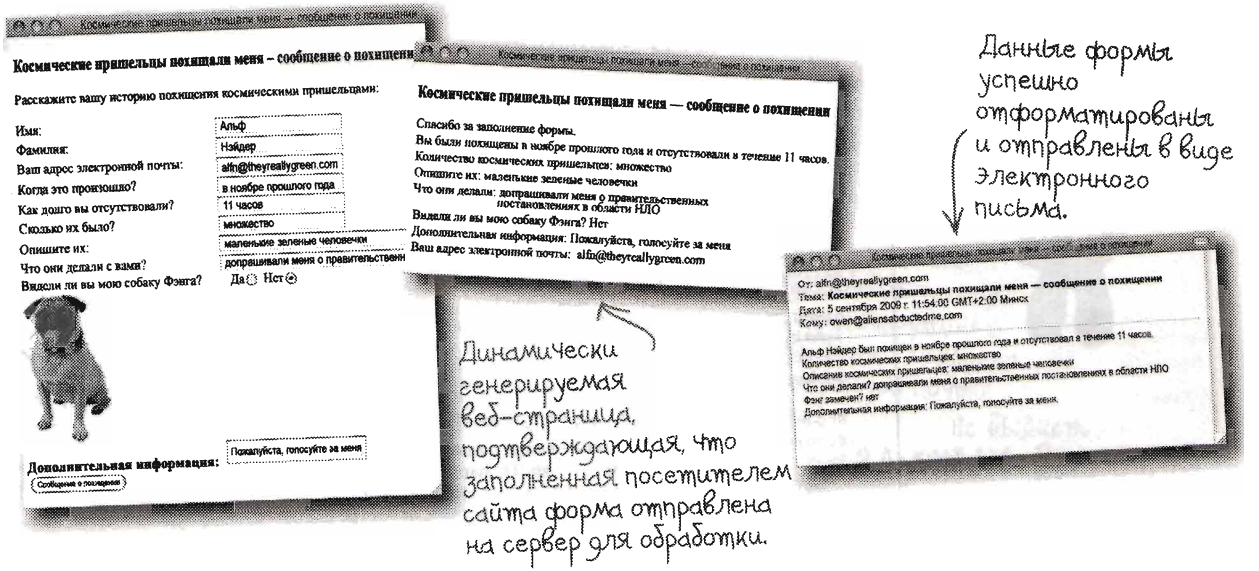
reportphp



Тест-драйв

Завершите сценарий Оуэна и испытайте его.

Добавьте три новые переменные, связанные с электронной почтой (`$to`, `$subject` и `$msg`), в сценарий `report.php`, а также вызов функции `mail()`. Убедитесь в том, что переменная `$to` содержит ваш адрес электронной почты, а не адрес Оуэна. Загрузите сценарий на ваш веб-сервер, откройте его в своем браузере и введите в форму информацию о похищении космическими пришельцами. Нажмите кнопку «Сообщение о похищении», чтобы отправить форму на сервер. Подождите несколько секунд и проверьте ваш электронный почтовый ящик на наличие письма.



Запомни!

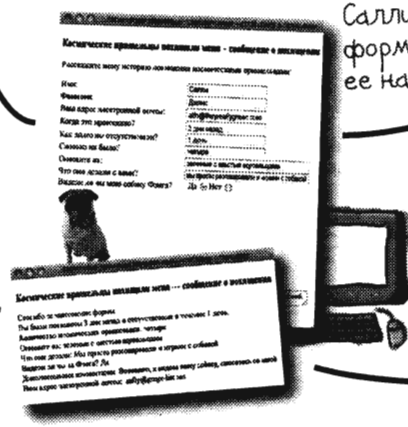
Вам, возможно, понадобится настроить PHP на вашем веб-сервере, чтобы он знал, как отправлять электронную почту.

Если функция `mail()` *не работает, проблема может заключаться в том, что в вашем PHP поддержка электронной почты настроена неправильно. Обратитесь на* `www.php.net/mail` *для получения более подробной информации об этом.*

Оуэн начинает получать электронные письма

Оуэн волновался, сможет ли он гарантированно получать из Сети информацию о похищении космическими пришельцами напрямую на свой почтовый адрес. Теперь ему не нужно волноваться, узнает ли он о том, что кто-то видел его собаку, потому что у него будут электронные адреса всех, кто вступит с ним в контакт. Более того, он сможет просматривать их ответы в любое свободное время.

Салли была недавно похищена.



Салли заполнила форму и отправила ее на сервер.

```
<form action
= "report.php"
...
```

Атрибут action тега <form> запускает механизм обработки формы сценарием report.php.

PHP-сценарий динамически создал HTML-страницу подтверждения.

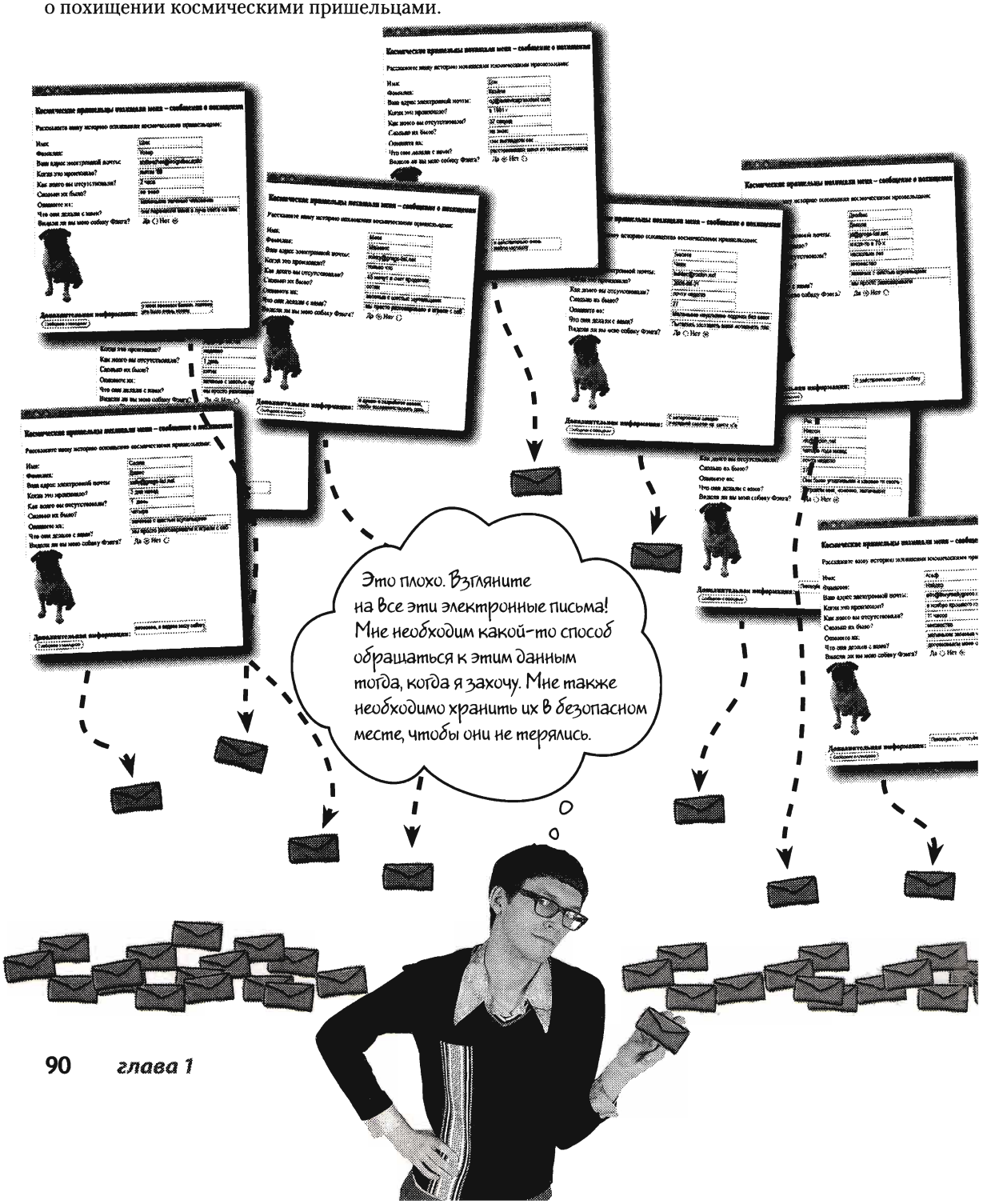
Это замечательно! С сообщениями о похищениях, присланными мне по электронной почте, я знаю, что найду Фэнга.

PHP-сценарий также создает электронное письмо и отправляет его Оуэну.

Теперь Оуэн счастлив, так как получает электронные письма с сообщениями о похищениях пришельцами посредством своей веб-формы.

Оуэн начинает терять электронные письма

То, что Оуэн получает электронные письма, хорошо. Но плохо то, что он получает их все больше и больше. Писем приходит настолько много, что ему становится трудно их отслеживать. Его почтовый ящик забит, и он уже случайно удалил несколько посланий. Оуэну необходим более удобный способ хранить данные о похищении космическими пришельцами.

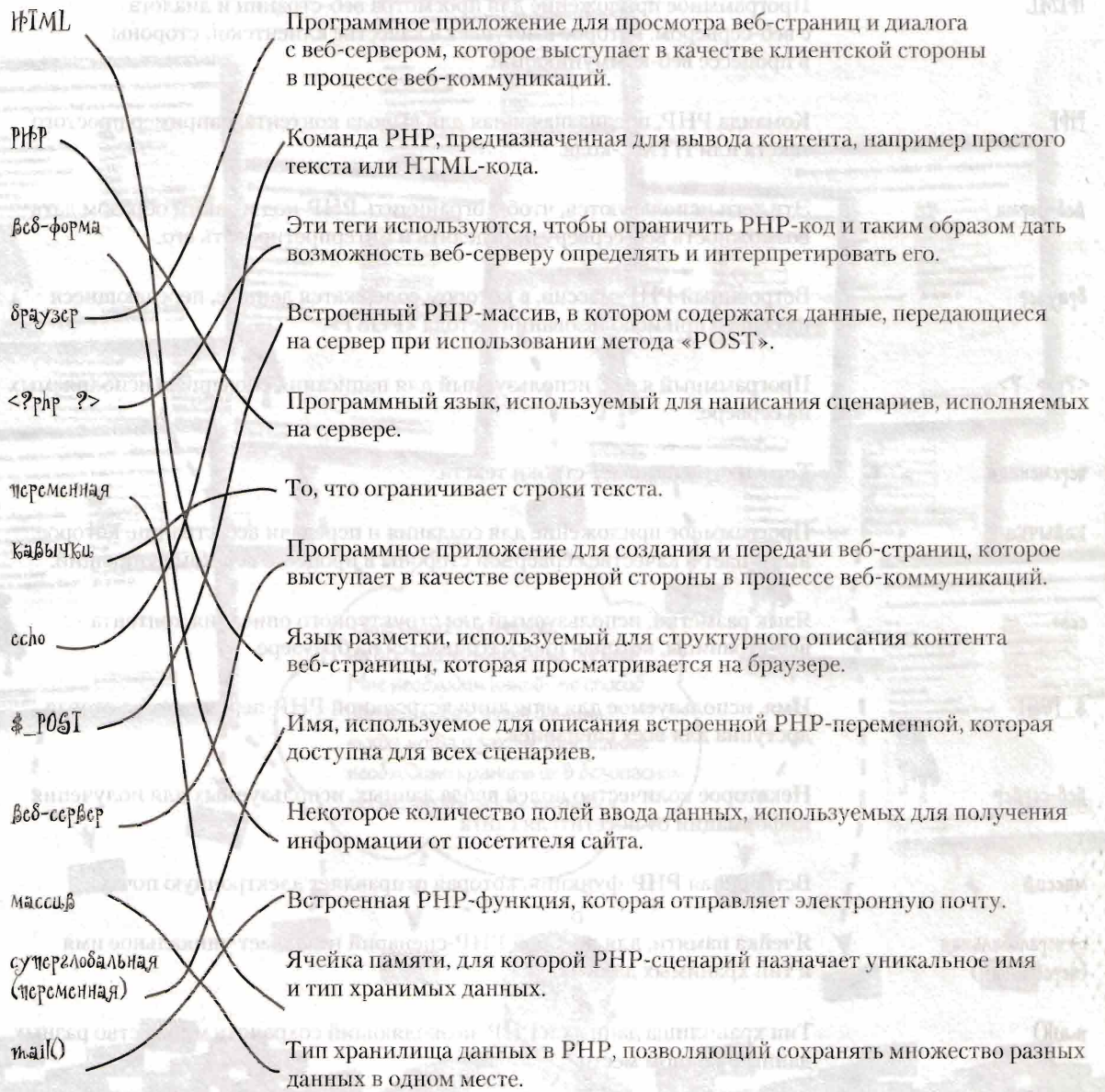


Космические пришельцы в голове?
Вытряхните их оттуда, соотнеся HTML- и PHP-компоненты с тем, что каждый из них делает.

HTML	Программное приложение для просмотра веб-страниц и диалога с веб-сервером, которое выступает в качестве клиентской стороны в процессе веб-коммуникаций.
PHP	Команда PHP, предназначенная для вывода контента, например простого текста или HTML-кода.
веб-форма	Эти теги используются, чтобы ограничить PHP-код и таким образом дать возможность веб-серверу определять и интерпретировать его.
браузер	Встроенный PHP-массив, в котором содержатся данные, передающиеся на сервер при использовании метода «POST».
<?php ?>	Программный язык, используемый для написания сценариев, исполняемых на сервере.
переменная	То, что ограничивает строки текста.
кавычки	Программное приложение для создания и передачи веб-страниц, которое выступает в качестве серверной стороны в процессе веб-коммуникаций.
echo	Язык разметки, используемый для структурного описания контента веб-страницы, которая просматривается на браузере.
$_POST	Имя, используемое для описания встроенной PHP-переменной, которая доступна для всех сценариев.
веб-сервер	Некоторое количество полей ввода данных, используемых для получения информации от посетителя сайта.
массив	Встроенная PHP-функция, которая отправляет электронную почту.
суперглобальная (переменная)	Ячейка памяти, для которой PHP-сценарий назначает уникальное имя и тип хранимых данных.
mail()	Тип хранилища данных в PHP, позволяющий сохранять множество разных данных в одном месте.

Что к чему
Решение

Космические пришельцы в голове? Вытряхните их оттуда, соотнеся каждый HTML- и PHP-компонент с тем, что каждый из них делает.



Ваш инструментарий PHP и MySQL

В главе 1 вы узнали, как использовать PHP, чтобы вдохнуть жизнь в веб-форму Оуэна. Взгляните на все, что вы уже изучили...

PHP

Серверный язык сценариев, который позволяет вам манипулировать контентом веб-страницы на сервере перед тем, как она будет отправлена на браузер.

PHP-сценарий

Текстовый файл, содержащий PHP-код для выполнения различных задач на сервере.

переменная

Контейнер для хранения данных. В PHP имя переменной всегда начинается со знака доллара, например: $variable_name.

$_POST

Специальная переменная, которая содержит данные формы.

<?php ?>

Между этими тегами должен размещаться PHP-код в вашем сценарии.

mail()

PHP-функция для отправления электронной почты. Она принимает в качестве параметров тему электронного письма, его содержание и адрес электронной почты назначения. Вы также можете указать (выборочно) адрес электронной почты отправителя.

echo

PHP-команда для вывода данных и отправки их браузеру. Ее синтаксис:

```
echo 'Hello World';
```

MySQL

Приложение, которое позволяет вам сохранять данные в таблицах базы данных, а также добавлять и извлекать эти данные с помощью языка запросов SQL.

SQL

Язык запросов для работы с базами данных, например MySQL.

клиентский код

Код, который интерпретируется на браузере.

Серверный код

Код, который интерпретируется на веб-сервере, а не на клиентском компьютере.

массив

Структура данных, содержащая несколько значений. Доступ к каждому значению осуществляется через его индекс.

Экранированная последовательность

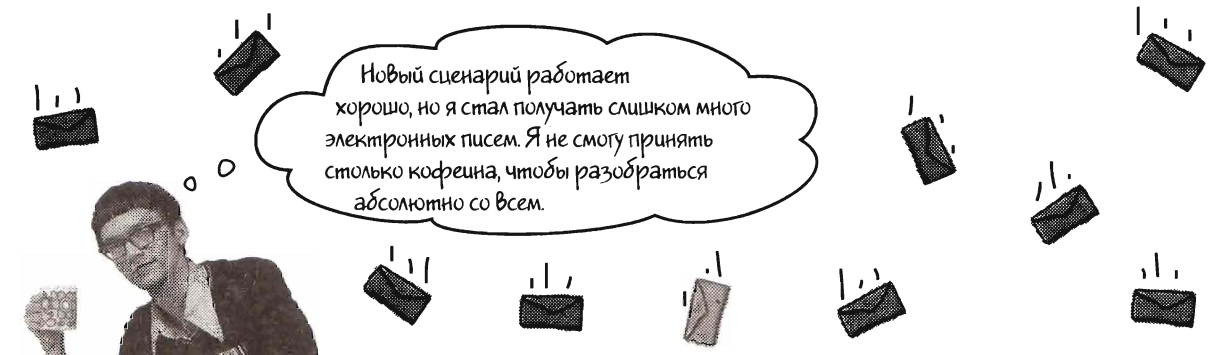
Предоставляет возможность добавить символ, который ввести трудно (клавиша с таким символом отсутствует на клавиатуре) или интерпретация которого может вызвать конфликт с другим кодом PHP. Например, \n — символ новой строки.

Как все это согласуется между собой

Перед тем как начинать создание системы, очень важно понимать, как отдельные ее элементы согласуются между собой. Вы создали свой первый PHP-сценарий, и он хорошо работает. Но получение результатов обработки вашей формы больше вас устраивать не может. Вам необходим способ, позволяющий сохранять эти результаты так долго, как это вам необходимо. С другой стороны, этот способ должен позволять вам обращаться к данным по мере необходимости. Хорошо известная система управления базами данных MySQL может надежно хранить ваши данные. Но вам необходимо как-то подключить ваш сценарий к этой системе, чтобы воспользоваться теми возможностями, которые она предоставляет.

это новая глава

PHP-форма Оуэна работает хорошо. Слишком хорошо...



Сценарий Оуэна с поддержкой электронной почты работал успешно, пока приходило немного сообщений, но сейчас он стал получать электронные письма в большом количестве — значительно больше того, которое он в состоянии обработать.

Он случайно удалил несколько электронных писем не читая. Другая их часть была отправлена в каталог, предназначенный для сбора спама, который он никогда не просматривает. Фактически часть электронных писем была скрыта от него в тот самый момент, когда ему крайне необходимо было их прочитать... Оуэну необходим способ сохранять все сообщения так, чтобы он мог просматривать их, когда у него возникает возможность, и легко находить те, которые имеют отношение к Фэнгу.

Поддержание в порядке всех сообщений о похищениях, прибывающих на почтовый ящик Оуэна, в конце концов приведет его в состояние крайнего перевозбуждения от выпитого кофе.

Космические пришельцы похищали меня — сообщение о похищении — Исходящие

От: sally@gregs-list.net
Тема: **Космические пришельцы похищали меня — сообщение о похищении**
Дата: 1 октября 2008 г. 12:11:29 PM CDT
Кому: owen@aliensabductedme.com

Салли Джонс была похищена 3 дня назад и отсутствовала в течение 1 дня.
Количество космических пришельцев: четыре
Описание космических пришельцев: зеленые с шестью щупальцами
Что они делали? Мы просто разговаривали и играли с собакой
Фэнг замечен? Да
Дополнительная информация: Возможно, я видела вашу собаку, свяжитесь со мной.

В этом потерянном сообщении о похищении упоминалось о том, что собаку видели. Это как раз та информация, которая крайне необходима Оуэну.

Оуэну необходимо, чтобы сообщения, подобные этому, сохранялись в одном надежном месте, где бы он мог внимательно просматривать их, отбирая те, в которых упоминается, что Фэнг был замечен.

Это как раз тот случай, где база данных **MySQL** может помочь...

На тот случай, если вы этого не знаете: многие люди произносят слово MySQL как «май-эс-кью-эль».

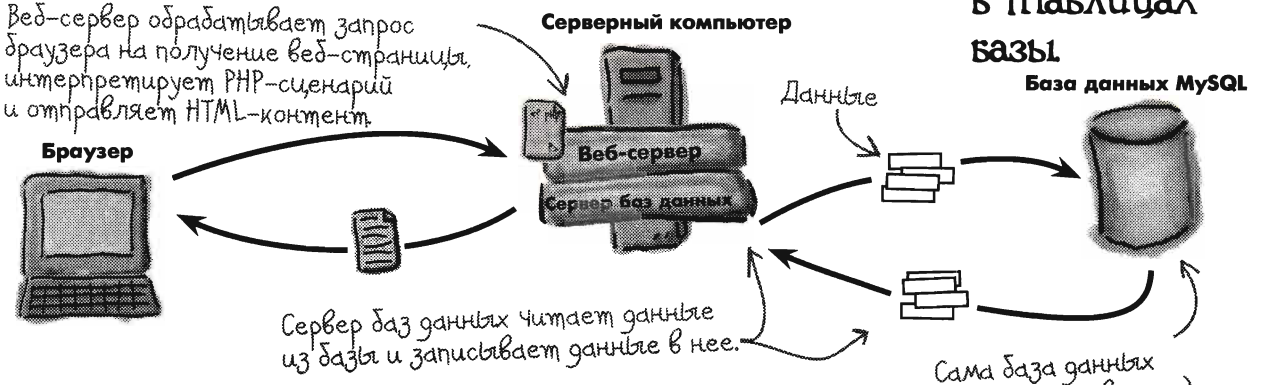
MySQL — замечательное средство для хранения данных

Оуэну действительно необходимо хранить сообщения о похищении космическими пришельцами в более удобном и безопасном месте, чем электронный почтовый ящик. Ему нужна база данных, которую можно рассматривать как что-то вроде виртуального шкафа, в котором данные находятся в высоко структурированном, систематизированном виде. При этом вы можете извлекать из нее именно те данные, которые вам необходимы, и в тот момент времени, когда вам это нужно.

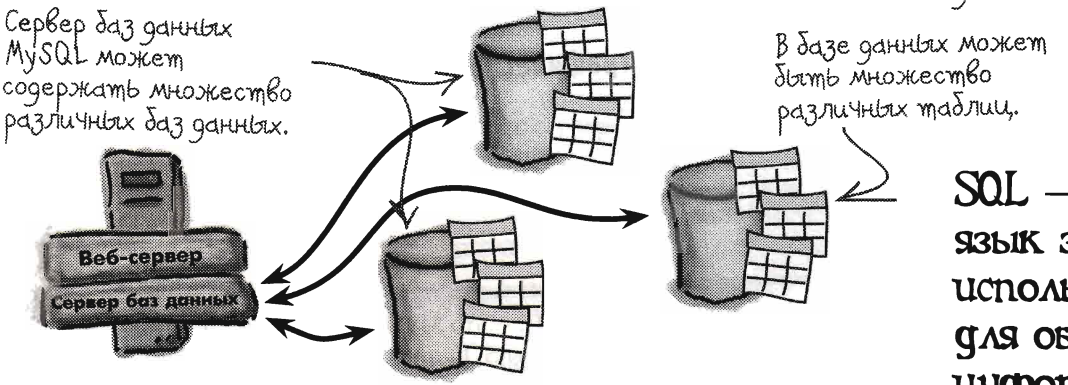
SQL-часть в слове MySQL означает структурированный язык запросов (Structured Query Language).

База данных управляется специальной программой, называемой **сервером баз данных**. В нашем случае это **MySQL**. Вы обмениваетесь информацией с сервером баз данных на языке, который он понимает; в нашем случае это структурированный язык запросов **SQL**. Обычно сервер баз данных работает на том же компьютере, что и веб-сервер, образуя вместе с ним ансамбль, читающий и записывающий информацию и предоставляющий ее посетителям сайта.

MySQL хранит данные в таблицах базы.



В структуре базы данных MySQL существуют таблицы, в колонках и рядах (строках) которых хранятся данные, которые могут быть связаны между собой по определенным критериям. Большинство веб-приложений используют одну или более таблиц внутри одной базы данных, примерно так же, как хранятся различные папки с файлами внутри одного шкафа.



SQL — это язык запросов, используемый для обмена информацией с базой данных MySQL.

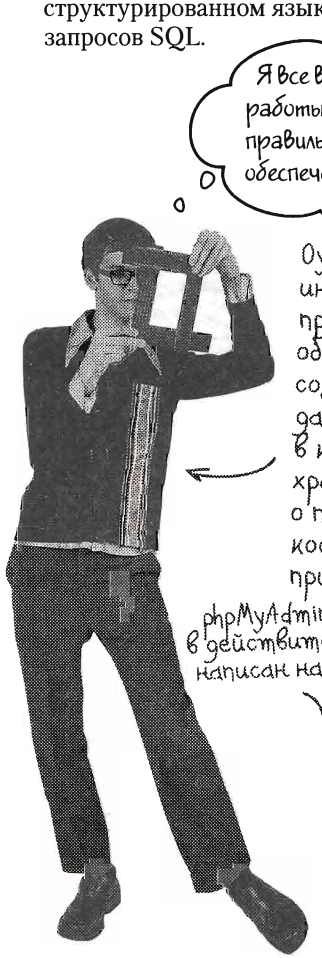
Используя данные о похищении космическими пришельцами, сохраненные в базе данных MySQL, Оуэн в любое удобное для него время может анализировать сообщения, полученные от тех посетителей его сайта, кто ответил «Да» на вопрос о Фэнге. Все, что ему необходимо для этого, — это написать небольшой SQL-код для обмена информацией с сервером баз данных.

Оуэну необходима база данных MySQL

Итак, решено: MySQL — отличная база данных, и Оуэну она необходима для хранения сообщений о похищениях космическими пришельцами. Тогда он сможет внести в сценарий `report.php` изменения, которые позволят сохранять данные в таблице базы данных вместо того, чтобы получать их по электронной почте. В таблице данные будут безопасно и надежно сохраняться по мере поступления их от жертв похищений, предоставляя Оуэну достаточно времени для отбора тех из них, в которых могут содержаться сведения о Фэнге. Но вначале о главном — базе данных!

Создание базы данных MySQL требует обмена информацией с MySQL-сервером баз данных.

Создание базы данных MySQL требует наличия сервера баз данных MySQL и специального инструментального программного обеспечения. Причина этого в том, что в отличие от веб-сервера обмен информацией с сервером баз данных должен осуществляться с использованием запросов, составленных на специальном структурированном языке запросов SQL.

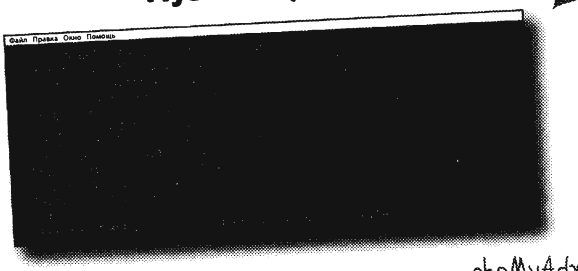


Я все время слышу, что успех выполнения работы зависит от выбранного инструмента. Как мне правильно выбрать инструментальное программное обеспечение для создания базы данных и таблицы?

Оуэну необходимо инструментальное программное обеспечение, чтобы создать базу данных/таблицу, в которой он будет хранить сообщения о похищениях космическими пришельцами.

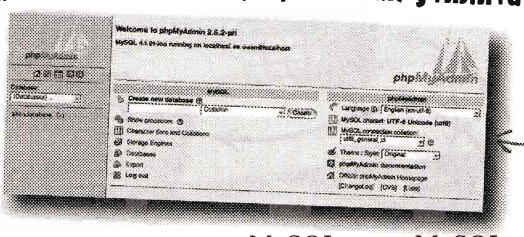
MySQL-терминал — это утилита, реализующая интерфейс командной строки, то есть предоставляющая пользователю возможность вводить запросы SQL в командной строке в текстовом режиме.

MySQL-терминал



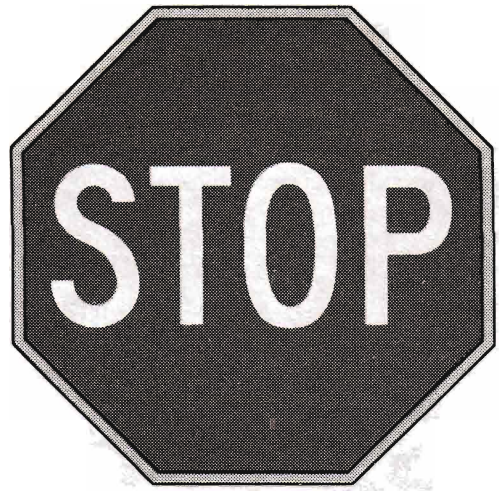
phpMyAdmin в действительности написан на PHP.

phpMyAdmin — графическая утилита



phpMyAdmin — это графическая утилита, позволяющая пользователю создавать базу данных и таблицы через веб-интерфейс.

Две популярные утилиты MySQL — это MySQL-терминал и phpMyAdmin. Обе позволяют выполнять запросы SQL для создания баз данных и таблиц, добавлять данные, просматривать их и т. д., но phpMyAdmin продвинулся на шаг вперед, реализуя веб-интерфейс с поддержкой принципа «указал и щелкнул». Некоторые хостинговые компании включают phpMyAdmin в стандартный MySQL-сервис, в то время как MySQL-терминал может быть использован для получения доступа к большинству MySQL-инсталляций.



У вас должен быть установлен сервер баз данных MySQL, прежде чем вы приступите к чтению этой страницы.

Без такого сервера невозможно помочь Оуэну! Если сервер баз данных MySQL у вас уже установлен и находится в рабочем состоянии, продолжайте чтение дальше. Если нет — обратитесь к приложению II и установите сервер согласно инструкциям, изложенным в нем. Если вы пользуетесь услугами хостинговой компании, предлагающей MySQL, попросите их предоставить вам доступ к серверу баз данных MySQL. Для получения такого доступа вам понадобятся некоторые параметры, и сейчас самое время в них разобраться. Они будут нужны вам позднее, поэтому тщательно проверьте их, прежде чем записать.

- ☐ **Размещение моего MySQL-сервера (IP-адрес или доменное имя):** .....
- ☐ **Мое имя пользователя базы данных:** .....
- ☐ **Мой пароль:** .....

Вам необходимо все это проверить.

Если вы боитесь, что данная книга попадет в плохие руки, не заполняйте эту строку.

После получения этой информации о сервере баз данных MySQL все, что вам остается, — это удостовериться, что сервер находится в рабочем состоянии. Отметьте одну из позиций внизу, для того чтобы убедиться, что вы можете успешно получить доступ к MySQL-серверу.

- ☐ **Я могу успешно получить доступ к MySQL-серверу, используя MySQL-терминал.**
- ☐ **Я могу успешно получить доступ к MySQL-серверу, используя phpMyAdmin.**
- ☐ **Я могу успешно получить доступ к MySQL-серверу, используя** .....

Вам достаточно отметить только одну позицию.

Если вы нашли какую-нибудь другую утилиту MySQL, выполняющую эти задачи, запишите ее сюда.

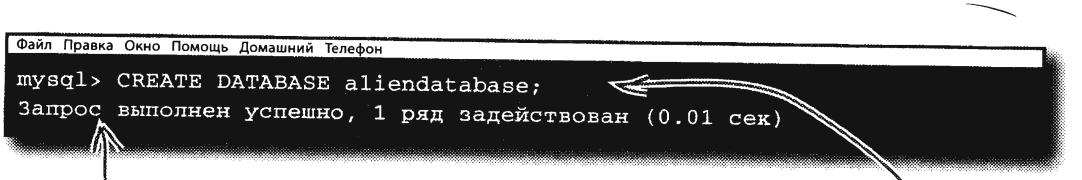
Создание базы данных и таблицы в MySQL

В некоторых инсталляциях MySQL база данных уже создана. Если в вашей инсталляции нет базы, необходимо ее создать, используя SQL-запрос `CREATE DATABASE` в MySQL-терминале. Но сначала вам необходимо открыть MySQL-терминал. Для этого введите команду `mysql` в окне операционной системы (в Windows это «Командная строка»). В случае успешного выполнения этой команды вы увидите приглашение терминала MySQL: **mysql>**.

Для создания новой базы данных о похищениях космическими пришельцами введите:

```
CREATE DATABASE aliendatabase;
```

как показано на рисунке:



Сервер MySQL обычно сообщает о результате выполнения запроса.

Когда вы пользуетесь терминалом, всегда ставьте точку с запятой в конце каждого запроса.

Перед тем как создавать таблицу в базе данных, вы должны выбрать ее, то есть сделать ее активной. Введите запрос:

```
USE aliendatabase;
```

SQL-запрос на создание таблицы выглядит посложнее, так как вы должны точно указать, данные каких типов будут сохраняться в этой таблице. Давайте взглянем подробнее на этот SQL-запрос, прежде чем вводить его в терминале:

Это SQL-запрос на создание новой таблицы.

```
CREATE TABLE `aliens_abduction` (
  `first_name` VARCHAR(30),
  `last_name` VARCHAR(30),
  `when_it_happened` varchar(30),
  `how_long` VARCHAR(30),
  `how_many` VARCHAR(30),
  `alien_description` VARCHAR(100),
  `what_they_did` VARCHAR(100),
  `fang_spotted` VARCHAR(10),
  `other` VARCHAR(100),
  `email` VARCHAR(50)
);
```

Все остальное — это информация о том, данные каких типов будут сохраняться в этой таблице.

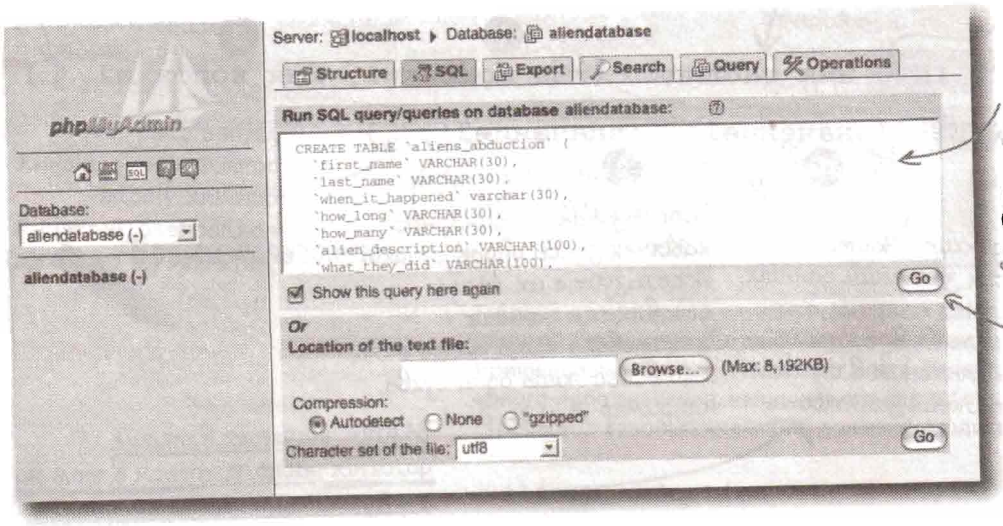
Все стандартные запросы, вводимые в MySQL-терминале, должны заканчиваться точкой с запятой.

Для того чтобы создать новую таблицу, введите большой запрос `CREATE TABLE` в MySQL-терминале (вы можете найти код этого запроса на веб-странице по адресу `www.headfirstlab.com/books/hfphp`). После успешного выполнения этого запроса у вас будет сияющая новизной таблица `aliens_abduction`.

Ответ сервера MySQL «Query OK» (Запрос выполнен успешно) говорит вам о том, что таблица была создана без проблем.

```
Файл Правка Окно Помощь Домашний Телефон
mysql> CREATE TABLE `aliens_abduction` (
    `first_name` VARCHAR(30),
    `last_name` VARCHAR(30),
    `when_it_happened` varchar(30),
    `how_long` VARCHAR(30),
    `how_many` VARCHAR(30),
    `alien_description` VARCHAR(100),
    `what_they_did` VARCHAR(100),
    `fang_spotted` VARCHAR(10),
    `other` VARCHAR(100),
    `email` VARCHAR(50)
);
Запрос выполнен успешно, 0 рядов задействовано (0.14 сек)
```

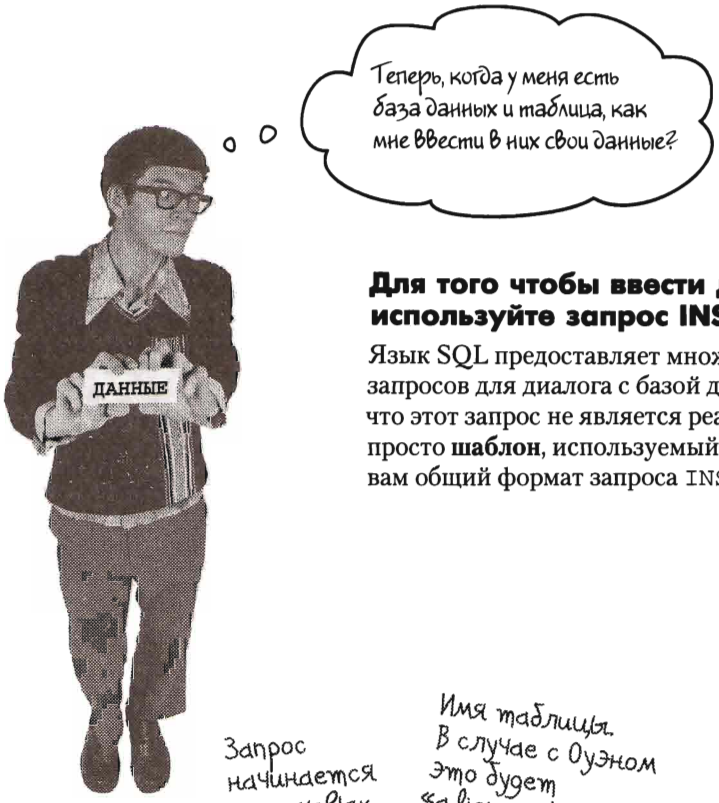
В вашей инсталляции MySQL вам может быть предложена основанная на веб-технологии утилита phpMyAdmin, которая позволяет получить доступ к вашей базе данных и таблицам через графический интерфейс. Пользовательский интерфейс phpMyAdmin-утилиты позволит вам пройти весь путь создания базы данных и таблиц с помощью необходимых кнопок и других органов управления или ввести SQL-запросы непосредственно, как будто вы находитесь в MySQL-терминале. Щелкните кнопкой мыши на ярлыке, чтобы перейти на вкладку окна phpMyAdmin, которая работает так же, как MySQL-терминал.



Вы можете ввести такой же запрос, как вы вводили в MySQL-терминале. Просто нажмите кнопку «Go» (Начать), чтобы запрос был выполнен.

После ввода SQL-запроса нажмите эту кнопку, чтобы выполнить его.

Таким образом, вкладка с ярлыком SQL-утилиты phpMyAdmin предоставляет возможность вводить SQL-запросы непосредственно, как будто вы используете MySQL-терминал.



Для того чтобы ввести данные в таблицу, используйте запрос INSERT.

Язык SQL предоставляет множество продвинутых запросов для диалога с базой данных. Имейте в виду, что этот запрос не является реальным запросом, это просто **шаблон**, используемый для того, чтобы показать вам общий формат запроса `INSERT`.

Запрос начинается с ключевых слов INSERT INTO.

Имя таблицы. В случае с Оуэном это будет «aliens_abduction».

Следующая часть — это имена колонок таблицы, разделенные запятыми.

Далее следуют имена остальных колонок. После имени последней колонки запятая не ставится.

```
INSERT INTO имя_таблицы (имя_колонки1, имя_колонки2, ...)
VALUES ('значение1', 'значение2', ...)
```

Это ключевое слово сообщает, что далее последует список значений для колонок.

Следующая часть — список значений данных, которые должны быть добавлены в таблицу. Эти значения в списке разделены запятыми.

Использование одинарных кавычек здесь оправданно. Используйте их, когда добавляете данные текстового типа, даже если это всего лишь одна буква, например 'A' или 'Я'.

Далее следуют остальные значения данных. После последнего значения запятая не ставится.

ВАЖНО: значения в списке должны располагаться в том же порядке, что и имена колонок.

Очень важно отметить то, что значения данных в скобках во втором списке должны располагаться **в том же порядке, что и имена колонок таблицы**. Это позволяет серверу согласовать значения данных с именами колонок, к которым они относятся.

Запрос INSERT в действии

Далее опишем, как запрос INSERT может быть использован для того, чтобы добавить данные о похищениях космическими пришельцами в новую таблицу Оуэна aliens_abduction.

Запомни!

Порядок имеет значение!

*Значения данных, подлежащие добавлению, должны быть перечислены точно в **таком же порядке, что и имена колонок таблицы.***

Это имя таблицы, в которую будут добавлены данные. Это НЕ имя базы данных.

Имена ваших колонок в списке, заключенном в скобки. Они отделены друг от друга запятыми.

```
INSERT INTO aliens_abduction (first_name, last_name,
when_it_happened, how_long, how_many, alien_description,
what_the_did, fang_spotted, other, email)
VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре',
'зеленые с шестью щупальцами', 'мы просто разговаривали и играли с собакой',
'да', 'возможно, я видела вашу собаку, свяжитесь со мной',
'sally@gregs-list.net')
```

Второй список значений данных для всех колонок заключен в скобки. Они также отделены друг от друга запятыми.

В конце SQL-запросов, встроенных в PHP-команду, точка с запятой не ставится. Она всегда ставится только в конце самой PHP-команды.

Все эти значения относятся к текстовому типу, поэтому они заключены в одинарные кавычки.

Время поработать

Таблица aliens_abduction, показанная ниже, пока не содержит данных. Запишите в нее данные о похищении Салли космическими пришельцами. Если вам не хватит места в ячейке для добавления какого-либо значения, напишите его над таблицей и укажите стрелкой, в какой из ячеек оно должно находиться.

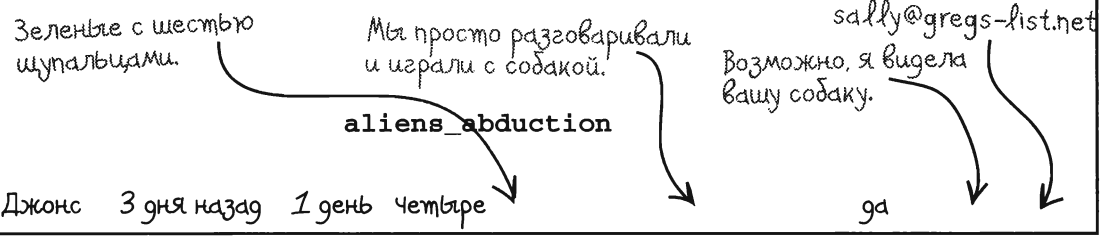
Это имена колонок.

aliens_abduction

first_name	last_name	when_it_happened	how_long	how_many	alien_description	what_they_did	fang_spotted	other	email

Решение задачи

Таблица `aliens_abduction`, показанная ниже, пока не содержит данных. Запишите в нее данные о похищении Салли космическими пришельцами. Если вам не хватит места в ячейке для добавления какого-либо значения, напишите его над таблицей и укажите стрелкой, в какой из ячеек оно должно находиться.



не бывает глупых вопросов

В: Я не уверен, что до конца понял разницу между базой данных и таблицей. Разве данные хранятся не в них обеих?

О: Это так, но необходимо некоторое уточнение. Таблицы служат для разделения данных на связанные между собой группы, в результате чего вы имеете дело не просто с одним огромным массивом данных. Это примерно как разница между тем, чтобы свалить множество пар обуви в одну большую коробку, и тем, чтобы вначале разложить каждую пару в отдельную, небольшую. Большая коробка — это база данных, маленькие коробки — таблицы. Таким образом, данные хранятся в таблицах, а таблицы — в базе данных.

В: Что в действительности является MySQL-терминалом? Как мне найти его на моем компьютере?

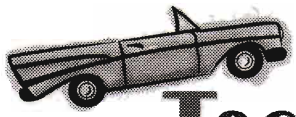
О: MySQL-терминал — это техническое средство, позволяющее получить доступ к базе данных MySQL через интерфейс командной строки. MySQL-терминал не является единственной программой, обеспечивающей этот доступ. Это, скорее, соединение, которое вы устанавливаете из «стандартного» терминала, используя интерфейс командной строки. В роли такого «стандартного» терминала может выступать, например, терминальное приложение в Mac OS X. Способ, с помощью которого вы достигаете соединения с MySQL-терминалом, в значительной степени зависит от используемой вами операционной системы, а также от того, является ли ваш MySQL-сервер локальным или удаленным (то есть размещенным на вашем компьютере или где-то в другом месте). В приложении II изложено более подробно, какие шаги необходимо предпринять, чтобы получить доступ к MySQL-терминалу.

В: А что по поводу phpMyAdmin? Где я могу найти это приложение?

О: В отличие от MySQL-терминала phpMyAdmin — это веб-приложение, предоставляющее доступ к базе данных MySQL. По сути, оно является PHP-приложением, и это объясняет тот факт, что вы всегда получаете к нему доступ из веб-сервера, вместо того чтобы устанавливать как локальную клиентскую программу. Множество веб-хостинговых компаний предоставляют phpMyAdmin как составную часть своих стандартных хостинговых планов, поэтому вполне возможно, что оно для вас уже установлено. Если же нет, вы можете загрузить phpMyAdmin и установить его самостоятельно. Оно доступно для загрузки с сайта `www.phpmyadmin.net`. Но не забывайте, что это приложение должно быть установлено на веб-сервер и настроено так, чтобы обеспечить доступ к вашей базе данных MySQL, так же как любое другое PHP- и MySQL-веб-приложение.

В: Мне доступен и MySQL-терминал, и phpMyAdmin. Что из них я должен использовать для получения доступа к моей базе данных?

О: Это вопрос исключительно личных предпочтений. Преимущество phpMyAdmin заключается в том, что вы можете работать с вашей базой данных и таблицами визуально, без необходимости вводить SQL-запросы. Это может оказаться очень удобным, если вы чувствуете себя достаточно уверенно в SQL и не хотите вручную вводить запросы для выполнения каждой мелкой операции. Тем не менее на этом этапе, чтобы более полно понять, как происходит ваше взаимодействие с данными в базе, лучше использовать SQL-запросы. А в этом смысле оба приложения работают достаточно хорошо.



Тест-драйв

Сохраните наблюдения о похищениях космическими пришельцами в вашей базе данных с помощью SQL-запроса INSERT.

Используя инструментальное MySQL-приложение, такое как MySQL-терминал, или вкладку с ярлыком SQL в phpMyAdmin, введите запрос `INSERT` для сообщений о похищениях космическими пришельцами. В качестве примера ниже приведен запрос `INSERT` о похищении Салли Джонс.

```
INSERT INTO aliens_abduction (first_name, last_name,
  when_it_happened, how_long, how_many, alien_description,
  what_the_did, fang_spotted, other, email)
  VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре',
  'зеленые с шестью щупальцами', 'мы просто разговаривали и играли с собакой',
  'да', 'возможно, я видела вашу собаку, свяжитесь со мной',
  'sally@gregs-list.net')
```

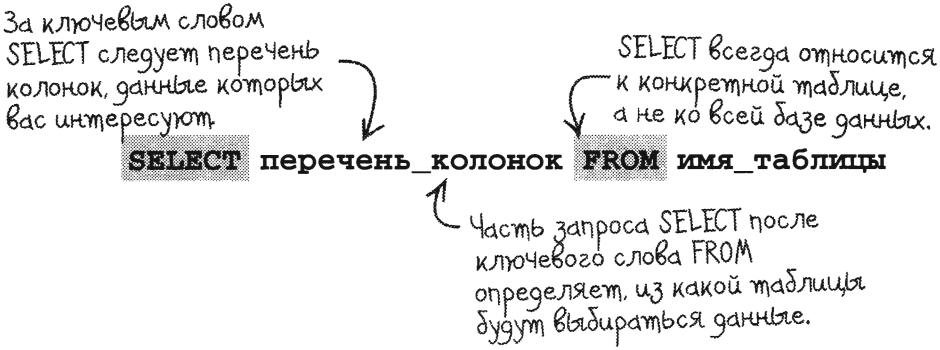
```
Файл Правка Окно Помощь
mysql> INSERT INTO aliens_abduction (first_name, last_name,
    when_it_happend, how_long, how_many, alien_description,
    what_the_did, fang_spotted, other, email)
    VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре',
    'зеленые, с шестью щупальцами', 'мы просто разговаривали
     и играли с собакой',
    'да', 'возможно я видела вашу собаку, Свяжитесь со мной',
    'sally@gregs-list.net')
Запрос выполнен успешно. 1 строка задействована (0.0005 сек)
```

В результате исполнения запроса INSERT в MySQL-терминале к таблице aliens_abduction была добавлена одна строка.

Запрос INSERT был выполнен успешно. Запишите внизу, как мы можем убедиться в том, что данные были добавлены.

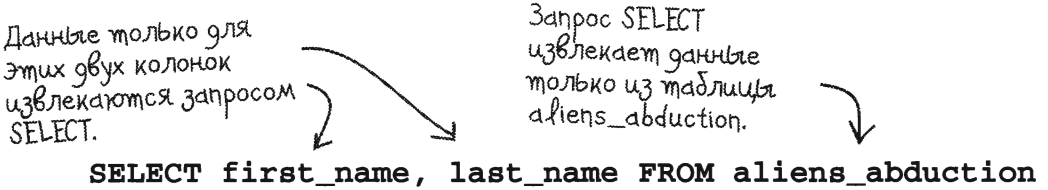
Используйте SELECT, чтобы извлечь данные из таблицы

Добавление данных в таблицу — это очень удобная вещь, но трудно избавиться от чувства беспокойства по поводу того, что вы не получили подтверждения, что данные действительно прошли весь путь до таблицы. Это примерно как если бы вы вносили деньги на свой счет в банке, но никогда не имели возможности проверить баланс. Запрос `SELECT` — это именно то, что позволит вам контролировать «баланс» своей таблицы в базе данных. Вернее, `SELECT` позволяет вам запросить отдельные колонки данных вашей таблицы.

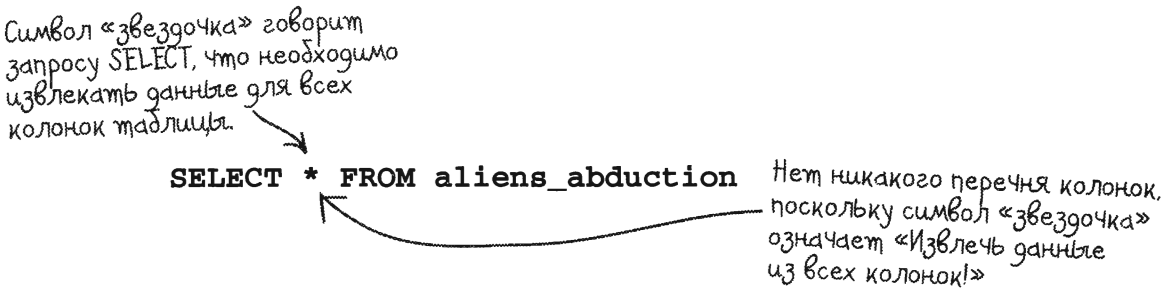


Запрос SQL SELECT извлекает данные колонок из таблицы

Имена колонок, перечисляемых в запросе, должны отделяться друг от друга запятыми. Независимо от количества колонок в таблице извлекаются данные только для колонок, указанных в этом перечне. Этот запрос `SELECT` извлекает только имя и фамилию похищенных космическими пришельцами из таблицы `aliens_abduction`:



Для проверки выполнения запроса `INSERT` вам необходим быстрый способ, позволяющий **просмотреть все данные**, а не только данные для отдельных колонок, имеющиеся в таблице. В запросе `SELECT` для подобных случаев предусмотрено сокращенное указание перечня всех колонок:

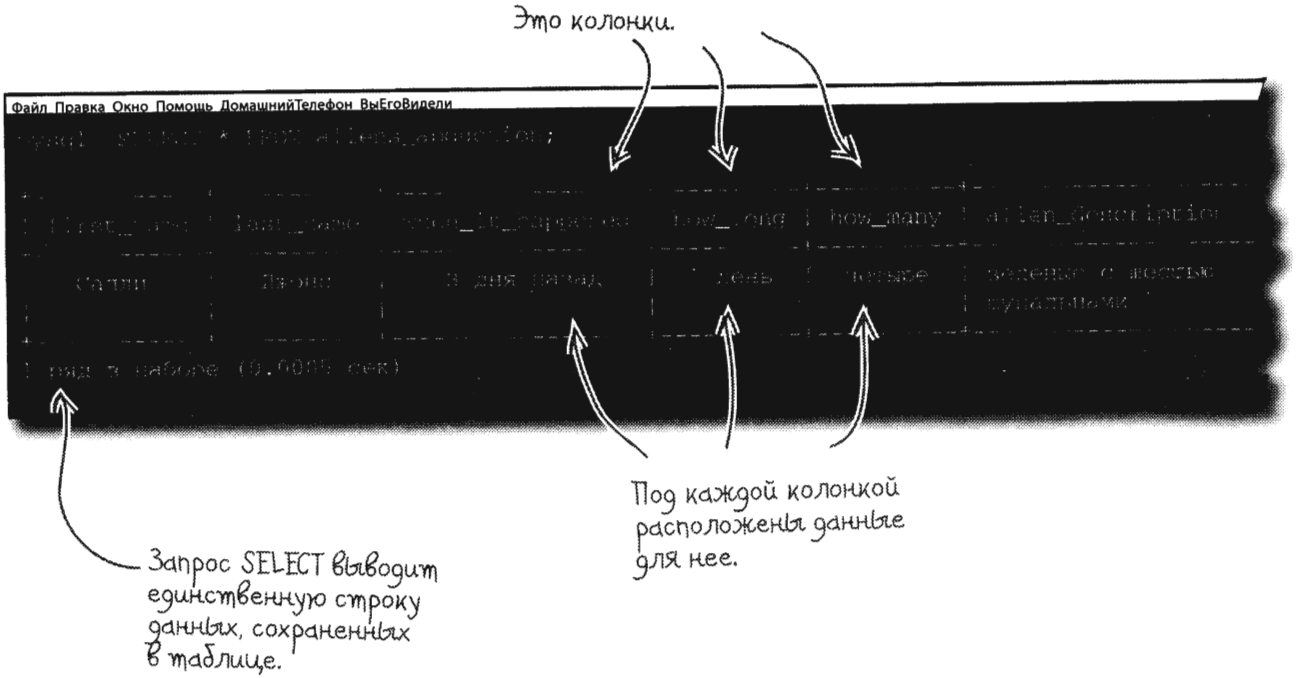


Тест-драйв

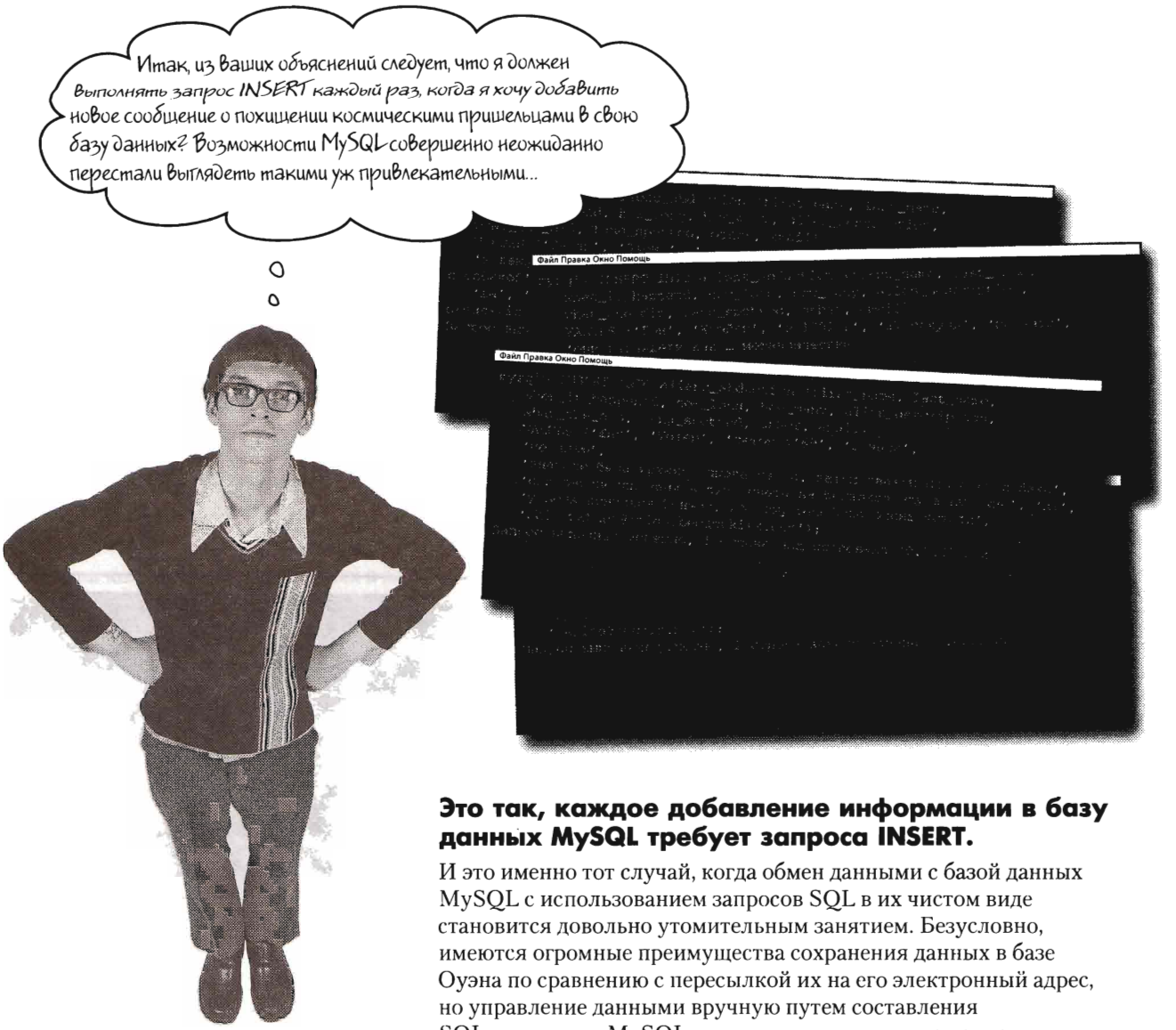
Убедитесь в том, что запрос INSERT на добавление данных по похищениям космическими пришельцами выполнился, путем извлечения данных с помощью запроса SELECT.

Выполните запрос `INSERT`, используя MySQL-терминал, чтобы просмотреть все содержимое таблицы `aliens_abduction`. Убедитесь, что в результате выполнения запроса появилась новая строка с данными, которые вы только что добавили.

```
SELECT * FROM aliens_abduction
```



Сколько строк данных в вашей таблице?



Это так, каждое добавление информации в базу данных MySQL требует запроса INSERT.

И это именно тот случай, когда обмен данными с базой данных MySQL с использованием запросов SQL в их чистом виде становится довольно утомительным занятием. Безусловно, имеются огромные преимущества сохранения данных в базе Оуэна по сравнению с пересылкой их на его электронный адрес, но управление данными вручную путем составления SQL-запросов в MySQL не выглядит удачным решением.

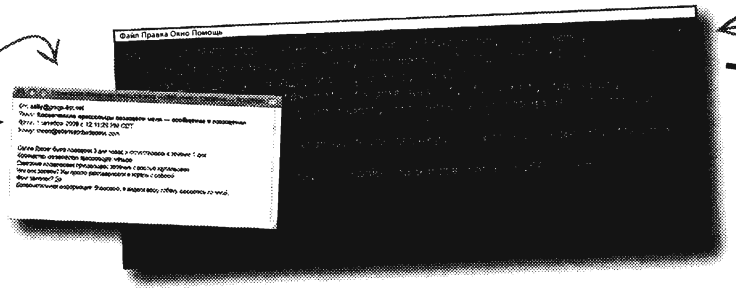
Как вы думаете, проблема добавления информации в базу данных MySQL Оуэна может быть решена?

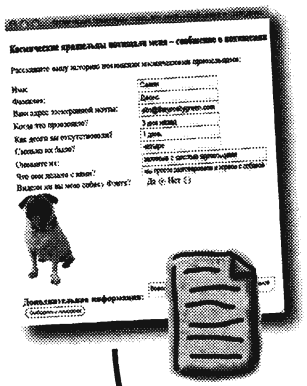
Пусть PHP обрабатывает эти утомительные запросы SQL

Решение проблемы Оуэна заключается не в том, чтобы избежать SQL, а в том, чтобы автоматизировать его с помощью PHP. В PHP можно задать такой код, в результате интерпретации которого на сервере будет составлен и передан на исполнение SQL-запрос и у вас не возникнет необходимости в использовании какого-либо инструментального программного обеспечения MySQL. Это означает, что HTML-форма Оуэна может вызвать PHP-сценарий для добавления информации в базу данных после того, как эта форма передана на сервер для обработки. Никаких электронных писем, никаких SQL-утилит, никаких проблем!

Оуэн создает SQL-запрос INSERT, который добавляет данные из электронного письма в базу данных.

HTML-форма генерирует электронное письмо, которое поступает к Оуэну, после чего он должен вручную добавить полученные данные в базу.

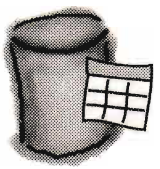




report.html

Без помощи PHP: требуется вручную добавлять в базу данных каждое сообщение о похищении космическими пришельцами с помощью SQL-запроса INSERT.

С помощью PHP: PHP-сценарий, используя SQL-запрос INSERT, автоматически дополнит базу данных, после того как форма будет передана на сервер для обработки.



```
<?php
   $dbc = mysqli_connect('data.aliensubductedme.com', 'owen', 'aliensrool',
'aliensdatabase')
      or die ('Ошибка соединения с MySQL-сервером');
   $query = "INSERT INTO aliens_abduction (first_name, last_name, " .
      "when_it_happend, how_long, how_many, alien_description, " .
      "what_the_did, fang_spotted, other, email) " .
      "VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре', " .
      "'зеленые с шестью щупальцами', 'мы просто разговаривали и играли с собакой', " .
      "'да', 'возможно, я видела вашу собаку, свяжитесь со мной', 'sally@gregs-list.net')";
   $result = mysqli_query($dbc, $query)
      or die ('Ошибка при выполнении запроса к базе данных.');
   mysqli_close($dbc)
?>
```

report.php

HTML-форма вызывает PHP-сценарий, который добавляет данные формы в базу.

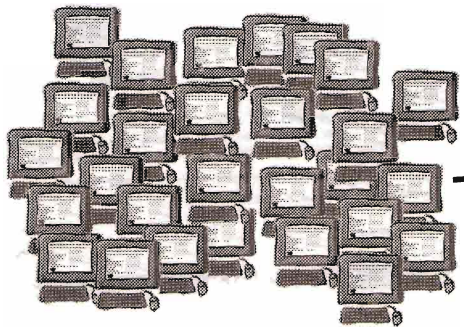
PHP-сценарий создает запрос INSERT, который добавляет данные формы в базу... Оуэн не принимает в этом непосредственного участия!

PHP управляет движением данных формы

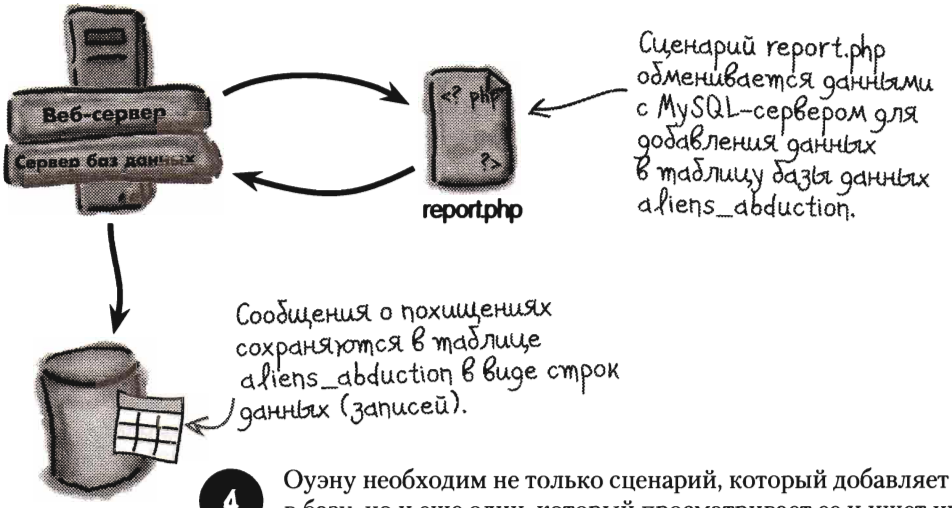
PHP расширяет возможности веб-формы Оуэна о похищениях космическими пришельцами, позволяя сценарию **отправлять данные формы *непосредственно* в базу данных** вместо электронного почтового ящика. Давайте рассмотрим рисунок, показывающий более подробно то, как работает это приложение, когда на сцену выходит PHP.

1 Салли заполняет форму о похищении космическими пришельцами и нажимает кнопку «Сообщение о похищении». Информация передается сценарию `report.php` на сервере.

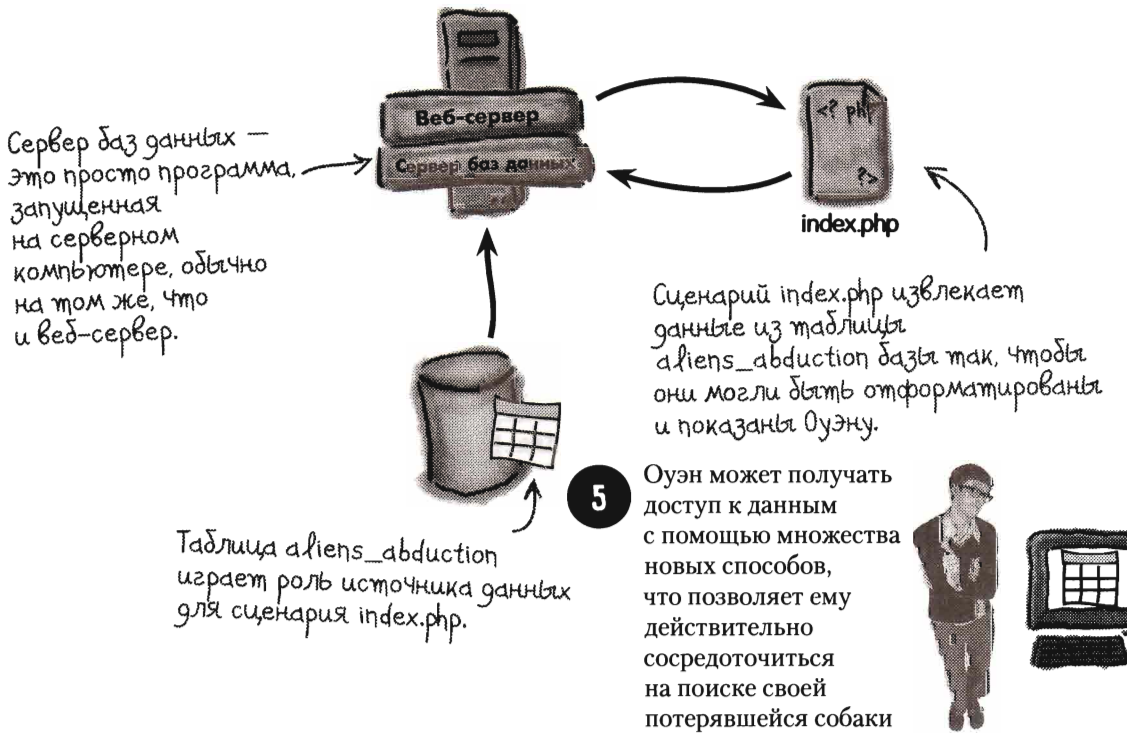
2 Все больше и больше людей продолжают заполнять и передавать формы на обработку.



3 Сценарий Оуэна `report.php` соединяется с базой данных MySQL и добавляет информацию из формы после каждой обработки, используя SQL-запрос `INSERT`.



4 Оуэну необходим не только сценарий, который добавляет данные в базу, но и еще один, который просматривает ее и ищет нужные данные. Фактически этот сценарий должен играть роль главной страницы его сайта. Сценарий `index.php` соединяется с базой данных, извлекает данные о похищении космическими пришельцами и показывает их Оуэну.



5 Оуэн может получать доступ к данным с помощью множества новых способов, что позволяет ему действительно сосредоточиться на поиске своей потерявшейся собаки Фэнга.

Соединение с вашей базой данных из PHP

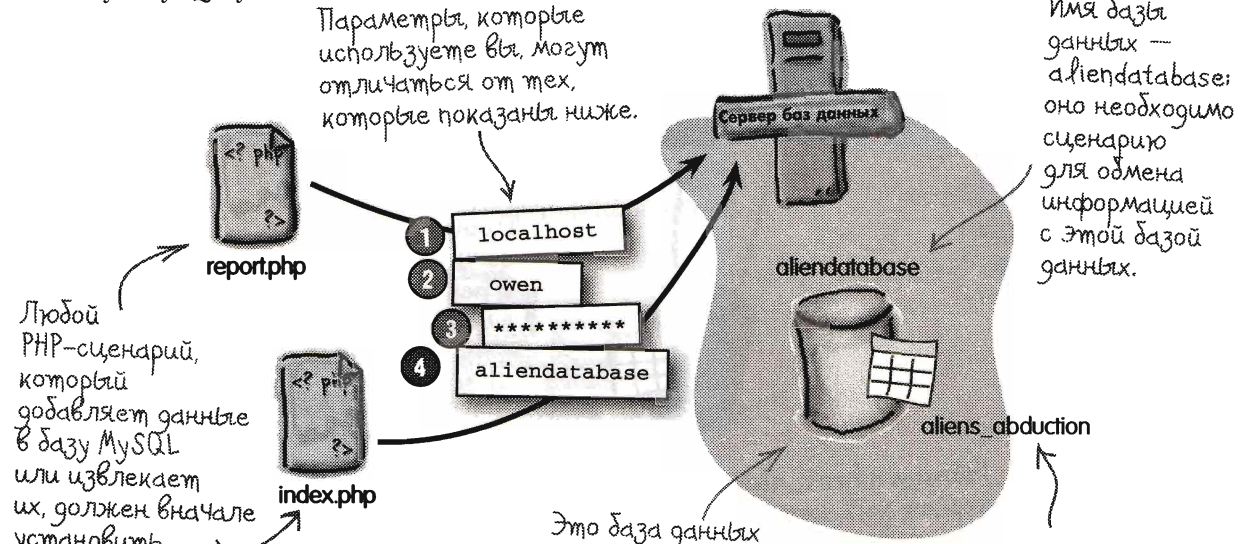
Прежде чем PHP-сценарий сможет добавлять или извлекать данные из базы MySQL, он должен подключиться к ней. Подключение к базе данных MySQL из PHP во многом подобно тому, как это происходит при доступе к базе данных из MySQL-утилит. Этот процесс требует ввода такой же информации. Помните три параметра, которые вы записали немного ранее при чтении этой главы? Вот они, вместе с еще одним, новым — именем базы данных. Давайте запишем их еще раз.

Отдел технического обслуживания вашей хостинговой компании или ваш веб-мастер могут дать вам имя этого хоста. Или, если ваш веб-сервер и MySQL-сервер установлены на одном и том же компьютере, вы можете использовать в качестве этого имени слово localhost.

1. **Размещение моего MySQL-сервера (IP-адрес или доменное имя)** .....
2. **Мое имя пользователя базы данных:** .....
3. **Мой пароль:** .....
4. **Имя моей базы данных:** .....

Имя базы данных, которую вы создали ранее, — aliendatabase. Если по каким-то соображениям вы дали своей базе данных другое имя или решили использовать существующую базу данных, применяйте это имя.

Размещение MySQL-сервера, имя пользователя базы данных, его пароль и имя самой базы данных — все это необходимо для установки соединения с базой данных MySQL из PHP-сценария. После того как соединение будет установлено, сценарий получит возможность делать SQL-запросы, точно так же как вы делали это вручную, используя MySQL-утилиты.



Добавление данных с помощью PHP-сценария

Осуществление MySQL-запроса с помощью PHP-сценария требует от вас вначале установки соединения с базой данных. Затем вы должны создать запрос в виде PHP-строки. Он не будет выполнен до тех пор, пока вы не отправите его на сервер баз данных. И, наконец, после того как вы завершите выполнение всех запросов к базе данных, необходимо закрыть соединение. Все эти задачи кодируются в PHP-сценарии. Вот пример, в котором показано, что происходит при добавлении новой строки данных о похищении космическими пришельцами:

Соединение с базой данных MySQL.

Здесь должны быть четыре параметра: ВАШИ, а не Оуэна.

```
<?php
    $dbc = mysqli_connect('data.aliensubductedme.com', 'owen', 'aliensrool', 'aliensdatabase')
        or die ('Ошибка соединения с MySQL-сервером');

$query = "INSERT INTO aliens_abduction (first_name, last_name, ".
        "when_it_happend, how_long, how_many, alien_description, " .
        "what_they_did, fang_spotted, other, email) " .
        "VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре', " .
        "''зеленые с шестью щупальцами', 'мы просто разговаривали и играли с собакой', " .
        "''да', 'возможно, я видела вашу собаку, свяжитесь со мной.', " . 'sally@gregs-list.net')";

$result = mysqli_query($dbc, $query)
        or die ('Ошибка при выполнении запроса к базе данных.');

  mysqli_close($dbc);
?>
```

Возможно, вы будете использовать localhost вместо доменного имени хоста в качестве места размещения вашей базы данных.

Создайте запрос INSERT в виде строки PHP-кода.

Здесь будьте особенно внимательны с использованием одинарных и двойных кавычек, так же как и с пробелами перед и после них!

Передайте серверу баз данных MySQL на выполнение запрос INSERT.

Эти функции требуют, чтобы ваш веб-сервер поддерживал PHP версии 4.1 или выше.

Как вы думаете, что делает каждая из этих PHP-функций в сценарии?

```
mysqli_connect()
mysqli_query()
mysqli_close()
```

Используйте PHP-функции для соединения с базой данных

Существуют три основные PHP-функции, используемые для обмена информацией с базой данных MySQL: `mysqli_connect()`, `mysqli_query()` и `mysqli_close()`. Если вы заметили повторяющийся префикс — это не совпадение. Все современные PHP-функции, которые осуществляют взаимодействие с MySQL, начинаются с `mysqli_`.

Более ранние наборы PHP-функций, которые осуществляли взаимодействие с MySQL, начинались с `mysql_` без буквы «i». «i» — это начальная буква слова improved (усовершенствованный), поэтому функции `mysqli_` в настоящее время считаются более предпочтительными.

`mysqli_connect()`

Открывает соединение с базой данных MySQL, используя информацию, содержащуюся в четырех параметрах, о которых вы уже знаете.

`mysqli_query()`

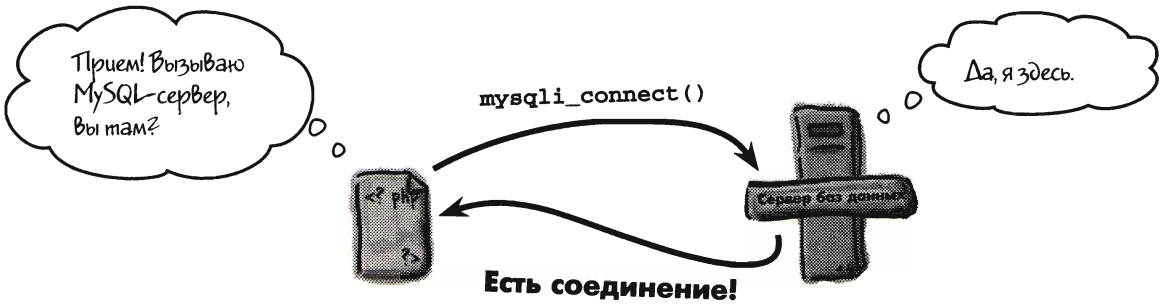
Осуществляет запрос к базе данных MySQL, который часто связан с добавлением или извлечением данных из нее.

`mysqli_close()`

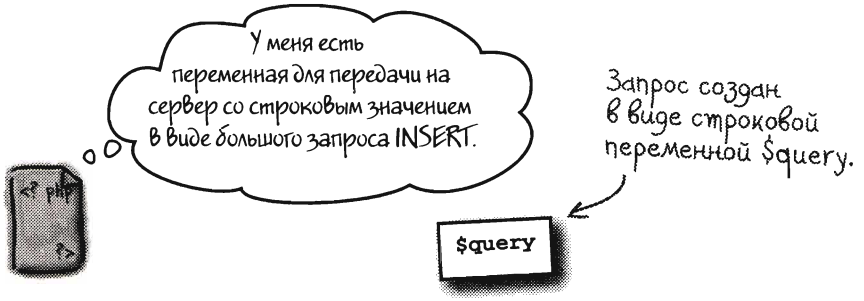
Закрывает соединение *с базой данных MySQL.*

Использование этих трех функций обычно происходит по вполне предсказуемым этапам.

1 Откройте соединение с базой данных, используя функцию mysqli_connect().
Передайте функции в качестве аргументов адрес размещения сервера, имя пользователя и его пароль, чтобы получить разрешение на обмен информацией с сервером баз данных MySQL. Укажите также имя базы данных, так как соединение осуществляется с конкретной базой.

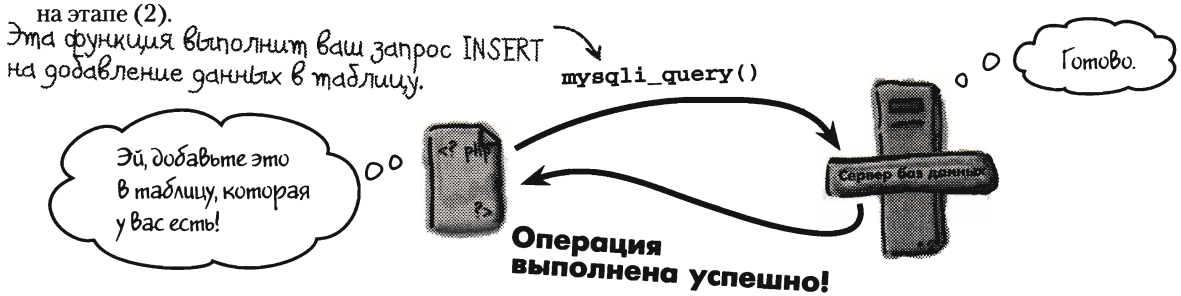


2 Создание переменной и присвоение ей значения строки SQL-запроса.
Для обмена информацией с сервером баз данных вы должны использовать SQL-запросы. Например, запрос `INSERT` необходим для того, чтобы добавить данные в таблицу `aliens_abduction`. Не существует специальных требований к выбору имени этой переменной (кроме общих требований к PHP-переменным), но непосредственное имя `$query` (переводится с английского как «запрос») можно считать удачным.



3 Передача запроса на сервер для выполнения с использованием функции mysqli_query().

Используйте переменную `$query` в качестве аргумента функции `mysqli_query()` для передачи на сервер баз данных MySQL информации, необходимой для добавления данных в таблицу `aliens_abduction`. С функцией `mysqli_query()` вы должны передать серверу ссылку на соединение, которое вы создали на этапе (1), и имя переменной, содержащей строку вашего запроса, созданную на этапе (2).



4 Закрытие соединения с базой данных с использованием функции mysqli_close().

И, наконец, функция `mysqli_close()` сообщает серверу баз данных MySQL о том, что вы прекращаете обмен информацией с ним.

Это имя ссылки на соединение с базой данных MySQL.

Если что-нибудь пойдет не так, дальнейший ход программы будет прекращен и вы получите это сообщение.

Это SQL-запрос INSERT, в результате выполнения которого данные добавляются в вашу базу.

```
<?php
  $dbc = mysqli_connect('data.aliensubductedme.com', 'owen', 'aliensrool', 'aliensdatabase')
    or die ('Ошибка соединения с MySQL-сервером');
  $query = "INSERT INTO aliens_abduction (first_name, last_name, " .
      "when_it_happened, how_long, how_many, alien_description, " .
      "what_the_did, fang_spotted, other, email) " .
      "VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре', " .
      "'зеленые с шестью щупальцами', 'мы просто разговаривали и играли с собакой', " .
      "'да', 'возможно, я видела вашу собаку, свяжитесь со мной', 'sally@gregs-list.net')";
  $result = mysqli_query($dbc, $query)
    or die ('Ошибка при выполнении запроса к базе данных.');
  mysqli_close($dbc);
?>
```

С помощью функции mysqli_query() PHP обменивается информацией с сервером баз данных MySQL. Код, содержащийся в переменной $query, — это код SQL, а не PHP.

Здесь мы закрываем соединение.

Давайте взглянем поближе на каждую из этих PHP-функций по работе с базами данных, начиная с `mysqli_connect()`...

Открытие соединения с помощью функции mysqli_connect()

Чтобы ваш PHP-сценарий смог создать соединение с базой данных с помощью функции `mysqli_connect()`, вам необходима некоторая информация, с которой вы начинаете знакомиться все ближе и ближе. Да, это именно та информация, которую вы использовали ранее, когда работали с MySQL-терминалом, плюс имя базы данных.

1. Соединение с помощью mysqli_connect().
2. Составление строки запроса.
3. Выполнение запроса с помощью mysqli_query().
4. Закрытие соединения с помощью mysqli_close().

Кто?

Ваше имя пользователя базы данных и пароль.
Вам необходимо ваше имя пользователя и пароль для сервера баз данных, на котором размещена ваша база. Эти данные вы либо устанавливаете сами, либо получаете от хостинговой компании, когда MySQL устанавливается впервые. Если вы устанавливаете свой собственный MySQL, следуйте инструкциям по выбору безопасных имен пользователей и особенно паролей.

Что?

Имя вашей базы данных.
В нашем примере мы назвали базу данных `aliendatabase`. Ваша база данных может иметь любое другое имя, которое вы решили ей дать, когда создавали ее. Если базу данных для вас создавала ваша хостинговая компания, используйте то имя, которое вам сообщат в ее отделе технической поддержки.

Где?

Расположение вашей базы данных (доменное имя, IP-адрес или `localhost`).
В нашем примере мы использовали несуществующее имя для места расположения базы данных Оуэна. Вы должны использовать реальное место расположения вашего MySQL-сервера. Часто это `localhost`, если сервер баз данных расположен на том же компьютере, что и веб-сервер. Ваша хостинговая компания сообщит вам эту информацию. Это может быть также IP-адрес или доменное имя, как у Оуэна; например, `вашсевер.вашпровайдер.com`.

Имя пользователя и его пароль для сервера баз данных, а также расположение MySQL-сервера и его имя в функции `mysqli_connect()` должны заключаться в кавычки.

```
$dbc = mysqli_connect('data.
aliensubductedme.com',
    'owen',
    'aliensrool',
    'aliendatabase');
```

Используйте эту переменную для проведения различных операций с базой данных.

Имя пользователя.

Размещение базы данных.

Пароль.

Имя базы данных.

В результате вызова функции создается PHP-переменная со значением ссылки на соединение с базой данных. Используя эту переменную, вы можете обмениваться информацией с базой данных. В нашем примере эта переменная имеет имя `$dbc`, но вы можете дать ей любое другое понравившееся вам имя.

Функция mysqli_connect() рассматривает свои аргументы (имя пользователя и его пароль для сервера баз данных, а также расположение MySQL-сервера и его имя) как строки, поэтому вы должны заключать их в кавычки.

Время поработать

Ниже следует несколько примеров PHP-кода, обеспечивающего соединение с базой данных. Посмотрите на каждый из них и запишите, будет ли он работать, и если не будет, то как можно исправить это положение. Также обведите карандашом каждый проблемный, по вашему мнению, фрагмент кода.

```
$dbc = mysqli_connect('data.aliensubductedme.com', 'owen', 'aliensrool',
'aliensdatabase');
```

```
$dbc = mysqli_connect('data.aliensubductedme.com', 'owen', 'aliensrool',
"aliensdatabase")
```

```
$fangisgone = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool',
  'aliendatabase');
```

```
$dbc = mysqli_connect('localhost', 'owen', 'aliensrool', 'aliendatabase');
```

```
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', '', 'aliendatabase');
```

```
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool');
mysqli_select_db($dbc, 'aliendatabase');
```

Решение задачи

Ниже следует несколько примеров PHP-кода, обеспечивающего соединение с базой данных. Посмотрите на каждый из них и запишите, будет ли он работать, и если не будет, то как можно исправить это положение. Также обведите карандашом каждый проблемный, по вашему мнению, фрагмент кода.

```
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool',
  'aliendatabase');
```

Эта строка будет работать.

Здесь необходима точка с запятой для того, чтобы указать окончание PHP-предложения.

В этой книге мы используем одинарные кавычки и резервируем двойные кавычки для использования их в SQL-запросах.

```
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool',
  "aliendatabase")
```

Эта строка не будет работать, потому что в ее конце отсутствует точка с запятой. Двойные кавычки допускаются, так же как и одинарные.

Не слишком информативное имя для обозначения переменной соединения с базой данных.

```
$fangisgone = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool',
  'aliendatabase');
```

Это будет работать, вот только использовано не слишком информативное имя для переменной соединения с базой данных.

При условии, что сервер баз данных установлен на том же компьютере, что и веб-сервер.

```
$dbc = mysqli_connect('localhost', 'owen', 'aliensrool', 'aliendatabase');
```

Это будет работать при условии, что сервер баз данных установлен на том же компьютере, что и веб-сервер.

Пустой пароль — не слишком удачная идея.

```
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', '', 'aliendatabase');
```

Если вы установите пустой пароль для вашей базы данных, это будет работать. Вот только это далеко не самая удачная идея! Вам следует всегда устанавливать пароль для каждой базы данных.

В случае, если четвертый аргумент не указан здесь, вам необходимо будет выбрать базу данных вызовом функции mysqli_select_db().

```
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool');
mysqli_select_db($dbc, 'aliendatabase');
```

Это вопрос с подвохом! В функции mysqli_connect() четвертый аргумент — имя базы данных — необязателен. Вы можете не передавать его этой функции, а вместо этого указывать имя вашей базы данных в функции mysqli_select(). Таким образом, этот код эквивалентен предыдущему коду, в котором вы указывали все четыре аргумента в функции mysqli_connect().

Это тот случай, когда свои возможности демонстрирует PHP-функция die().

PHP-функция `die()` прерывает ход выполнения сценария и выводит сообщение о коде, который выполнить не удалось. Хотя это и не раскрывает всех причин неудачи в подробностях, функция `die()` все же сообщает нам: что-то случилось и необходимо принимать меры по решению проблемы. Если что-то не так с одним из четырех аргументов, необходимых для соединения функции `mysqli_connect()`, или сервер баз данных не может быть найден, функция `die()` останавливает ход выполнения сценария и выводит сообщение об ошибке, которое передается ей в качестве аргумента в скобках.

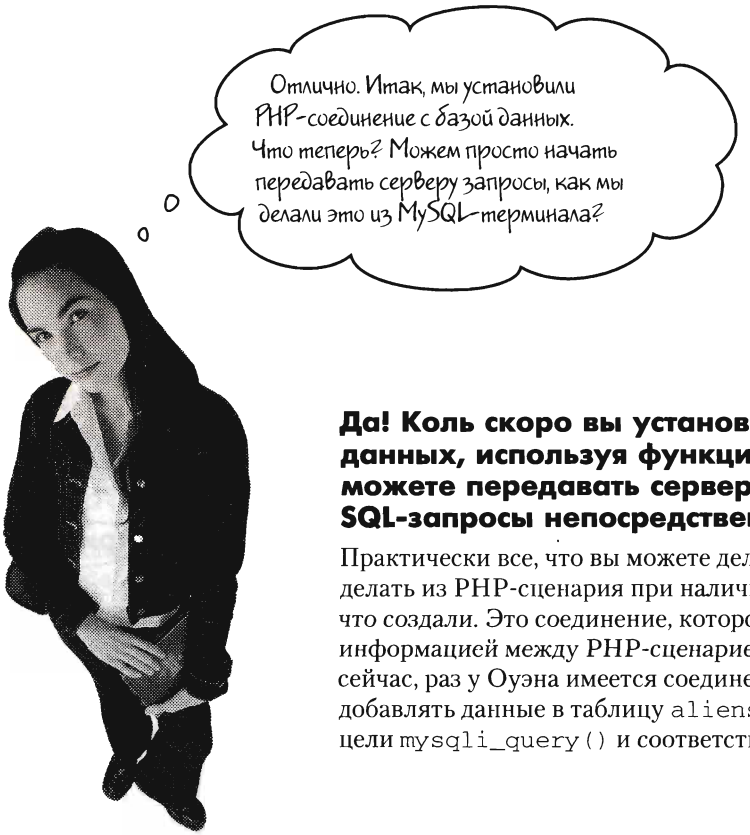
Функция die() вызывается в случае, если соединение не может быть установлено.

Если что-то не так с одним из четырех аргументов, необходимых для соединения функции mysqli_connect(), последует ответная реакция.

```
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool', 'aliendatabase')
   or die ('Ошибка соединения с MySQL-сервером');
```

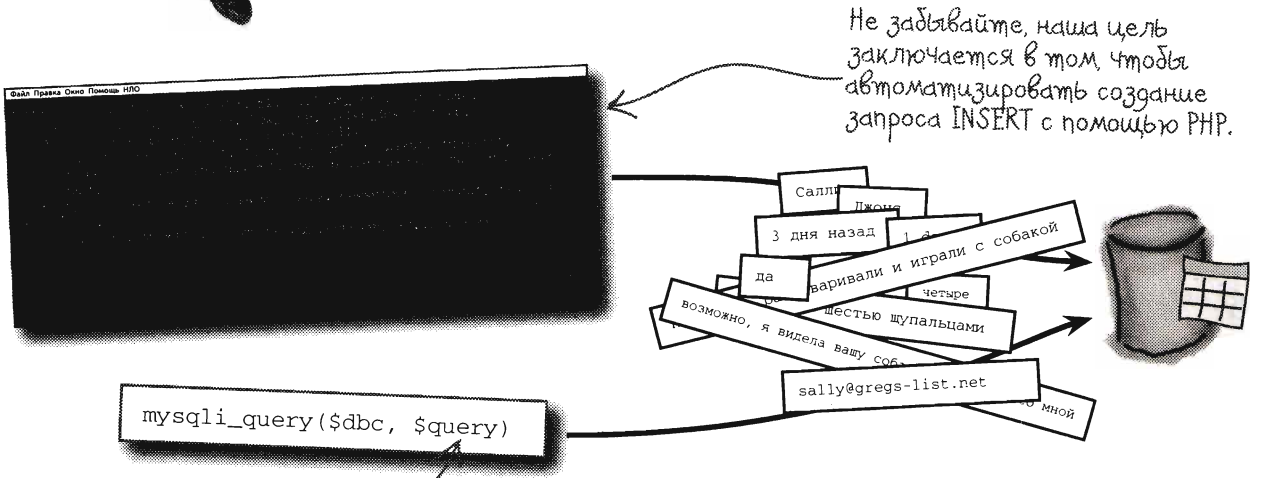
Это сообщение будет продублировано на веб-странице, если соединение не удастся.

Здесь не нужна точка с запятой, так как технически вся строка «mysqli(...) or die(...)» является PHP-предложением, поэтому точка с запятой ставится в конце всей строки.



Да! Коль скоро вы установили соединение с базой данных, используя функцию mysqli_connect(), можете передавать серверу на выполнение SQL-запросы непосредственно из PHP.

Практически все, что вы можете делать в MySQL-терминале, вы можете делать из PHP-сценария при наличии соединения, которое вы только что создали. Это соединение, которое устанавливает канал обмена информацией между PHP-сценарием и базой данных MySQL. Например, сейчас, раз у Оуэна имеется соединение с базой данных, он может начать добавлять данные в таблицу `aliens_abduction`, используя для этой цели `mysqli_query()` и соответствующий SQL-код.



Функции `mysqli_query()` необходимо передать в качестве аргумента SQL-запрос в виде PHP-строки (значения переменной `$query`), для того чтобы произвести добавление данных о похищении космическими пришельцами.

Построение запроса INSERT в PHP

1. ~~Соединение с помощью mysqli_connect().~~
2. Составление строки запроса.
3. Выполнение запроса с помощью mysqli_query().
4. Закрытие соединения с помощью mysqli_close().

SQL-запросы в PHP представлены как текстовые строки и обычно передаются функции `mysqli_query()` в виде строчных переменных. Так как текстовые строки, содержащие SQL-запросы, могут иметь достаточно большую длину, часто возникает необходимость составить строку запроса из нескольких строк меньшего размера. Запрос Оуэна `INSERT` наглядно это демонстрирует.

Это строчная переменная PHP, которая содержит запрос INSERT.

Оператор «точка» говорит PHP соединить эту строку со следующей строкой.

```
$query = "INSERT INTO aliens_abduction (first_name, last_name, " .
    "when_it_happened, how_long, how_many, alien_description, " .
    "what_the_did, fang_spotted, other, email) " .
    "VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре', " .
    "'зеленые с шестью щупальцами', " .
    "'мы просто разговаривали и играли с собакой', " .
    "'да', 'возможно, я видела вашу собаку, свяжитесь со мной', " .
    "'sally@gregs-list.net') ";
```

Строка запроса распределена между несколькими строками кода, чтобы сделать его более удобным для прочтения и понимания. Точки говорят PHP соединить все короткие строки в одну длинную.

Так как суммарно весь код этой строки — это PHP-код, в его конце должна стоять точка с запятой.

После составления SQL-запроса `INSERT` в виде строки текста вы можете передавать его функции `mysqli_query()` для выполнения.

Не бывает глупых вопросов

В: Почему SQL-код INSERT называется запросом? Разве «запрос» (query) не означает, что мы просим что-нибудь у базы данных?

О: Да, «запрос» (query) действительно означает, что вы просите что-нибудь… вы просите базу данных сделать что-нибудь. В MySQL-приложениях баз данных слово «запрос» употребляется в общем смысле, который имеет отношение к любому действию над данными, включая как их добавление, так и извлечение.

В: Почему строка запроса INSERT не создается как одна большая строка?

О: Не забывайте, что строка запроса `INSERT` сохраняется в переменной как одна большая строка, хотя и создается из нескольких строк меньшего размера. В идеале строка запроса `INSERT` могла бы кодироваться в виде одной большой строки, но, как многие другие строки SQL-запросов, строка `INSERT` обычно имеет большую длину и не помещается в строку кода, имеющую «нормальную» длину. Поэтому удобнее рассматривать такую длинную строку запроса составленной из нескольких коротких строк, «склеенных» между собой оператором PHP «точка».

В: Обязательно ли перечислять имена колонок в запросе INSERT?

О: Нет. Вы можете не перечислять имена колонок в запросе `INSERT`. Но в этом случае вы должны перечислить значения для всех колонок таблицы строго в том порядке, в котором они перечислены в структуре таблицы. Принимая это во внимание, более удобно и безопасно перечислять имена колонок в запросе.

Выполнение запроса к базе данных MySQL из PHP

1. ~~Соединение с помощью mysqli_connect().~~
2. ~~Составление строки запроса.~~
3. Выполнение запроса с помощью mysqli_query().
4. Закрытие соединения с помощью mysqli_close().

Функции `mysqli_query()` необходимы два аргумента для выполнения запроса: ссылка на соединение с базой данных и строка SQL-запроса.

```
mysqli_query(ссылка_на_соединение_с_базой_данных, запрос);
```

Это ссылка на соединение с базой данных, которое уже было установлено ранее вызовом функции mysqli_connect().

Это SQL-запрос, который будет выполнен... тот, который мы сохранили в строковой переменной.

Ссылка на соединение с базой данных была возвращена вам ранее функцией `mysqli_connect()`. Если вам не совсем понятно, о чем идет речь, вот код, в результате выполнения которого устанавливается это соединение.

Не забывайте: эти переменные могут отличаться для вашей конфигурации базы данных.

```
$dbc = mysqli_connect('data.aliensubductedme.com', 'owen', 'aliensrool', 'aliensdatabase');
    or die ('Ошибка соединения с MySQL-сервером');
```

Ссылка на соединение с базой данных была сохранена в этой переменной ранее.

Итак, у вас есть соединение с базой данных (переменная `$dbc`) и SQL-запрос (переменная `$query`). Все, что осталось, — это только передать их функции `mysqli_query()` в качестве аргументов.

Это SQL-запрос.

```
$result = mysqli_query($dbc, $query);
    or die('Error querying database.');
```

Это ссылка на соединение с базой данных.

Это результат запроса.

SQL-запрос — это требование, написанное на языке SQL и отправленное на сервер баз данных для выполнения.

Этот код показывает, что вызов `mysqli_query()` не является односторонним процессом. Функция возвращает вам информацию, содержание которой присваивает переменной `$result`. Но никаких реальных данных в результате выполнения запроса `INSERT` не возвращается. Эта переменная только информирует вас о том, удачно или неудачно завершился вызов функции `mysqli_query ()`.

Функции mysqli_query( ) необходимы ссылка на соединение с базой данных и строка SQL-запроса, для того чтобы выполнить этот запрос.

Закрытие соединения с помощью функции mysqli_close()

1. ~~Соединение с помощью mysqli_connect().~~
2. ~~Составление строки запроса.~~
3. ~~Выполнение запроса с помощью mysqli_query().~~
4. Закрытие соединения с помощью mysqli_close().

Так как мы заинтересованы в исполнении только запроса `INSERT`, обмен информацией на этом заканчивается, по крайней мере, для сценария. А раз вы завершили работу с базой данных, соединение с ней необходимо закрыть. Соединение с базой данных будет закрыто автоматически, как только посетитель сайта перейдет с текущей страницы на новую, но, как привычка закрывать за собой дверь, закрытие соединения после окончания работы с ним считается хорошим стилем программирования. Функция `mysqli_close()` закрывает соединение с базой данных.

Хорошая привычка — закрывать соединение с базой данных MySQL сразу же после окончания работы с ней.

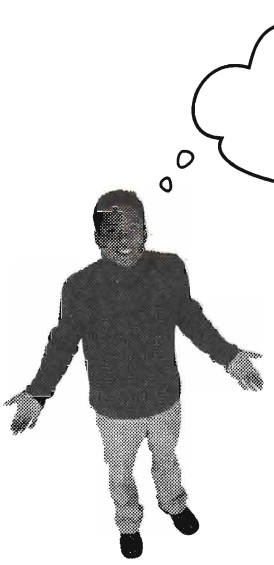
```
mysqli_close(ссылка_на_соединение_с_базой_данных);
```

Это ссылка на соединение с базой данных, которое вы использовали для обмена информацией с ней.

В случае со сценарием Оуэна нам необходимо передать функции `mysqli_close()` в качестве аргумента ссылку на соединение с базой данных, которая сохранена в переменной `$dbc`.

```
mysqli_close($dbc);
```

Значением этой переменной является ссылка на соединение с базой данных, которое было создано вызовом функции mysqli_connect() ранее, когда оно открывалось впервые.



Сервер баз данных выделяет ограниченное количество соединений, доступных одновременно, поэтому чем раньше они будут освобождаться, тем лучше.

Как только вы закрываете соединение, выделенные для него ресурсы освобождаются и могут быть использованы для создания нового соединения. Если вы работаете с разделяемой базой данных, вам будет выделено, например, пять соединений. И если вы создадите в таком окружении еще одно приложение с использованием баз данных, вам может не хватить доступных соединений, так как они не будут вовремя освобождаться.

не бывает глупых вопросов

В: Разве мы не можем просто записать весь SQL-код непосредственно в функцию mysqli_query() вместо переменной $query?

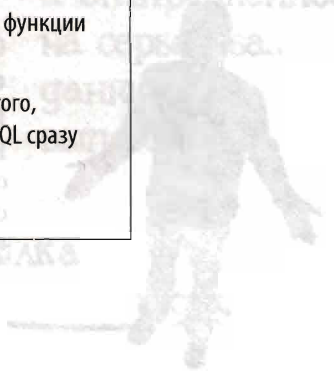
О: Можете, но это затрудняет чтение и понимание кода. Представление SQL-кода в виде строчной переменной с последующей передачей этой переменной функции `mysqli_query()` в качестве аргумента делает ваш код немного легче для прочтения и понимания, что упрощает его сопровождение.

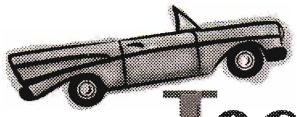
В: Должен ли код, в котором используется запрос INSERT, делать что-либо с результатом, возвращаемым функцией mysqli_query()?

О: Скорее всего, да. Пока что мы использовали `die()` для того, чтобы прервать ход выполнения сценария и передать браузеру сообщение об ошибке, в случае если что-то идет не так. Со временем вы, возможно, захотите передать пользователю более подробную информацию, если выполнение запроса закончилось неудачей. В этом случае вы можете использовать результат запроса для определения причин неудачи.

КЛЮЧЕВЫЕ МОМЕНТЫ

- Для создания соединения необходимо определить размещение сервера баз данных, имя пользователя и его пароль, а также имя базы данных.
- Функция `mysqli_connect()` создает соединение между вашим PHP-сценарием и сервером баз данных MySQL.
- Функция `die()` прерывает ход выполнения сценария и передает браузеру сообщение об ошибке, если создать соединение не удалось.
- Выполнение SQL-запроса из сценария PHP включает представление запроса в виде текстовой строки с последующей передачей его в качестве аргумента функции `mysqli_query()`.
- Вызывайте функцию `mysqli_close()` для того, чтобы закрыть PHP-соединение с базой данных MySQL сразу же после окончания работы с ней.





Тест-драйв

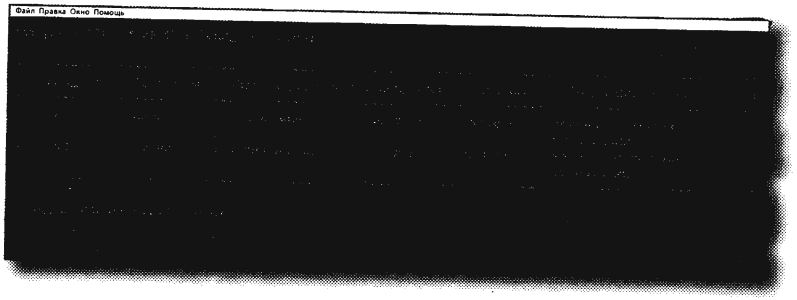
Замените код с электронной почтой в сценарии Оуэна report.php так, чтобы он добавлял данные в базу данных MySQL, и проверьте его работу.

Удалите код в сценарии Оуэна `report.php`, который отправляет данные формы по электронной почте. На его место добавьте код, который вызывает соединение с базой данных MySQL, создает SQL-запрос в виде PHP-строки, передает запрос серверу баз данных на выполнение и, наконец, закрывает соединение.

Это новый PHP-код по обмену информацией с базой данных, над которым вы работали. Не вводите теги <?php ?> в сценарий report.php, так как вы добавляете этот код в то место сценария, которое уже находится внутри этих тегов.

```
<?php
   $dbc = mysqli_connect('data.aliensubductedme.com', 'owen', 'aliensrool',
'aliensdatabase')
      or die ('Ошибка соединения с MySQL-сервером');

   $query = "INSERT INTO aliens_abduction (first_name, last_name, " .
      "when_it_happened, how_long, how_many, alien_description, ".
      "what_the_did, fang_spotted, other, email) " .
      "VALUES ('Салли', 'Джонс', '3 дня назад', '1 день', 'четыре', " .
      "'зеленые с шестью щупальцами', 'мы просто разговаривали и играли с собакой', " .
      "'да', 'возможно, я видела вашу собаку, свяжитесь со мной', 'sally@gregs-list.net')";

   $result = mysqli_query($dbc, $query)
      or die ('Ошибка при выполнении запроса к базе данных.');

   mysqli_close($dbc)
?>
```

Загрузите новый файл `report.php` на ваш веб-сервер и откройте страницу `report.html` в браузере, чтобы получить доступ к форме «Сообщения о похищениях космическими пришельцами». Заполните поля формы и нажмите кнопку «Сообщение о похищении», чтобы сохранить данные в базе. Затем запустите свою инструментальную программу MySQL и выполните запрос `SELECT`, чтобы увидеть изменения, произошедшие в базе данных.



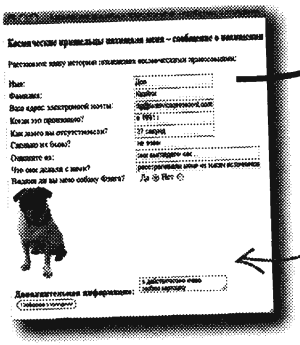
Это правильно? Напишите, как вы думаете: то ли это, что сценарий должен был сделать, и почему.

Подождите секунду. Разве главная задача не заключается в том, чтобы взять данные в форме и сохранить их в базе данных? Похоже, по запросу вводятся одни и те же данные независимо от того, что было введено в форму. Я не вижу, как этот PHP-сценарий автоматизирует процесс.

Это большая проблема. Запрос INSERT должен быть составлен так, чтобы добавлять данные формы, а не статичные строки.

В запрос, который мы составили, включены жестко запрограммированные строки вместо текстовых данных, введенных в форму о похищении космическими пришельцами. Для того чтобы сценарий работал с формой, необходимо передавать данные из полей формы в строку запроса.

Этим данным необходимо попасть в строку запроса $query.



Форма о похищении космическими пришельцами — это место, откуда поступают данные посетителей сайта.

СИЛА МЫСЛИ

Какой PHP-код может помочь нам передать значения полей формы в INSERT-запрос?

$_POST передает данные формы

Хорошая новость заключается в том, что сценарий `report.php` содержит данные формы, сохраненные в суперглобальной переменной `$_POST`. Помните этот код?

```
$name = $_POST['firstname'] . ' ' . $_POST['lastname'];
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$how_many = $_POST['howmany'];
$alien_description = $_POST['aliendescription'];
$what_they_did = $_POST['whattheydid'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
$other = $_POST['other'];
```

Суперглобальную переменную $_POST мы уже использовали, чтобы получить значения каждого поля формы данных Оуэна и присвоить эти значения соответствующим переменным.

Не забывайте: имя, которое вы используете в качестве индекса в переменной $_POST, должно соответствовать имени поля HTML-формы.

Таким образом, раз вы уже имеете под рукой данные формы, вам осталось только включить их в строку запроса `INSERT` о похищении космическими пришельцами. Только вначале вам нужно будет сделать небольшое изменение. Раз вы теперь не будете отправлять данные формы по электронной почте, вам больше не нужна переменная `$email`. Вам все еще необходимы имя и фамилия посетителя сайта, чтобы добавить эти данные в базу, но в отдельных переменных.

```
$first_name = $_POST['firstname'];
$last_name = $_POST['lastname'];
```

Значения имени и фамилии посетителя сайта теперь присвоены индивидуальным переменным, поэтому они могут быть сохранены каждое в своей колонке таблицы aliens_abduction.

УПРАЖНЕНИЕ

Напишите PHP-код для создания строки запроса `INSERT` Оуэна и присвойте ее значение переменной `$query`. В результате выполнения этого запроса в таблицу `aliens_abduction` должны быть добавлены действительные данные формы.

.....

.....

.....

.....

РЕШЕНИЕ К УПРАЖНЕНИЮ

Напишите PHP-код для создания строки запроса `INSERT` Оуэна и присвойте ее значение переменной `$query`. В результате выполнения этого запроса в таблицу `aliens_abduction` должны быть добавлены действительные данные формы.

Имена колонок появились в SQL-запросе точно так же, как и прежде.

```
$query = "INSERT INTO aliens_abduction (first_name, last_name, when_it_happened, how_long, " .
  "how_many, alien_description, what_they_did, fang_spotted, other, email) " .
  "VALUES ('$first_name', '$last_name', '$when_it_happened', '$how_long', '$how_many', " .
  "'$alien_description', '$what_they_did', '$fang_spotted', '$other', '$email')";
```

Вместо статичных данных о похищении Салли Джонс теперь мы вставляем те данные, которые посетитель сайта ввел в форму.

Порядок следования переменных должен строго соответствовать порядку следования имен колонок в таблице, чтобы каждое значение было сохранено в своей колонке.

не бывает глупых вопросов

В: Должен ли я создавать все эти переменные, предназначенные для сохранения данных, содержащихся в $_POST? Разве нельзя просто сослаться на $_POST непосредственно в строке $query?

О: Вы можете. Нет ничего такого, что бы препятствовало ссылаться на `$_POST` в запросе непосредственно. Тем не менее считается хорошим стилем программирования изолировать данные формы, прежде чем манипулировать ими. Это важно потому, что достаточно распространенной практикой является обработка данных формы перед сохранением их в базе данных. Например, существуют изощренные хакерские методы, предназначенные для того, чтобы перехватить ваши запросы путем ввода в форму потенциально опасных данных. Вы узнаете, как предотвращать такие попытки, в главе 6. Для упрощения в этой главе мы не будем никаким образом обрабатывать данные формы, но это не означает, что вам не следует думать о будущем и взять себе в привычку сохранять данные формы в своих собственных переменных, прежде чем ссылаться на них в запросах.

В: Хорошо. Имеет ли значение, в каких случаях используются одинарные кавычки, а в каких — двойные? Могу ли я ограничивать одинарными кавычками весь запрос целиком, а двойными — переменные?

О: Да, это имеет значение. И вы не можете ограничивать одинарными кавычками весь запрос целиком, а двойными — переменные. Причина в том, что PHP обрабатывает строки по-разному в зависимости от того, ограничены они одинарными кавычками или двойными. Разница между этими двумя подходами заключается в том, что одинарные кавычки представляют текст, заключенный между ними, буквально, как есть, в то время как над текстом, заключенным в двойные кавычки, производятся дополнительные действия. В результате этих действий имена переменных, находящихся между двойными кавычками, заменяются на их значения, что достаточно удобно и объясняет тот факт, что двойные кавычки обычно более предпочтительны при построении строк SQL-запросов.

В: Разве нельзя создавать строку запроса простой конкатенацией переменных и SQL-кода?

О: Можно, и если бы вы последовали этой схеме, то могли использовать одинарные кавычки вместо двойных. Но строки запросов часто бывают очень сложными, поэтому все, что вы предпринимаете, чтобы сделать их более легкими для прочтения и понимания, — это хорошая практика. Строки запросов, ограниченные двойными кавычками, с непосредственно встроенными в них переменными, удобнее для понимания, чем конкатенации переменных и SQL-кода, ограниченного одинарными кавычками.

УПРАЖНЕНИЕ

Давайте, используя все, что мы узнали, закончим сценарий Оуэна по обработке данных формы, чтобы он мог успешно сохранять в базе данных информацию о похищениях космическими пришельцами. Закончите PHP-код, чтобы завершить сценарий `report.php`.

```
<?php
  ..........................................................

  ..........................................................
  $when_it_happened = $_POST['whenithappened'];
  $how_long = $_POST['howlong'];
  $how_many = $_POST['howmany'];
  $alien_description = $_POST['aliendescription'];
  $what_they_did = $_POST['whattheydid'];
  $fang_spotted = $_POST['fangspotted'];
  $email = $_POST['email'];
  $other = $_POST['other'];

  $dbc = ..........................................................................................

    ..........................................................................................

  $query = "INSERT INTO aliens_abduction (first_name, last_name, when_it_happened, how_long, "
    "how_many, alien_description, what_they_did, fang_spotted, other, email) " .
    "VALUES ('$first_name', '$last_name', '$when_it_happened', '$how_long', '$how_many', " .
    "'$alien_description', '$what_they_did', '$fang_spotted', '$other', '$email')";

  $result = ..................................................

    ..........................................................

  ..................................

    echo 'Спасибо за заполнение формы.<br />';
    echo 'Вы были похищены ' . $when_it_happened;
    echo ' и отсутствовали в течение ' . $how_long . '<br />';
    echo 'Количество космических пришельцев: ' . $how_many . '<br />';
    echo 'Опишите их: ' . $alien_description . '<br />';
    echo 'Что они делали с вами? ' . $what_they_did . '<br />';
    echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
    echo 'Дополнительная информация: ' . $other . '<br />';
    echo 'Ваш адрес электронной почты ' . $email;
?>
```

РЕШЕНИЕ К УПРАЖНЕНИЮ

Давайте, используя все, что мы узнали, закончим сценарий Оуэна по обработке данных формы, чтобы он мог успешно сохранять в базе данных информацию о похищениях космическими пришельцами. Закончите PHP-код, чтобы завершить сценарий `report.php`.

Две новые переменные содержат имя и фамилию посетителя сайта, как было введено в форме.

```
<?php
  $first_name = $_POST['firstname'];
  $last_name = $_POST['lastname'];
  $when_it_happened = $_POST['whenithappened'];
  $how_long = $_POST['howlong'];
  $how_many = $_POST['howmany'];
  $alien_description = $_POST['aliendescription'];
  $what_they_did = $_POST['whattheydid'];
  $fang_spotted = $_POST['fangspotted'];
  $email = $_POST['email'];
  $other = $_POST['other'];
```

Вы должны создать соединение с базой данных, передав информацию, необходимую для него, прежде чем отправлять на выполнение любой запрос.

Запрос создан в виде PHP-строки с использованием данных, введенных в форму.

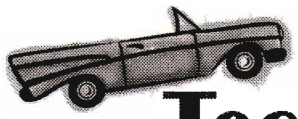
```
  $dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool', 'aliendatabase')
    or die ('Ошибка соединения с MySQL-сервером');

  $query = "INSERT INTO aliens_abduction (first_name, last_name, when_it_happened, how_long, "
    "how_many, alien_description, what_they_did, fang_spotted, other, email) " .
    "VALUES ('$first_name', '$last_name', '$when_it_happened', '$how_long', '$how_many', " .
    "'$alien_description', '$what_they_did', '$fang_spotted', '$other', '$email')";

  $result = mysqli_query($dbc, $query)
    or die ('Ошибка при выполнении запроса к базе данных.');

  mysqli_close($dbc);
```

В результате выполнения запроса на сервере данные сохраняются в базе.

Закрытие соединения с базой данных.

```
   echo 'Спасибо за заполнение формы.<br />';
   echo 'Вы были похищены ' . $when_it_happened;
   echo ' и отсутствовали в течение ' . $how_long . '<br />';
   echo 'Количество космических пришельцев: ' . $how_many . '<br />';
   echo 'Опишите их: ' . $alien_description . '<br />';
   echo 'Что они делали? ' . $what_they_did . '<br />';
   echo 'Видели ли вы мою собаку Фэнга? ' . $fang_spotted . '<br />';
   echo 'Дополнительная информация: ' . $other . '<br />';
   echo 'Ваш адрес электронной почты ' . $email;
?>
```

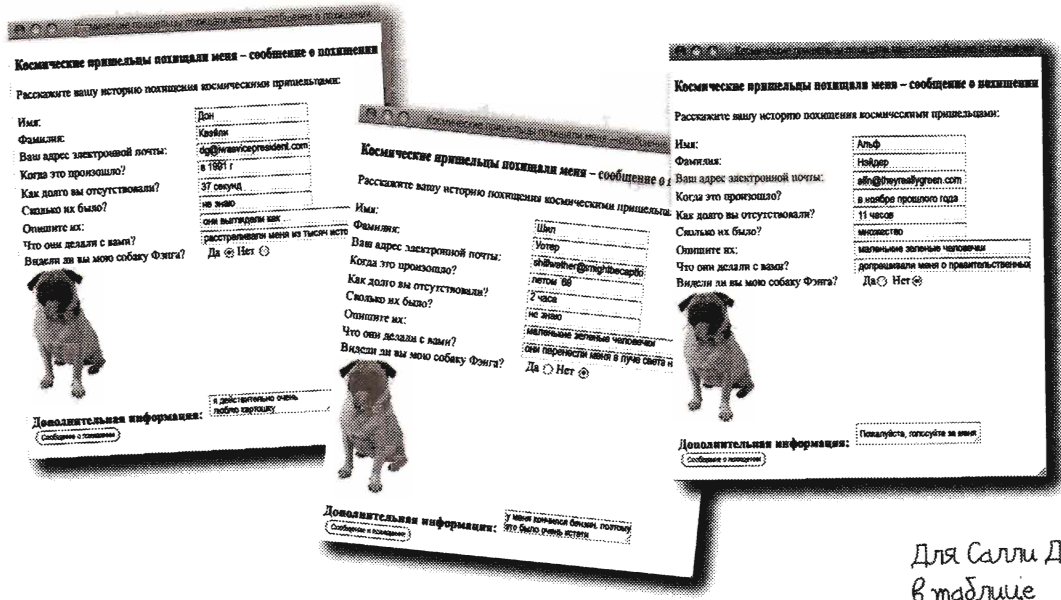
Подтвердите успешное сохранение данных формы в базе данных, точно так же как вы делали это в старом сценарии.



Тест-драйв

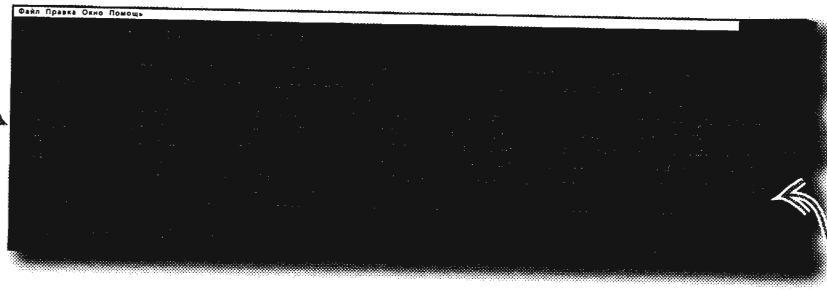
Измените сценарий Оуэна report.php так, чтобы он использовал действительные данные формы при выполнении запроса INSERT.

Удалите переменную `$name` в сценарии `report.php`, добавьте переменные `$first_name` и `$last_name`, а также значение переменной `$query` так, чтобы вместо статичного текста в строке запроса `INSERT` использовались данные формы. Загрузите новую версию сценария и проверьте его работу, вызывая форму несколько раз и вводя в нее каждый раз новые данные.



А теперь в MySQL-терминале введите запрос `SELECT` и проверьте содержимое таблицы `aliens_abduction`.

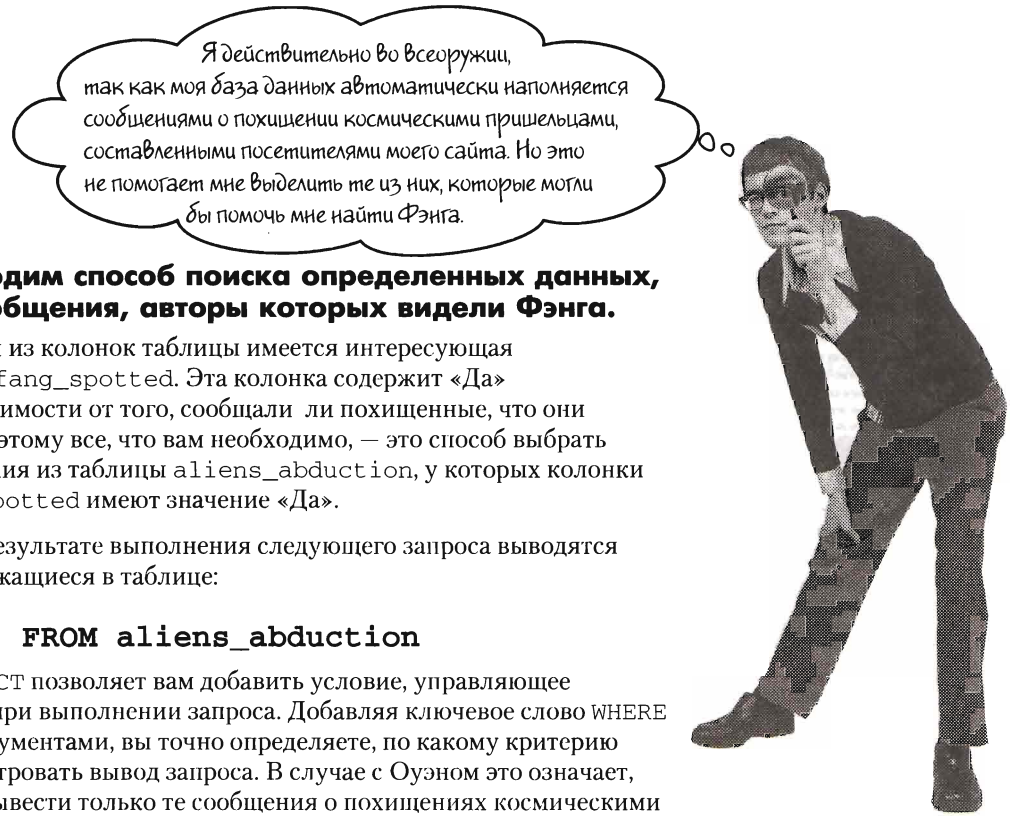
Как вы и ожидали, новые сообщения о похищении космическими пришельцами появились в таблице.



Для Салли Джонс в таблице имеется одна лишняя строка данных, которая появилась до того, как вы откорректировали запрос INSERT. Не волнуйтесь, в следующей главе вы узнаете, как удалять ненужные записи.

Оуэну необходима помощь в «просеивании» его данных

Новый и улучшенный сценарий `report.php` делает свое дело и автоматизирует процесс добавления в базу данных сообщений о похищении космическими пришельцами. Оуэн может просто сидеть, откинувшись на спинку стула, давая возможность сообщениям собираться в... но вот опять проблема! Поступление большего количества информации вовсе не приводит автоматически к увеличению шансов найти среди всего этого множества сообщения о похищении космическими пришельцами, имеющие отношение к потенциальному обнаружению Фэнга.



Оуэну необходим способ поиска определенных данных, таких как сообщения, авторы которых видели Фэнга.

Вы знаете, в какой из колонок таблицы имеется интересующая нас информация: `fang_spotted`. Эта колонка содержит «Да» или «Нет» в зависимости от того, сообщали ли похищенные, что они видели Фэнга. Поэтому все, что вам необходимо, — это способ выбрать только те сообщения из таблицы `aliens_abduction`, у которых колонки с именем `fang_spotted` имеют значение «Да».

Вы знаете, что в результате выполнения следующего запроса выводятся все данные, содержащиеся в таблице:

```
SELECT * FROM aliens_abduction
```

SQL-запрос `SELECT` позволяет вам добавить условие, управляющее выводом данных при выполнении запроса. Добавляя ключевое слово `WHERE` (англ. «где») с аргументами, вы точно определяете, по какому критерию вы хотите отфильтровать вывод запроса. В случае с Оуэном это означает, что необходимо вывести только те сообщения о похищениях космическими пришельцами, в которых колонка `fang_spotted` имеет значение «Да».

Не забывайте, что без условия WHERE в результате выполнения запроса будут выведены все данные, содержащиеся в таблице.

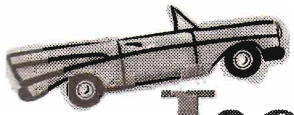
Должно быть указано значение колонки для вывода данных только с этим значением.

Имя колонки.

```
SELECT * FROM aliens_abduction WHERE fang_spotted = 'yes'
```

Эта часть запроса SELECT остается без изменения — условие WHERE отвечает за фильтрацию результата.

Это условие сокращает количество выводимых данных, оставляя только те из них, в которых колонка fang_spotted имеет значение «Да».



Тест-драйв

Проверьте выполнение запроса SELECT с условием WHERE, устанавливающим критерии поиска.

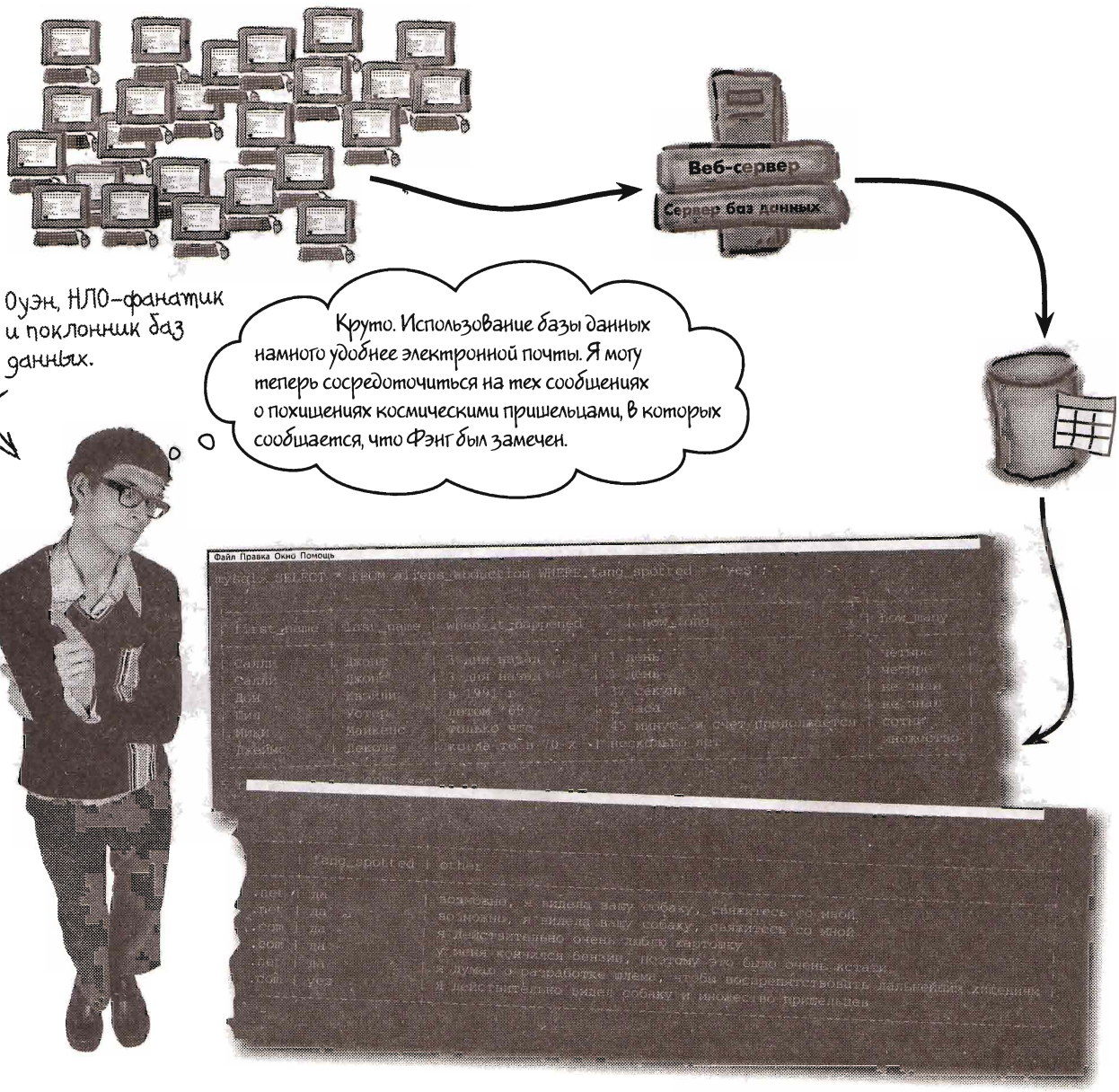
Используйте в своем MySQL-терминале запрос `SELECT` с условием `WHERE` для поиска только тех сообщений о похищениях космическими пришельцами, в которых говорится о том, что Фэнг был замечен.

```
Файл Правка Окно Помощь
mysql> SELECT * FROM aliens_abduction WHERE fang_spotted = 'yes';
+------------+-----------+-----------------+-----------------------------------+----------+
| first_name | last_name | when_it_happened | how_long                          | how_many |
+------------+-----------+-----------------+-----------------------------------+----------+
| Салли      | Джонс     | 3 дня назад     | 1 день                            | четыре   |
| Салли      | Джонс     | 3 дня назад     | 1 день                            | четыре   |
| Дон        | Квэйли    | в 1991 г.       | 37 секунд                         | не знаю  |
| Шил        | Уотер     | летом '69       | 2 часа                            | не знаю  |
| Мики       | Майкенс   | только что      | 45 минут… и счет продолжается     | сотни    |
+------------+-----------+-----------------+-----------------------------------+----------+
5 рядов в наборе (0.0005 сек)

+-----+--------------+-----------------------------------------------------------------+
|     | fang_spotted | other                                                           |
+-----+--------------+-----------------------------------------------------------------+
| .net | да          | возможно; я видела вашу собаку, свяжитесь со мной               |
| .net | да          | возможно, я видела вашу собаку, свяжитесь со мной               |
| .com | да          | я действительно очень люблю картошку                            |
| .com | да          | у меня кончился бензин, поэтому это было очень кстати           |
| .net | да          | я думаю о разработке шлема, чтобы воспрепятствовать дальнейшим похищениям |
+-----+--------------+-----------------------------------------------------------------+
```

Во всех этих записях колонки fang_spotted имеют значение «Да».

Оуэн на пути обнаружения Фэнга

Благодаря PHP и его функциям по обмену информацией с MySQL сервер баз данных Оуэна получает данные HTML-формы и сохраняет их в таблице базы данных. Эти данные терпеливо и в безопасности ожидают момента, пока у Оуэна не появится необходимость перебрать их. И как только он будет к этому готов, все, что ему понадобится для того, чтобы выбрать только те сообщения о похищениях космическими пришельцами, которые, возможно, имеют отношение к Фэнгу, — это простой запрос SELECT.

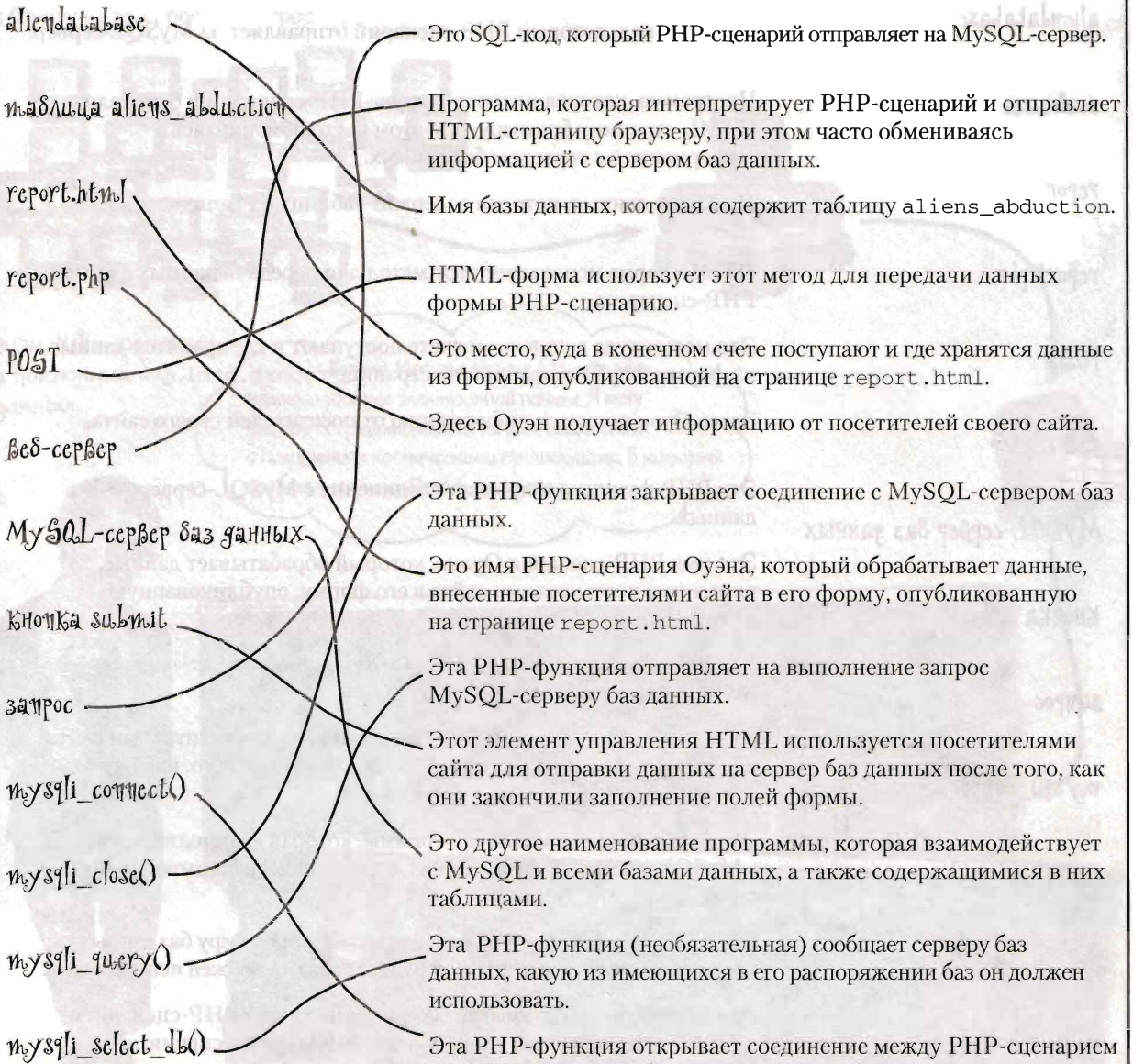


Попробуйте определить, что делают эти HTML-, PHP- и MySQL-компоненты.

aliendatabase	Это SQL-код, который PHP-сценарий отправляет на MySQL-сервер.
таблица aliens_abduction	Программа, которая интерпретирует PHP-сценарий и отправляет HTML-страницу браузеру, при этом часто обмениваясь информацией с сервером баз данных.
report.html	Имя базы данных, которая содержит таблицу `aliens_abduction`.
report.php	HTML-форма использует этот метод для передачи данных формы PHP-сценарию.
POST	Это место, куда в конечном счете поступают и где хранятся данные из формы, опубликованной на странице `report.html`.
веб-сервер	Здесь Оуэн получает информацию от посетителей своего сайта.
MySQL-сервер баз данных	Эта PHP-функция закрывает соединение с MySQL-сервером баз данных.
кнопка submit	Это имя PHP-сценария Оуэна, который обрабатывает данные, внесенные посетителями сайта в его форму, опубликованную на странице `report.html`.
запрос	Эта PHP-функция отправляет на выполнение запрос MySQL-серверу баз данных.
mysqli_connect()	Этот элемент управления HTML используется посетителями сайта для отправки данных на сервер баз данных после того, как они закончили заполнение полей формы.
mysqli_close()	Это другое наименование программы, которая взаимодействует с MySQL и всеми базами данных, а также содержащимися в них таблицами.
mysqli_query()	Эта PHP-функция (необязательная) сообщает серверу баз данных, какую из имеющихся в его распоряжении баз он должен использовать.
mysqli_select_db()	Эта PHP-функция открывает соединение между PHP-сценарием и MySQL-сервером баз данных, таким образом обеспечивая им возможность обмениваться информацией.

Что к чему. Решение

Попробуйте определить, что делают эти HTML-, PHP- и MySQL-компоненты.



не бывает глупых вопросов

В: Это очень хорошо, что я узнал про добавление данных в таблицу MySQL, но мне все еще непонятно, как создаются таблицы и базы данных, в которых они содержатся. Как это происходит?

О: Хороший вопрос. Вы действительно должны понимать, как создавать свои собственные таблицы, а не просто использовать кем-то данный вам код. Пока что вы создали таблицу без особого понимания синтаксиса запроса CREATE TABLE. Это хорошо для единственной таблицы Оуэна, но когда у вас возникнет необходимость в создании множества таблиц с собственноручно разработанной структурой, этих знаний будет недостаточно. Вам будет необходимо рассмотреть более внимательно все данные, которые вы намерены сохранять в базе, и серьезно подумать о наилучшем способе их представления. Это главная цель следующей главы... Вы готовы?

Создание ваших собственных данных

У вас не всегда будут необходимые вам данные.

Иногда вам придется создавать данные, прежде чем вы сможете их использовать. А иногда вам придется создавать таблицы, чтобы хранить эти данные. А иногда вам придется создавать базу данных, чтобы хранить данные, которые вам нужно создавать, прежде чем вы сможете их использовать. Запутались? Разберетесь! Приготовьтесь узнать, как создавать свои собственные базы данных и таблицы. И после всего этого вы создадите свое первое PHP- и MySQL-приложение.

Магазин «Элвис» открыт

Элмер Пристли открыл свой магазин «Элвис»: `MakeMeElvis.com`. Спрос огромен. Он продал множество полиэстеровых комбинезонов в обтяжку с заклепками, уйму искусственных баков и сотни солнцезащитных очков.

Каждый раз, когда кто-нибудь покупал у Элмера что-либо, он оставлял новый адрес электронной почты. Элмер использовал эти адреса, для того чтобы рассылать информационные бюллетени о распродажах в своем магазине. Пока что ему приходится вручную находить каждый адрес электронной почты в своем списке и копировать его в адресную строку электронного письма при отправлении объявлений о распродажах. Это работает, но требует слишком много времени и усилий.

У Элмера есть 328 адресов электронной почты, собранных на данный момент, но их количество растет с каждым днем.

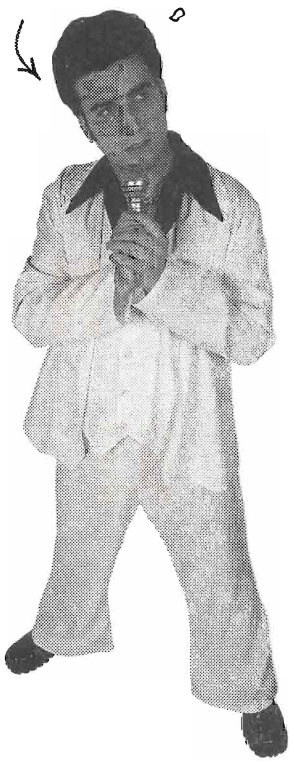
Список адресов электронной почты покупателей Элмера:

Андерсон	Джилиан	jill_anderson@breakneckpizza.com
Джоф	Кевин	joffe@simuduck.com
Ньюсом	Аманда	aman2luv@breakneckpizza.com
Гарсия	Эд	ed99@b0tt0msup.com
Раундтри	Джо-Ан	jojoround@breakneckpizza.com
Бриггс	Крис	cbriggs@boards-r-us.com
Харт	Ллойд	hovercraft@breakneckpizza.com
Тот	Анна	AnneToth@feapinlimos.com
Уилли	Эндрю	andrewwilley@objectville.net
Палумбо	Том	palofmine@mightygumboll.net
Райан	Алама	angrypirate@breakneckpizza.com
Маккинли	Клэй	clay@starbuzzcoffee.com
Микер	Эн	annmeeker@chocoholic-inc.com
Пауэрс	Брайан	bp@honey-doit.com
Мэнсон	Анна	am86@objectville.net
Мендель	Дебра	debmonster@breakneckpizza.com
Тедеско	Дженис	janistedesco@starbuzzcoffee.com
Талвар	Викрам	vikt@starbuzzcoffee.com
Звед	Джо	szvedjoe@objectville.net
Шеридан	Дайана	sheridi@mightygumboll.net
Сноу	Эдвард	snowman@tikibeanlounge.com
Отто	Глен	glenn0098@objectville.net
Харди	Анна	anneh@b0tt0msup.com
Дил	Мэри	nobigdeal@starbuzzcoffee.com
Джагел	Эн	dreamgirl@breakneckpizza.com
Мелфи	Джеймс	drmelfi@b0tt0msup.com
Оливер	Ли	leeoliver@weatherorama.com
Паркер	Анна	annep@starbuzzcoffee.com
Риччи	Питер	ricciman@tikibeanlounge.com
Рено	Грейс	grace23@objectville.net
Мосс	Зельда	zelda@weatherorama.com
Дэй	Клиффорд	clifnight @breakneckpizza.com
Болгер	Джойс	joyce@chocoholic-inc.com
Блант	Анна	anneblant@breakneckpizza.com
Ботлинг	Линди	lindy@tikibeanlounge.com
Гэйрс	Фред	fgares@objectville.net
Джэкобс	Анна	anne99@objectville.net

Элмер, бесспорный король онлайновой продажи вещей Элвиса.

Это занимает слишком много времени. Лучше бы я тратил свое время на подражание Элвису, а не на ручную отправку электронных писем!

Элмер пишет вот такое электронное письмо и копирует каждый адрес электронной почты в строку «кому».



Новое электронное письмо

Кому:

Тема: Большая распродажа!

Дорогой друг Элвисониан!

На этой неделе в MakeMeElvis.com будет большая распродажа!

Баки из натурального конского волоса со скидкой 20 %!

И не забывай: покупаешь один полиэстеровый костюм — берешь второй бесплатно! Осталось только три дня!

Эти люди включены в список адресов электронной почты Элмера и с нетерпением ожидают, когда смогут с его помощью еще больше походить на Элвиса.

Элмер тратит слишком много времени на то, чтобы вручную находить каждый адрес электронной почты в списке и копировать его в адресную строку электронного письма своей почтовой программы. Он хотел бы упростить задачу заполнения строки адреса и отправления такого огромного количества электронных писем.

Элмеру необходимо приложение

Приложение — это программа, разработанная для выполнения определенной пользовательской задачи. Элмеру необходимо приложение, которое бы содержало в порядке его список адресов электронной почты и позволяло ему простым нажатием кнопки, размещенной на форме, отправлять электронные письма людям, включенным в этот список. Он хотел бы, чтобы это выглядело примерно так:

- ☑ **Открыть веб-страницу и ввести в форму содержание электронного письма.**
- ☑ **Нажать кнопку отправки формы на сервер для обработки, в результате чего это электронное письмо будет отправлено всем посетителям сайта MakeMeElvis.com, чьи адреса электронной почты имеются в списке.**
- ☑ **Автоматически обновлять список адресов электронной почты, предоставив возможность покупателям регистрироваться через веб-форму.**

Веб-приложение — это динамический сайт, который разработан, чтобы решать определенные задачи для посетителей этого сайта.

Это приложение для рассылки электронной почты напоминает приложение Оуэна о похищениях космическими пришельцами, но отличается тем, что в нем список адресов электронной почты Элмера будет обновляться автоматически и его электронные письма будут отправляться всем, кто перечислен в этом списке. Приложение Элмера практически полностью автоматизировано!

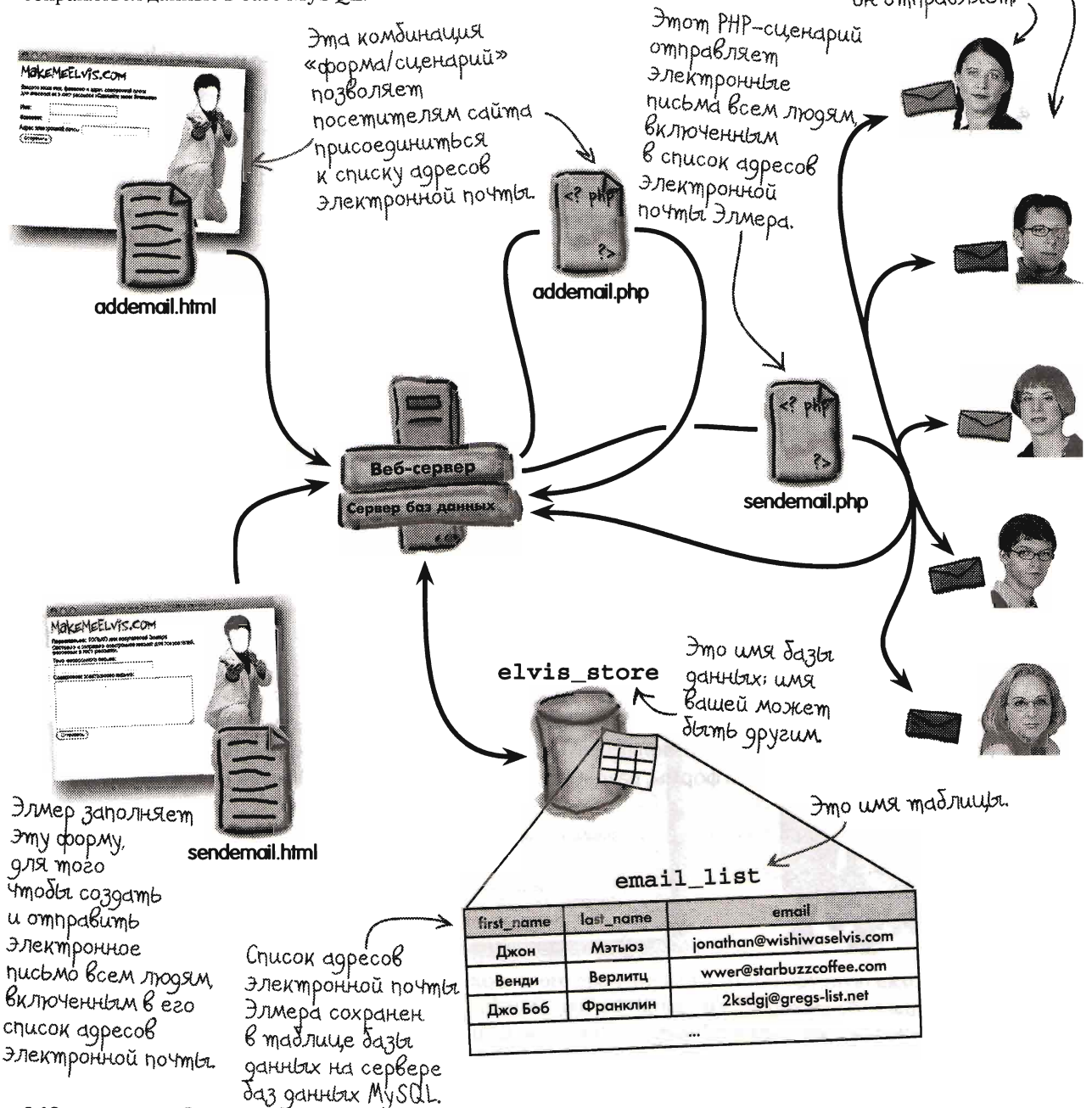
С перечнем задач, которые должно решать это приложение, у Элмера появляется возможность показать его во всем блеске...

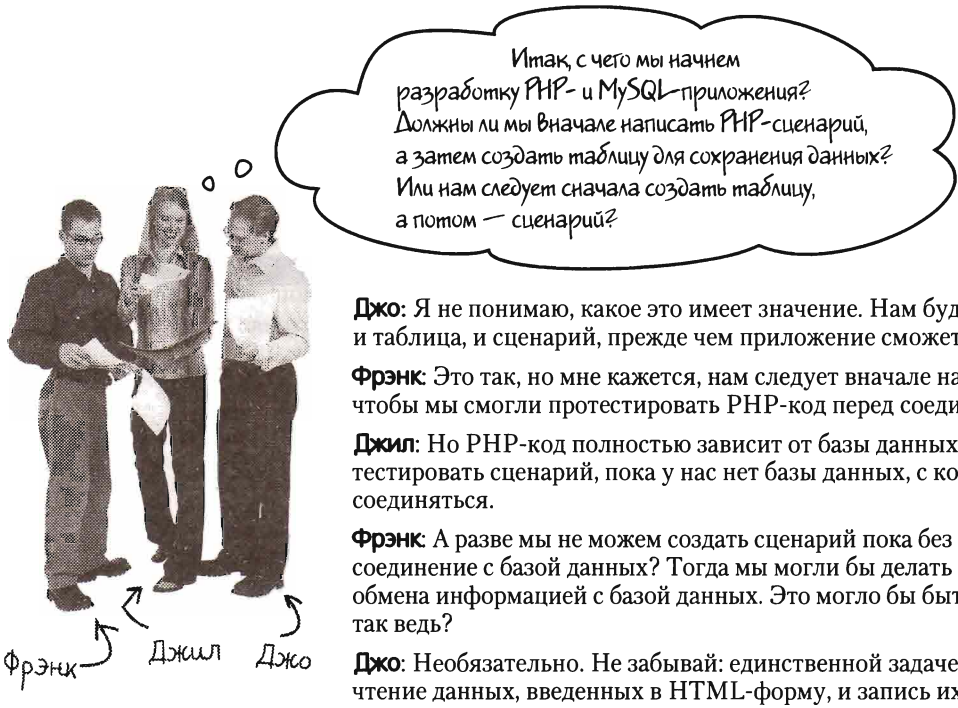
СИЛА МЫСЛИ

Приложение `MakeMeElvis.com` состоит из двух основных компонентов: веб-формы для составления электронного письма людям, перечисленным в списке адресов электронной почты Элмера, и веб-формы, позволяющей новым покупателям присоединиться к этому списку. Имея в виду две эти формы, разработайте схему приложения Элмера.

Визуализированная структура приложения Элмера

Всегда полезно визуализировать структуру приложения, перед тем как погружаться в его детальную разработку. Это означает, что необходимо определить, в решение каких задач будут вовлечены веб-страницы и сценарии, как они должны взаимодействовать друг с другом и, что, возможно, наиболее важно, как будут сохраняться данные в базе MySQL.





Джо: Я не понимаю, какое это имеет значение. Нам будут необходимы и таблица, и сценарий, прежде чем приложение сможет работать.

Фрэнк: Это так, но мне кажется, нам следует вначале написать сценарий, чтобы мы смогли протестировать PHP-код перед соединением с базой данных.

Джил: Но PHP-код полностью зависит от базы данных. Нам будет трудно тестировать сценарий, пока у нас нет базы данных, с которой он должен соединяться.

Фрэнк: А разве мы не можем создать сценарий пока без кода, обеспечивающего соединение с базой данных? Тогда мы могли бы делать все, кроме фактического обмена информацией с базой данных. Это могло бы быть вполне полезным, так ведь?

Джо: Необязательно. Не забывай: единственной задачей сценария является чтение данных, введенных в HTML-форму, и запись их в базу. Или, если речь идет о сценарии, отправляющем электронные письма людям, включенным в список адресов электронной почты, то его задача заключается в извлечении данных из базы и создании электронных писем для каждого пользователя из этого списка. В любом случае наличие базы данных — критическое условие.

Джил: Это так, но мы еще даже и не думали об HTML-форме. Как эта форма будет согласовываться со всеми остальными компонентами? Мне кажется, мы должны создать базу данных, прежде чем сможем даже подумать о написании сценариев.

Фрэнк: Вот именно! Сначала мы создадим HTML-форму, затем определим, какие данные пойдут в базу, и когда все будет готово, мы свяжем все вместе с помощью сценария.

Джо: Я не уверен, что действительно стоит так делать. Как мы можем создавать HTML-форму, если мы не уверены на 100 %, какие именно данные хотим получить от посетителя сайта?

Джил: Джо прав. HTML-форма опять возвращает нас к тому, что вначале необходимо определить, какие именно данные нужны нашему приложению. Все зависит от данных, поэтому нам, видимо, следует вначале создать базу данных и таблицы, затем HTML-форму и потом сценарий, который обрабатывает данные, введенные в форму.

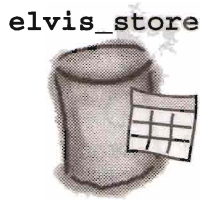
Фрэнк: Вы меня уговорили. Давайте делать так!

Джо: Я также думаю, что нам, возможно, необходимо поразмыслить еще и том, как согласовать отдельные этапы этого приложения.

Перечислите отдельные этапы, которые, на ваш взгляд, перейдут из схемы в окончательный вариант сайта MakeMeElvis.com.

ПЛАН — ПРЕЖДЕ ВСЕГО

Нам действительно необходим план действий по согласованию отдельных частей приложения. Разбивая приложение на отдельные этапы, мы сможем сосредоточиться на решении одной задачи и не запутаться в деталях.



1 Создание базы данных и таблицы для листа рассылки.

В этой таблице будут храниться имя, фамилия и адрес электронной почты каждого, кто включен в лист рассылки Элмера.

2 Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.

Здесь мы создаем форму и сценарий, которые дадут покупателю возможность легко ввести свое имя, фамилию и адрес электронной почты и затем добавят эти данные в лист рассылки.

3 Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.

И наконец, мы создаем веб-форму, которая позволит Элмеру составить текст электронного письма, и что более важно, сценарий, который, используя текст, написанный Элмером, отправит электронное письмо всем покупателям согласно списку, сохраненному в таблице листа рассылки.

Все начинается с таблицы

Фактически все начинается с базы данных, которая является контейнером для хранения данных. Помните, в предыдущей главе говорилось о том, что база данных внутренне разделена на другие контейнеры, называемые **таблицами**?

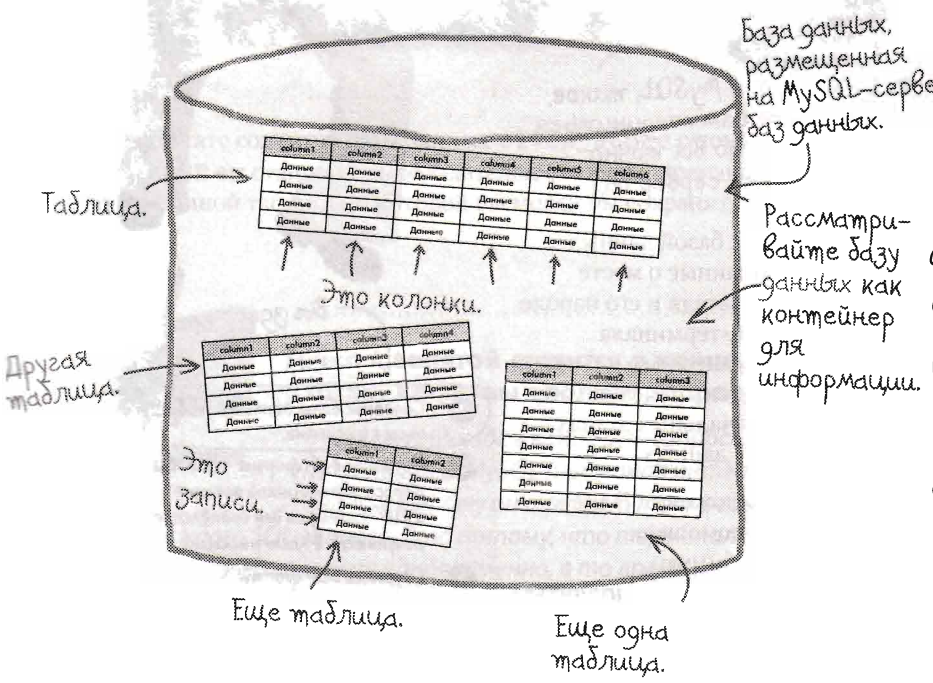
Каждая строка таблицы содержит информацию об одном конкретном объекте, а каждый столбец — конкретную характеристику этого объекта (имя, фамилия, адрес электронной почты...). Строки такой таблицы называются **записями**, столбцы — **колонками**.

Пример записи: «Венди, Верлиц, wwer@starbuzzcoffee.com».

База данных — это контейнер для хранения данных в структурированном виде.

Чаще таблицы в базе данных связаны между собой, хотя в ряде случаев эта связь и не прямая. Для веб-приложений является обычной практикой, когда таблицы связываются между собой через данные, содержащиеся в них. Но сами таблицы все равно состоят из колонок и записей.

Данные хранятся в таблице в колонках и записях.



Не бывает глупых вопросов

В: Где фактически сохраняются данные базы? Могу я увидеть файл?

О: Обычно данные базы сохраняются в файле на жестком диске. И хотя вы действительно можете посмотреть на эти файлы, многого вы в них не увидите. Файлы баз данных являются бинарными и не могут быть открыты просто с целью просмотра содержащихся в них данных. Вот почему необходим SQL — чтобы дать нам возможность просматривать базу и взаимодействовать с находящимися в ней данными.

Еще один контакт с MySQL-сервером баз данных

Структура приложения Элмера требует создания базы данных и таблицы. Большая часть ежедневной работы, связанной с базой данных, включает обмен информацией с таблицами. Но вы не можете просто перепрыгнуть через одну ступеньку и начать создавать таблицы, не создав предварительно базу данных, которой они должны принадлежать.

SQL-запрос `CREATE DATABASE` используется для того, чтобы создать базу данных. После того как это сделано, вы можете идти дальше, создавая таблицы с помощью SQL-запроса `CREATE TABLE`. **Но прежде чем вы сможете использовать любой из перечисленных запросов, вы должны соединиться с вашим MySQL-сервером баз данных**. Вы читали в предыдущей главе, как это делается и что для этого вам необходимо иметь некоторые данные.

Инструментальное программное обеспечение MySQL, такое, например, как MySQL-терминал, позволяет вам соединиться с MySQL-сервером баз данных при условии, что вы точно знаете, где в действительности расположен сервер баз данных, имя пользователя и его пароль.

Точно так же, как в случае открытия соединения с базой данных из PHP-сценария и обмена информацией с ней, данные о месте расположения сервера баз данных, имени пользователя и его пароле являются ключевыми при использовании MySQL-терминала или веб-приложения phpMyAdmin. И использование этого инструментального программного обеспечения очень полезно для создания базы данных и ее таблиц на первоначальных этапах разработки приложения, которое использует базу данных.

Так как создавать базу данных и таблицу для приложения Элмера нужно только один раз, есть смысл сделать это с помощью SQL-запросов вручную. Итак, запустите вашу любимую инструментальную программу MySQL и приготовьтесь выполнить первый этап создания приложения — сформировать базу данных и таблицу для листа рассылки.

1. **Создание базы данных и таблицы для листа рассылки.**
2. **Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.**
3. **Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.**

Создайте базу данных для электронных писем Элмера

Для создания новой таблицы для листа рассылки Элмера первое, что нам понадобится, — это создать базу данных `elvis_store`, которая и будет содержать эту таблицу с именем `email_list`. Мы будем использовать SQL-запросы для создания обоих компонентов. SQL-запрос, используемый для создания баз данных, — это `CREATE DATABASE` («создать базу данных»). Вы уже кратко рассматривали его использование в предыдущей главе. Давайте рассмотрим подробнее, как он действует.

```
CREATE DATABASE имя_базы_данных
```

Имя новой базы данных, которую необходимо создать.

CREATE DATABASE («создать базу данных») — это SQL-запрос, используемый для создания новой базы данных.

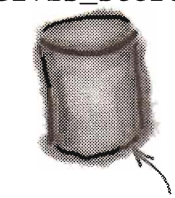
Вам необходимо указать имя новой базы данных после ключевых слов `CREATE DATABASE`. Вот так выглядит SQL-запрос на создание базы данных Элмера:

```
CREATE DATABASE elvis_store
```

Когда вы вводите SQL-запросы в окне терминала, вы всегда добавляете точку с запятой в конце запроса... Но никогда не делаете этого, когда запрос осуществляется через функцию PHP mysqli_query().

После того как вы передадите этот запрос серверу баз данных на выполнение, будет создана база данных.

```
Файл Правка Окно Помощь НебудьЖестоким
mysql> CREATE DATABASE elvis_store;
Запрос выполнен успешно, 1 ряд задействован (0.01 сек)
```



База данных создана, но она не сможет содержать данных без таблиц.

В результате создания базы данных `elvis_store` с помощью запроса `CREATE DATABASE` вы становитесь обладателем сияющей новизной базы данных, но без единой таблицы, в которой вы могли бы сохранять ваши данные...

Запомни!

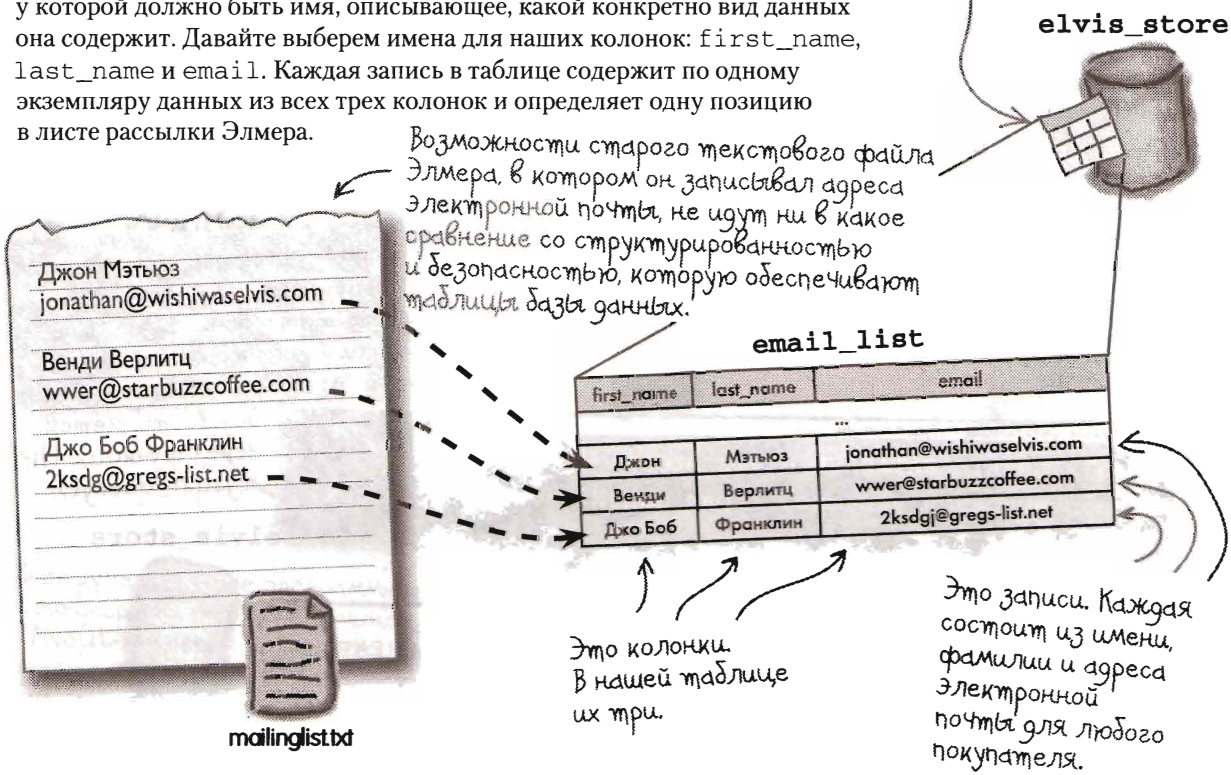
Точка с запятой ставится в конце SQL-запроса только тогда, когда вы вводите запрос в окне терминала.

В вашем PHP-коде в конце SQL-запросов точка с запятой не ставится. MySQL-терминал интерпретирует код иначе, поэтому он требует точки с запятой в конце каждого SQL-запроса. Это потому, что терминал может обрабатывать несколько запросов одновременно, в то время как в PHP вы должны передавать на выполнение по одному запросу за один раз.

Создайте таблицу в базе данных

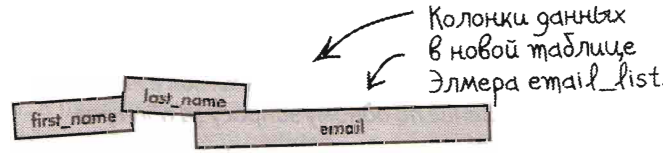
Вы должны знать, какие данные вы собираетесь сохранять в таблице, перед тем как ее создавать. Элмер хочет использовать имя и фамилию покупателя в своем листе рассылки, чтобы электронные письма, которые он отправляет, имели больший персональный оттенок. Добавьте сюда адреса электронной почты, и тогда станет ясно, что таблица Элмера `email_list` должна содержать три вида данных на каждую запись.

Каждый вид данных в таблице помещается в отдельную колонку, у которой должно быть имя, описывающее, какой конкретно вид данных она содержит. Давайте выберем имена для наших колонок: `first_name`, `last_name` и `email`. Каждая запись в таблице содержит по одному экземпляру данных из всех трех колонок и определяет одну позицию в листе рассылки Элмера.



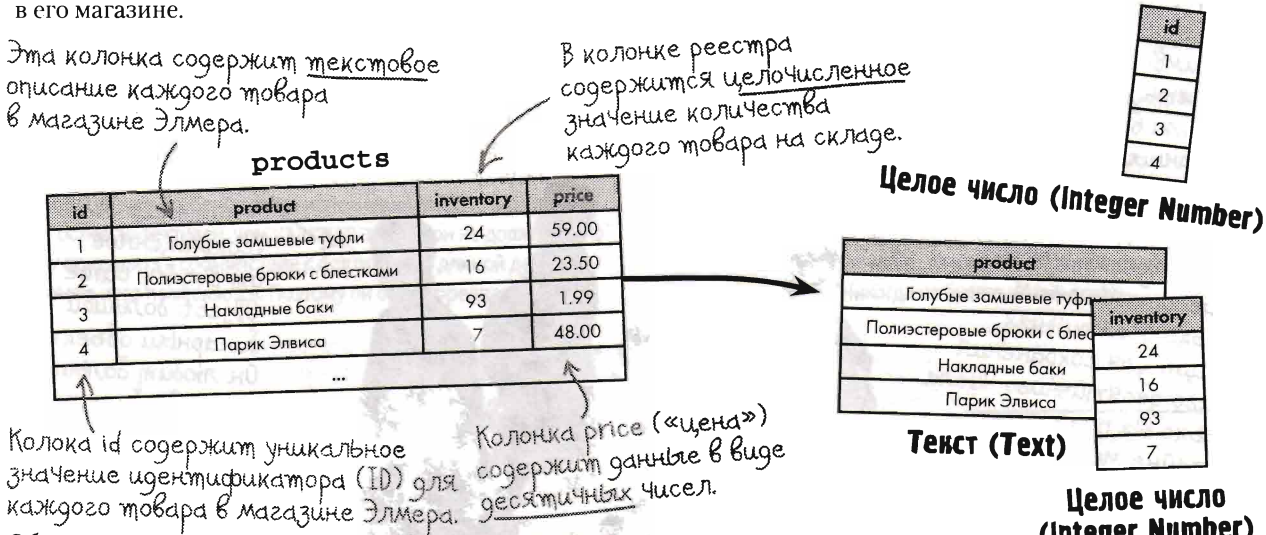
Записи в таблице располагаются горизонтально, а колонки — вертикально.

Итак, вы знаете, что имя, фамилия и адрес электронной почты покупателей должны быть созданы как колонки таблицы `email_list`. Проблема в том, что MySQL-таблицы структурированы в очень высокой степени и указания одного только имени колонки данных недостаточно. Вы должны сообщить базе данных немного больше о том, **какие** данные вы собираетесь хранить в данной колонке.



Нам необходимо указать типы наших данных

Когда вы создаете таблицу, вы должны сообщить серверу MySQL, данные какого типа будут храниться в каждой из ее колонок. Типы данных должны быть указаны для всех колонок таблицы, и в каждую колонку могут быть добавлены данные только того типа, который для нее определен. Это значит, что какие-то колонки могут хранить текстовые данные, какие-то — числа, какие-то — даты или время и т. д. В MySQL определено множество типов данных, и вы должны знать, какие из них соответствуют типам ваших данных. Предположим, у Элмера есть таблица с именем `products`, которая отслеживает движение товаров в его магазине.



Обратите внимание на то, что `product` — единственная колонка с текстовым типом данных в таблице `products`. Имеются также следующие типы: десятичное число для колонки `price` и целое число для колонок `id` и `inventory`. В MySQL для всех этих типов данных установлены свои наименования, так же как и для некоторых других типов.

Очень важно при создании колонок таблицы использовать оптимальный тип данных, чтобы добиваться наибольшей точности и эффективности. Например, **текстовые данные занимают больше места в памяти, чем целочисленные**, поэтому, если колонка будет содержать только целочисленные значения, хороший стиль программирования требует выбрать для нее тип данных «целое число». Кроме этого, после того как вы определили конкретный тип данных для какой-либо колонки, сервер баз данных MySQL не позволит вам ввести в нее данные любого другого типа. Поэтому, если для вашей колонки определен тип данных, скажем, «дата», вы получите сообщение об ошибке, если попытаетесь ввести данные любого другого типа, кроме даты.

Создавая таблицу, вы должны знать, какие типы данных будут содержаться в каждой из ее колонок.

Как вы думаете, использование разных типов данных лучше, чем использование просто текста для сохранения всех данных?

Познакомьтесь с некоторыми типами данных MySQL

Имеется несколько наиболее полезных типов данных MySQL. Не забывайте, что вы можете использовать любые из них для описания типа данных, которые могут быть сохранены в конкретной колонке таблицы. Это их забота — выполнять всю черную работу по сохранению ваших данных без осложнений.

CHAR, или CHARACTER. Ведет себя достаточно жестко и предпочитает данные в виде последовательности символов определенной длины. Может быть исключительно эффективной, если вы используете в качестве данных фрагменты текста фиксированной длины.

INT, или INTEGER (целое число), считает, что числа должны быть только целыми, но совершенно не возражает против отрицательных значений. Он может также сохранять небольшие целые числа. В этом случае его зовут TINYINT (маленькое целое).

DEC, сокращение от DECIMAL (десятичный). Он предоставит вам столько десятичных позиций для сохранения вашего десятичного числа, сколько вы попросите, по крайне мере, пока не переполнится.

Зовите его BLOB (сокращенное от Bynary Large OBject: большой бинарный объект). Он любит большие массивы бинарных данных.

DATETIME (дата, время) или TIMESTAMP (время (вместе с датой), когда произошло событие). Следит за датой и временем.

Хорошая подруга BLOB, зовут ее TEXT (текст). Она — большой мастер сохранять огромные объемы текста. Значительно больше, чем CHAR или VARCHAR.

Это VARCHAR, сокращение от VARiable CHARacter (последовательность символов переменной длины). Он сохраняет текстовые данные. Он достаточно гибок и может приспосабливаться к длине ваших данных, сохраняя только то, что вам необходимо, и не заполняя оставшееся пространство дополнительными пробелами.

DATE (дата) следит за датой. Правда, она совершенно не думает о времени. У нее есть также брат-близнец, TIME (время), который понятия не имеет, что такое дата.

В зависимости от версии вашего сервера баз данных MySQL длина может составлять 255 символов для версий ниже 5.0.3 и 65 535 символов для версий 5.0.3 и выше.

Не бывает глупых вопросов

В: Зачем мне вообще использовать тип CHAR, если VARCHAR делает то же самое, но со значительно большей гибкостью?

О: Ответ заключается в точности и эффективности. С позиции оптимальности вы должны разрабатывать структуру своих таблиц, чтобы она в наиболее точной степени моделировала структуру ваших данных. Если вы знаете без тени сомнения, что в колонке, содержащей краткое наименование штата (в США), будет всегда помещаться его двухбуквенная аббревиатура, имеет смысл предусмотреть в этой колонке только два символа, выбрав тип данных `CHAR(2)`. Тем не менее, если колонка для сохранения пароля может содержать любое количество символов, но не более 10, тогда выбор `VARCHAR(10)` более оправдан. Это объясняется особенностями исполнения MySQL. Тип данных `CHAR` более эффективен, чем `VARCHAR`, так как он не должен свою длину приводить в соответствие с фактической длиной данных, которые в него помещаются. Поэтому он более предпочтителен в случаях, когда вы знаете, что данные, помещаемые в определенную колонку, имеют фиксированную длину.

В: Зачем мне нужны числовые типы данных INT и DEC?

О: Все объясняется способом сохранения данных и эффективностью. Выбирая соответствующий тип данных для каждой колонки в своей таблице, вы уменьшаете размеры таблицы и увеличиваете скорость обработки данных. Сохранение числа в виде численного типа данных (INT, DEC), вместо сохранения его в виде текста, обычно значительно эффективнее.

В: Это все? Других типов данных больше нет?

О: Есть, но это наиболее часто применяемые типы. Лучше мы начнем пока с них, чем углубляться в изучение тех типов данных, которые вы, возможно, никогда не будете использовать.

Где мое место

Приведите каждый тип данных MySQL в соответствие с описанием данных, которые вы могли бы сохранить в таблице.

Тип данных	Описание
`INT`	Ваши имя и фамилия
`CHAR(1)`	Наименование штата США в виде двухбуквенной аббревиатуры
`DATE`	Стоимость парика Элвиса: 48.99
`TIME`	Сколько денег было выручено за альбом-бестселлер Элвиса
`VARCHAR(2)`	Дата похищения космическими пришельцами: 19.02.2004
`DEC(4,2)`	Количество баков Элвиса на складе
`VARCHAR(60)`	Вы видели собаку Оуэна? Y или N
`CHAR(2)`	Ваш адрес электронной почты
`DATETIME`	Когда вы обедаете
`DEC(10,2)`	Сколько космических пришельцев вы видели во время похищения
	Когда родился Элвис

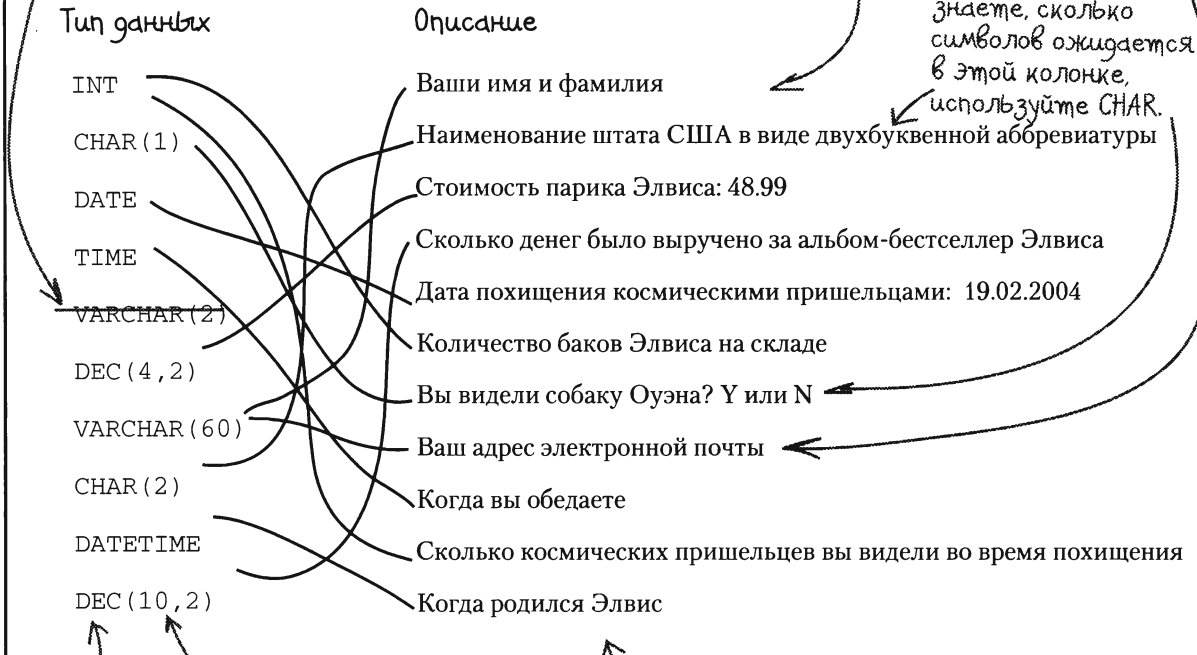
ГДЕ МОЕ МЕСТО

Приведите каждый тип данных MySQL в соответствие с описанием данных, которые вы могли бы сохранить в таблице.

Не нужен. Хотя он и может быть использован для сохранения двухбуквенной аббревиатуры наименования штата США, CHAR(2) является более удачным выбором, так как он эффективнее.

Если длина текста может меняться, VARCHAR — это оптимальный выбор. Сделайте его достаточно длинным, чтобы позволить сохранение данных любой длины.

Если вы точно знаете, сколько символов ожидается в этой колонке, используйте CHAR.



DEC обычно используется для сохранения цен или других десятичных чисел.

Эти два числа показывают, сколько цифр колонка предполагает увидеть перед десятичной запятой и сколько — после нее.

Вы, возможно, выбрали здесь DATE, но настоящие Элвисоманы знают точную дату и время.

В MySQL существует другой (потенциально лучший) метод представления данных «Да»/«Нет», чем использование CHAR(1), но и этот метод прост и достаточно эффективен.

Создайте вашу таблицу с помощью запроса

Да, мы все еще здесь.. Но мы почти готовы двигаться дальше.

1. Создание базы данных и таблицы для листа рассылки.
2. Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.
3. Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.

У нас есть все, что необходимо для создания нашей таблицы, даже хорошее имя (`email_list`). Есть у нас и имена для колонок: `first_name`, `last_name` и `email`. Чего у нас нет, так это типов данных для каждой колонки и SQL-запроса, который собрал бы все это вместе и создал таблицу. SQL-запрос для создания таблицы начинается с ключевых слов `CREATE TABLE` («создать таблицу»).

После ключевых слов `CREATE TABLE` следует имя таблицы. Две круглые скобки ограничивают разделенный запятыми список имен всех колонок. За каждым именем колонки следует тип данных, которые она будет содержать. Вот так выглядит этот запрос:

```
CREATE TABLE имя_таблицы
(
имя_колонки1 тип_данных1,
имя_колонки2 тип_данных2,
...
)
```

Имя таблицы.

Имя колонки.

Тип данных колонки.

Еще колонки, если необходимо.

Вы не обязаны включать в имена колонок символ подчеркивания для разделения слов, но хороший стиль программирования предполагает это.

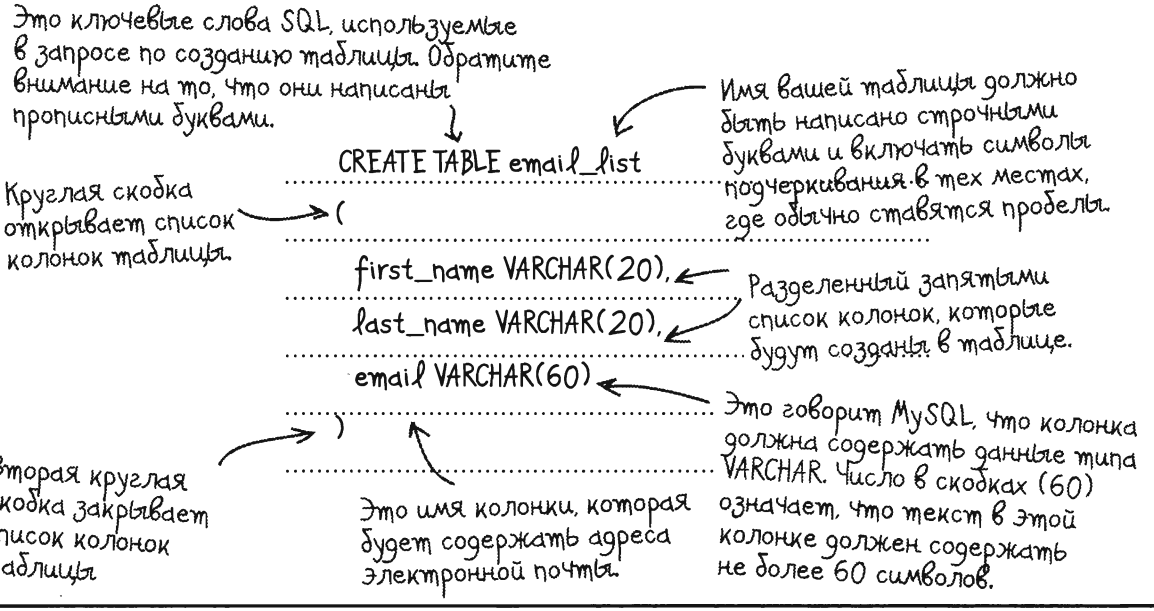
SQL-запрос CREATE TABLE используется для создания новой таблицы в базе данных.

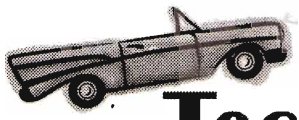
Время поработать

Напишите SQL-запрос, позволяющий создать таблицу Элмера `email_list` с тремя колонками данных: `first_name`, `last_name` и `email`.

Решение задачи

Напишите SQL-запрос, позволяющий создать таблицу Элмера `email_list` с тремя колонками данных: `first_name`, `last_name` и `email`.





Тест-драйв

Создайте базу данных и таблицу для Элмера.

Выполните запросы `CREATE DATABASE` и `CREATE TABLE`, чтобы создать базу данных `elvis_store` и в ней таблицу `email_list`, используя какое-нибудь из инструментальных MySQL-приложений.

```
CREATE DATABASE elvis_store

CREATE TABLE email_list(first_name VARCHAR(20), last_name VARCHAR(20), email VARCHAR(60))
```

Оба запроса будут выполнены без проблем? Если нет, напишите, что, по вашему мнению, может пойти не так.

Подождите, что-то здесь не ладится. Я ввел код, чтобы создать таблицу, точно такой же, как тот, что мы только что составили... И вот я получаю какое-то странное сообщение об ошибке.

Элмер раздражен, потому что его запрос CREATE TABLE безупречен, а MySQL, тем не менее, сообщает о какой-то ошибке.

С самим запросом CREATE TABLE все в порядке, но MySQL-терминал не знает, в какой базе нужно создавать таблицу... Это никуда не годится!

```
Файл Правка Окно Помощь Ой
mysql> CREATE TABLE email_list
(
  first_name VARCHAR(20),
  last_name VARCHAR(20),
  email VARCHAR(60)
);
ОШИБКА 1046 (3D000): База данных не выбрана
```

По каким-то причинам запрос CREATE TABLE не сработал при попытке выполнения его из MySQL-терминала.

Таблица ~~Телега~~ базы данных впереди ~~лошади~~

Элмер столкнулся с серьезной проблемой, связанной с тем, что MySQL-терминал не определяет автоматически, какую базу данных вы имеете в виду, когда вводите запрос. Конечно, он знает, что вы только что создали базу данных `elvis_store`, но на этом сервере может быть множество других. Он не может предполагать, что вы хотите иметь дело с той базой данных, которую вы только что создали.

К счастью, есть очень простое решение, с помощью которого вы можете сообщить MySQL-терминалу, к какой базе данных относятся все ваши последующие запросы...

Не бывает глупых вопросов

В: Что это за странное приглашение ->, которое я иногда вижу в моем MySQL-терминале?

О: Приглашение -> показывает вам, что вы вводите один большой запрос, разбивая его на несколько строк. MySQL таким образом говорит вам, что он знает, что вы все еще продолжаете вводить запрос, даже если вы нажали клавишу «Ввод» («Перевод строки»), чтобы разбить длинную строку запроса на несколько коротких. После того как вы поставите точку с запятой в конце запроса (неважно, сколько строк он занимает в окне терминала), MySQL выполнит его.

Выполните запрос USE перед использованием базы

Для того чтобы запрос `CREATE TABLE` работал, Элмеру необходимо выбрать базу данных в MySQL-терминале, и он будет знать, какой базе данных принадлежит новая таблица. Запрос `USE` («использовать») определяет базу данных как базу данных по умолчанию в терминале, и все последующие SQL-запросы будут относиться к ней.
Вот как это работает:

Запрос USE определяет базу данных по умолчанию для всех последующих SQL-запросов.

Запрос USE говорит MySQL, какую базу данных вы намерены использовать.

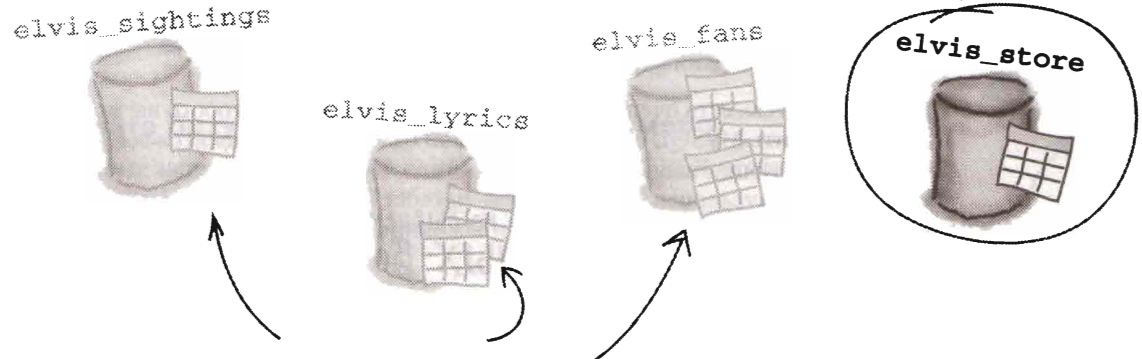
```
USE имя_базы_данных
```

Элмеру необходимо указать имя своей базы данных в запросе `USE`, чтобы получить доступ к таблице именно в этой базе данных.

Имя базы данных, которую вы хотели бы использовать.

```
USE elvis_store
```

Запрос USE выбирает базу данных, с которой вы хотите работать.



Как только вы выбрали базу данных для использования с помощью запроса USE, остальные базы данных на этом сервере баз данных будут игнорироваться, пока вы не выберете таким же способом другую.

Тест-драйв

Вначале выберите базу данных Элмера с помощью запроса USE, а затем создайте таблицу.

Выполните запрос `USE`, чтобы выбрать базу данных `elvis_store` в каком-нибудь из инструментальных MySQL-приложений, а затем выполните запрос `CREATE TABLE`, чтобы создать таблицу `email_list` в этой базе данных.

```
USE elvis_store

CREATE TABLE email_list(first_name VARCHAR(20), last_name VARCHAR(20), email VARCHAR(60))
```

При использовании графических инструментальных программ MySQL, таких как phpMyAdmin, нет необходимости выполнять запрос USE. Эти программы требуют выбрать нужную вам базу через органы управления графического интерфейса перед вводом SQL-запроса.

```
Файл Правка Окн Помощь ЛизаМария
mysql> USE elvis_store;
База данных изменена
mysql> CREATE TABLE email_list
(
    first_name VARCHAR(20),
    last_name VARCHAR(20),
    email VARCHAR(60)
);
Ваш SQL-запрос выполнен успешно (Длительность выполнения запроса
04481 сек)
```

Код создания таблицы такой же, как и прежде, но только требуется выбрать базу данных, прежде чем он сможет работать.

После выбора базы данных с помощью запроса USE создание таблицы протекает без каких-либо проблем.

В SQL отсутствует команда «Отменить ввод» в чистом виде, но возможность исправления ошибок все-таки существует.

Тем не менее, вы должны определить, что за ошибка была сделана, прежде чем приступать к ее исправлению. Предположим, таблица `email_list` выглядит так:

email_list

first_naem	last_name	email

Проверьте, что не так в этой таблице.
Есть какие-нибудь идеи, как можно исправить ошибку?

Запрос DESCRIBE показывает структуру таблицы

Для того чтобы исправить ошибку в таблице, ее нужно прежде найти. Даже если вы не предполагаете наличие ошибки, никогда не вредно лишний раз проверить свою работу. Запрос SQL `DESCRIBE` («составить описание») анализирует структуру таблицы и выводит список имен колонок, типов данных и другую информацию.

```
DESCRIBE имя_таблицы
```

Замена шаблона имя_таблицы на действительное имя таблицы дает нам следующий SQL-запрос:

```
DESCRIBE email_list
```

Имя таблицы, описание которой мы хотели бы видеть.

```
Файл Правка Окно Помощь Грейсленд
mysql> DESCRIBE email_list;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| first_naem | varchar(30) | YES  |     | NULL    |       |
| last_name  | varchar(30) | YES  |     | NULL    |       |
| email      | varchar(60) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
3 строки в списке (0.02 сек)
```

В колонке Field вы найдете имя каждой колонки вашей таблицы.

В колонке Type вы видите типы данных, которые вы установили для каждой колонки вашей таблицы.

MySQL не чувствителен к регистру клавиатуры, когда речь идет о ключевых словах, таких как типы данных, поэтому иногда вы будете видеть эти слова, написанные строчными буквами.

Не бывает глупых вопросов

В: Что это за колонки: Null («ноль», «отсутствие данных»), Key («ключ»), Default («значение по умолчанию») и Extra («дополнение»)?

О: MySQL позволяет вам установить различные опции (в SQL — ограничивающие условия) для каждой колонки вашей таблицы. Эти опции контролируют такие условия, как допускается ли отсутствие данных в колонке или установлено ли для нее значение по умолчанию. Мы рассмотрим эти опции немного позднее, когда их использование станет критичным для приложения.

В: Если в моей таблице имеются данные, можно ли с помощью этого запроса увидеть их?

О: Нет. `DESCRIBE` показывает вам только структуру таблицы, а не имеющиеся в ней данные. Не волнуйтесь, очень скоро вы увидите данные вашей таблицы… но сначала мы должны узнать, как поместить данные в таблицу.

В: Могу ли я точно так же увидеть структуру таблицы, используя phpMyAdmin?

О: Да. Графические инструментальные программы, такие как phpMyAdmin, позволяют видеть структуру таблиц с помощью запроса `DESCRIBE` или щелкнув на вкладке визуального просмотра таблицы. Какой конкретно инструментарий использовать для анализа ваших таблиц — зависит исключительно от вас.

Я исправила опечатку и попыталась выполнить запрос CREATE TABLE заново. Это не дало никакого результата. Я что, должна вначале удалить таблицу с опечаткой?

В имени первой колонки была случайно допущена опечатка: first_naem... ой!

```
Файл Правка Окно Помощь Опечатка?
mysql> DESCRIBE email_list;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| first_naem | varchar(30) | YES  |     | NULL    |       |
| last_name  | varchar(30) | YES  |     | NULL    |       |
| email      | varchar(60) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
3 строки в списке (0.02 сек)
```

Фактически это вам и необходимо сделать. Вы не можете пересоздать таблицу с помощью запроса CREATE TABLE, если такая таблица уже была создана.

После того как вы создали таблицу, она продолжает существовать и не может быть пересоздана с помощью нового запроса `CREATE TABLE`. Если вы хотите пересоздать таблицу с нуля, то должны сначала удалить существующую и затем начать все сначала. В SQL для удаления таблицы из базы данных используется запрос `DROP TABLE` («удалить таблицу»). В результате выполнения этого запроса таблица удаляется вместе со всеми хранящимися в ней данными. Так как в новой таблице пока никаких данных нет, мы ничего не теряем в результате операции по удалению старой таблицы и созданию новой с откорректированным именем `first_name`.

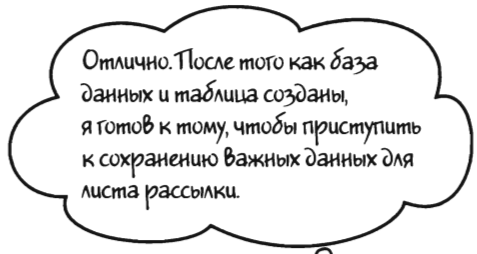
```
DROP TABLE email_list
```

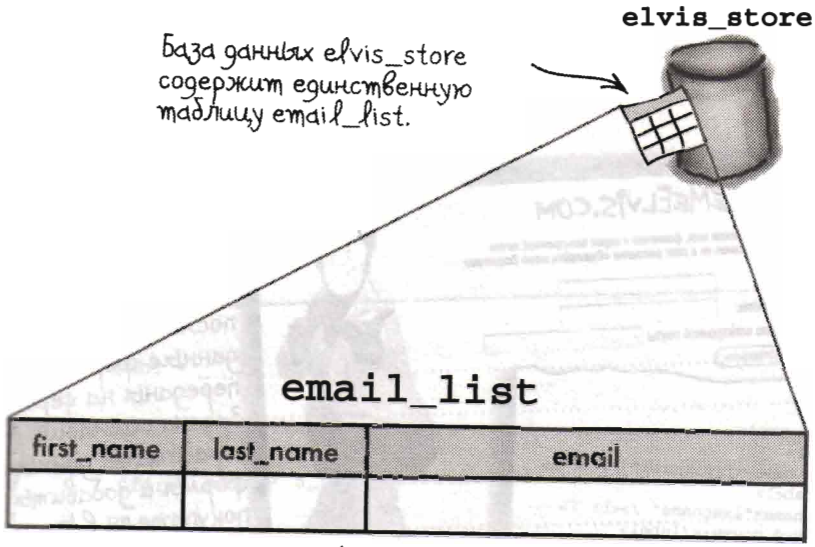
Имя таблицы, которую мы хотели бы удалить из базы данных.

Запрос DROP TABLE удаляет из базы данных таблицу и все данные, хранящиеся в ней.

Элмер готов заносить данные

SQL-запросы CREATE DATABASE, USE и CREATE TABLE были успешно использованы для создания базы данных листа рассылки и таблицы. Элмер мог бы быть более довольным только в том случае, если бы таблица была уже заполнена жаждущими покупателями. Это работа для PHP...





не бывает глупых вопросов

В: Эй, у меня есть книга Head First SQL (отличная, книга, кстати). В ней каждый раз, когда вы показываете код SQL, вы ставите точку с запятой в конце всех SQL-запросов. Почему здесь их нет?

О: Мы рады, что вам понравилась книга «Очертя голову в SQL». Разница заключается в том, что, когда вы имеете дело непосредственно с SQL, вы обязательно должны использовать точку с запятой, чтобы SQL мог понять, где заканчивается запрос, потому что MySQL может обрабатывать несколько запросов одновременно. В PHP при использовании функции `mysqli_query()` вы можете передавать на выполнение по одному SQL-запросу за один раз, поэтому точки с запятой не требуются. Но не забывайте, что вы обязательно должны ставить точку с запятой в конце каждого PHP-выражения.

В: Значит, если в моей таблице есть данные и я удалю ее, все мои данные будут потеряны?

О: Это так. Поэтому будьте внимательны, когда удаляете таблицы.

В: Значит, если мне необходимо удалить таблицу с данными, мне крупно не повезло?

О: Эй, совершенных людей не существует. Все совершают ошибки, и SQL предлагает запрос ALTER («изменить»), с помощью которого мы можем изменить существующую таблицу. Мы поговорим об этом запросе немного позднее.

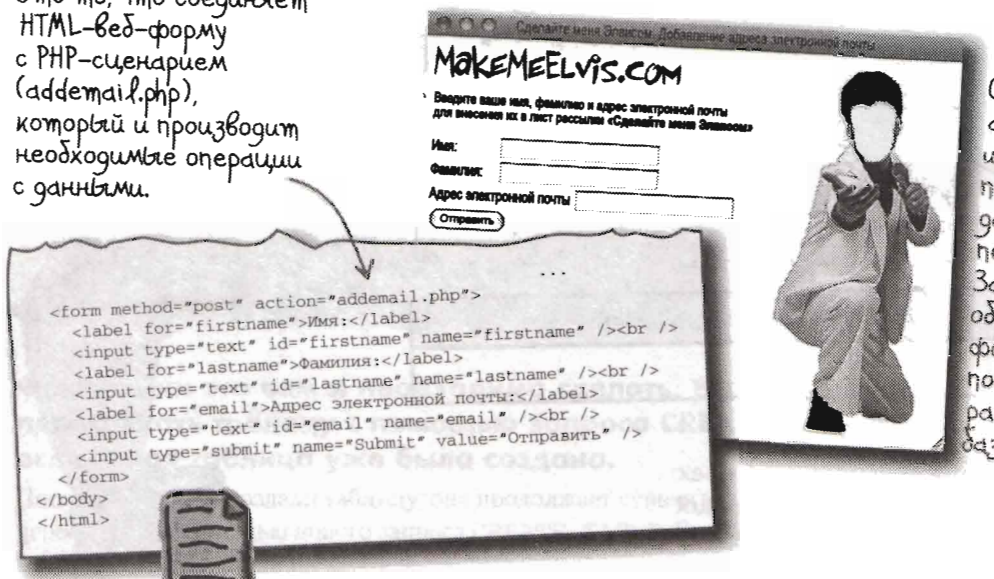
Создайте сценарий addemail.php

Элмеру необходима HTML-форма, которая могла бы принимать от покупателей их имена и адреса электронной почты. Получив такие данные, он мог бы передать их PHP-сценарию и сохранить в таблице `email_list`. В веб-форме (`addemail.html`) необходимо иметь три поля ввода данных и кнопку. Атрибут формы `action` является наиболее важной частью кода, так как его задача — передать данные формы сценарию `addemail.php`, к созданию которого мы приступаем.

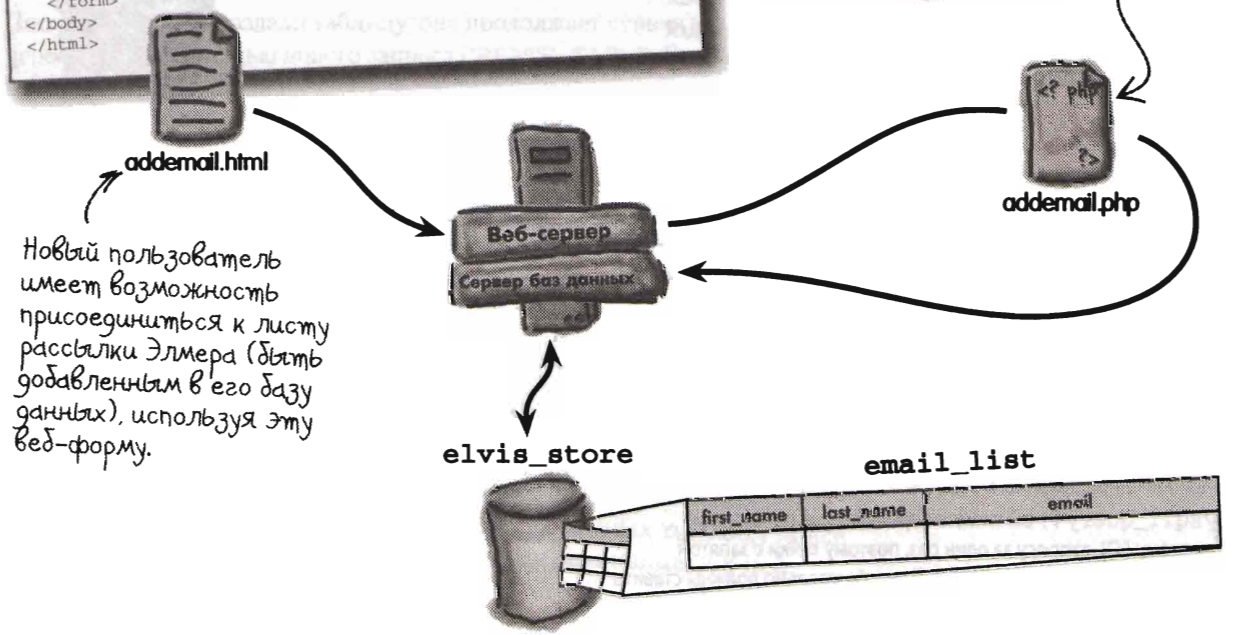
Вы здесь.

1. ~~Создание базы данных и таблицы для листа рассылки.~~
2. Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.
3. Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.

Атрибут action — это то, что соединяет HTML-веб-форму с PHP-сценарием (addemail.php), который и производит необходимые операции с данными.



Сценарий addemail.php начнет интерпретироваться после того, как данные формы будут переданы на сервер. Задача сценария — обработать данные формы и добавить покупателя в лист рассылки (таблицу базы данных).



Новый пользователь имеет возможность присоединиться к листу рассылки Элмера (быть добавленным в его базу данных), используя эту веб-форму.

Сценарий `addemail.php` обрабатывает данные формы «Добавление адреса электронной почты». Этот сценарий должен получить данные от формы, соединиться с базой данных `elvis_store` и добавить эти данные в таблицу `email_list` с помощью запроса `INSERT`. Помогите Элмеру. Напишите сначала пример SQL-запроса для добавления нового пользователя, а затем используйте этот запрос для окончания кода сценария.

Напишите здесь пример запроса, в результате выполнения которого данные будут добавлены в таблицу Элмера.

```
<?php
  $dbc = .....
   .....

  $first_name = $_POST['firstname'];
  .....
  .....

  $query = .....
    .....
  mysqli_query(.....)
   .....

  echo 'Покупатель добавлен.';

  .....
?>
```

addemail.php

Сценарий `addemail.php` обрабатывает данные формы «Добавление адреса электронной почты». Этот сценарий должен получить данные от формы, соединиться с базой данных `elvis_store` и добавить эти данные в таблицу `email_list` с помощью запроса `INSERT`. Помогите Элмеру. Напишите сначала пример SQL-запроса для добавления нового пользователя, а затем используйте этот запрос для окончания кода сценария.

```
INSERT INTO email_list (first_name, last_name, email)
  VALUES ('Джулиан', 'Оумс', 'julian@breakneckpizza.com')
```

Пример запроса INSERT переписан в виде PHP-строки, которая включает в себя данные формы, подлежащие добавлению.

Это значения элементов массива $_POST, в которых заключена переданная информация.

```
<?php
  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    or die('Ошибка соединения с MySQL-сервером.'); 'server.');

  $first_name = $_POST['firstname'];
  $last_name = $_POST['lastname'];
  $email = $_POST['email'];

  $query = "INSERT INTO email_list (first_name, last_name, email) "
    "VALUES ('$first_name', '$last_name', '$email')";

  mysqli_query( $dbc, $query )
    or die('Ошибка выполнения запроса к базе данных.');

  echo 'Customer added.';

  mysqli_close($dbc);
?>
```

addemail.php

Если вы хотите сделать страницу интереснее, можете добавить сюда гиперссылку, указывающую на вашу форму, с помощью HTML-тега <a>.

Тест-драйв

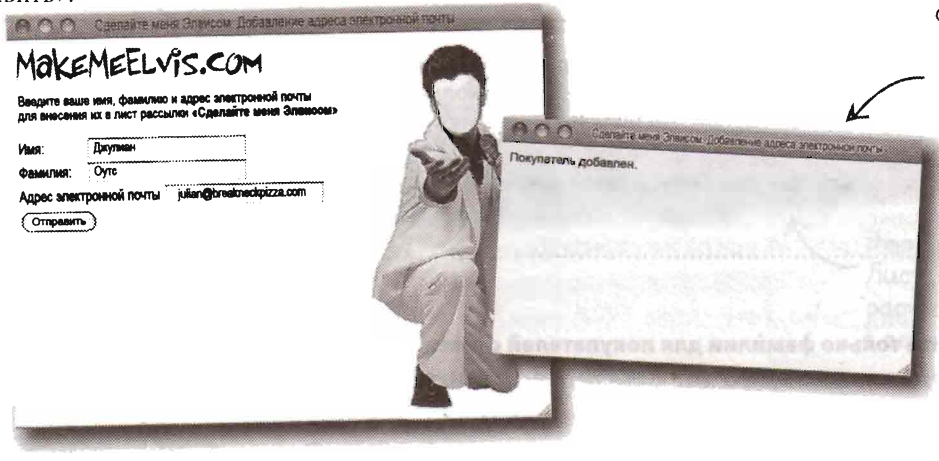
Проверьте форму «Добавление адреса электронной почты».

Загрузите код для веб-страницы «Добавление адреса электронной почты» с сайта «Лаборатория "Очертя голову"» по адресу `www.headfirstlabs.com/books/hfphp`. Этот код находится в каталоге `chapter03` и включает веб-форму Элмера `addemail.html`, каскадные таблицы стилей (`style.css`) и два файла изображений (`elvislogo.gif` и `blankface.jpg`).

Не забудьте изменить переменную, содержащую ссылку на соединение с сервером баз данных, с учетом вашего конкретного окружения.

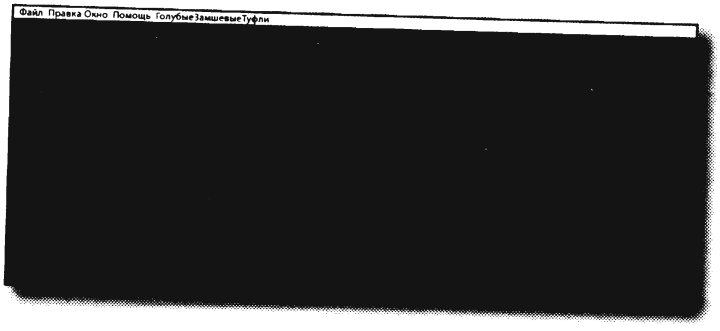
А теперь создайте новый текстовый файл `addemail.php` и введите весь код предыдущей страницы. Это сценарий, который будет обрабатывать данные веб-формы и добавлять новых покупателей в таблицу `email_list`.

Загрузите все файлы на ваш веб-сервер и откройте страницу `addemail.html` в веб-браузере. Введите данные нового покупателя в форму и нажмите кнопку «Отправить».



Добавление нового покупателя в лист рассылки подтверждено сценарием addemail.php.

Проверьте, что покупатель был добавлен в базу данных запросом `SELECT` из инструментальной программы MySQL .



Не бывает глупых вопросов

В: Есть ли в SQL другие символы, имеющие специальное значение, как «звездочка»?

О: Хотя в SQL есть и другие символы, имеющие специальное значение, «звездочка» — единственный, о котором вам необходимо пока знать. Более важным для наших текущих целей является то, что это единственный из специальных символов, используемый в запросах SELECT.

Время поработать

Так как в лист рассылки Элмера стали поступать данные, помогите ему написать несколько SQL-запросов, которые он мог бы использовать для поиска определенных данных покупателей.

Выберите все данные для покупателей с именем Мартин:

.....

Выберите только фамилии для покупателей с именем Буба:

.....

Выберите имена и фамилии покупателей, чей адрес электронной почты ls@objectville.net:

.....

Выберите все колонки для покупателей с именем Амбер и фамилией Маккарти:

.....

Файл Правка Окно Помощь ЗаконыЭлвиса

first_name	last_name	email
Джулиан	Оутс	julian@breakneckpizza.com
Кевин	Джонс	jones@simuduck.com
Аманда	Санчес	sunshine@breakneckpizza.com
Бо	Вэлэйс	bo@b0tt0msup.com
Амбер	Маккарти	amber@breakneckpizza.com
Кормак	Нертс	churst@boards-r-us.com
Джоис	Харпер	joyceharper@breakneckpizza.com
Стивен	Мэйер	meyers@leapenlimos.com
Мартин	Уилсон	martybaby@objectville.net
Уолт	Перала	walt@mightygumball.net
		craftsman@breakneckpizza.com
		joe_m@starbuzzcoffee.com
		bruce@chocoholic-inc.com
		pr@honey-doit.com
		bertieh@objectville.net
		gregeck@breakneckpizza.com
		wilmawu@starbuzzcoffee.com
		samjaffe@starbuzzcoffee.com
Луи	Шэффер	ls@objectville.net
Буба	Шекспир	bshakes@mightygumball.net
Джон	До	johndoe@tikibeanlounge.com

Отлично. Теперь эти покупатели могут подписываться на мой лист рассылки через веб-страницу. Этот лист очень хорошо составляется сам.

Это не конец таблицы. Лист рассылки Элмера растет теперь с огромной скоростью.

Но электронное письмо не будет составляться самостоятельно.

Элмеру все еще не хватает другой части веб-приложения — той, которая предоставит ему возможность составить содержание электронного письма и затем отправит его всем покупателям, перечисленным в его листе рассылки. Чтобы решить эту задачу, ему необходима новая HTML-форма и PHP-сценарий для обработки ее данных.

1. ~~Создание базы данных и таблицы для листа рассылки.~~
2. ~~Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.~~
3. Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.

Второй этап выполнен!

Решение задачи

Так как в лист рассылки Элмера стали поступать данные, помогите ему написать несколько SQL-запросов, которые он мог бы использовать для поиска определенных данных покупателей.

Выберите все данные для покупателей с именем Мартин:

```
SELECT * FROM email_list WHERE first_name = 'Мартин'
```

«Звездочка» говорит о том, что нужно выбрать значения из всех колонок таблицы.

Ключевое слово WHERE устанавливает ограничивающее условие, согласно которому выводятся только те записи, для которых имя покупателя — Мартин.

Выберите только фамилии для покупателей с именем Буба:

```
SELECT last_name FROM email_list WHERE first_name = 'Буба'
```

В результате выполнения запроса выводятся только значения колонок с фамилией покупателя.

Выберите имена и фамилии покупателей, чей адрес электронной почты ls@objectville.net:

```
SELECT first_name, last_name FROM email_list WHERE email = 'ls@objectville.net'
```

Вы определили вывод значений двух колонок, при этом имена колонок в запросе разделены запятыми.

Выберите все колонки для покупателей с именем Амбер и фамилией Маккарти:

```
SELECT * FROM email_list WHERE first_name = 'Амбер' AND last_name = 'Маккарти'
```

Ограничивающее условие может состоять из нескольких частей. В данном случае необходимо вывести все записи для пользователей с именем Амбер и фамилией Маккарти, поэтому дополнительное условие, следующее за условием, определенным ключевым словом WHERE, присоединяется с помощью ключевого слова AND («и»).

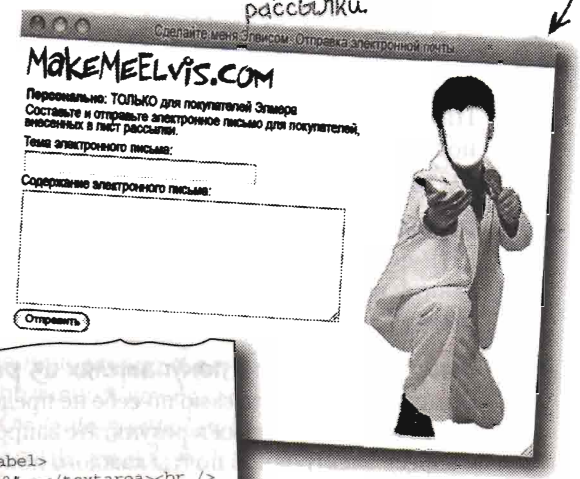
Другая сторона приложения Элмера

Задача отправки электронных писем людям, перечисленным в листе рассылки Элмера, напоминает в каком-то смысле задачу добавления сведений об этих людях в лист рассылки, потому что обе они решаются с использованием HTML-форм и PHP-сценариев. Главное отличие заключается в том, что отправка электронных писем требует данных всей таблицы `email_list`, в то время как сценарий `addemail.php` имеет дело с данными только одной ее записи.

Форма «Отправка электронной почты» дает возможность Элмеру ввести тему и содержание электронного письма и затем отправить его всем покупателям, перечисленным в листе рассылки.

1. ~~Создание базы данных и таблицы для листа рассылки.~~
2. ~~Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.~~
3. Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.

Вот так, мы уже на последнем этапе.




```
...
<form method="post" action="sendemail.php">
  <label for="subject">Тема электронного письма:</label>
  <input type="text" id="subject" name="subject" /><br />
  <label for="elvismail">Содержание электронного письма:</label>
  <textarea id="elvismail" name="elvismail" rows="8" cols="60" ></textarea><br />
  <input type="submit" name="Submit" value="Отправить" />
</form>
</body>
</html>
```

sendemail.html

Атрибут формы action включает сценарий sendemail.php.

Сценарий sendemail.php извлекает данные о покупателях из таблицы базы данных и отправляет электронное письмо каждому из них.

sendemail.php

elvis_store

email_list

first_name	last_name	email
Джулиан	Оутс	julian@breackneckpizza.com
Кевин	Джонс	jones@simuduck.com
Аманда	Санчес	sunshine@breakneckpizza.com
...		

Колесики и винтики сценария «Отправка электронной почты» (sendemail.php)

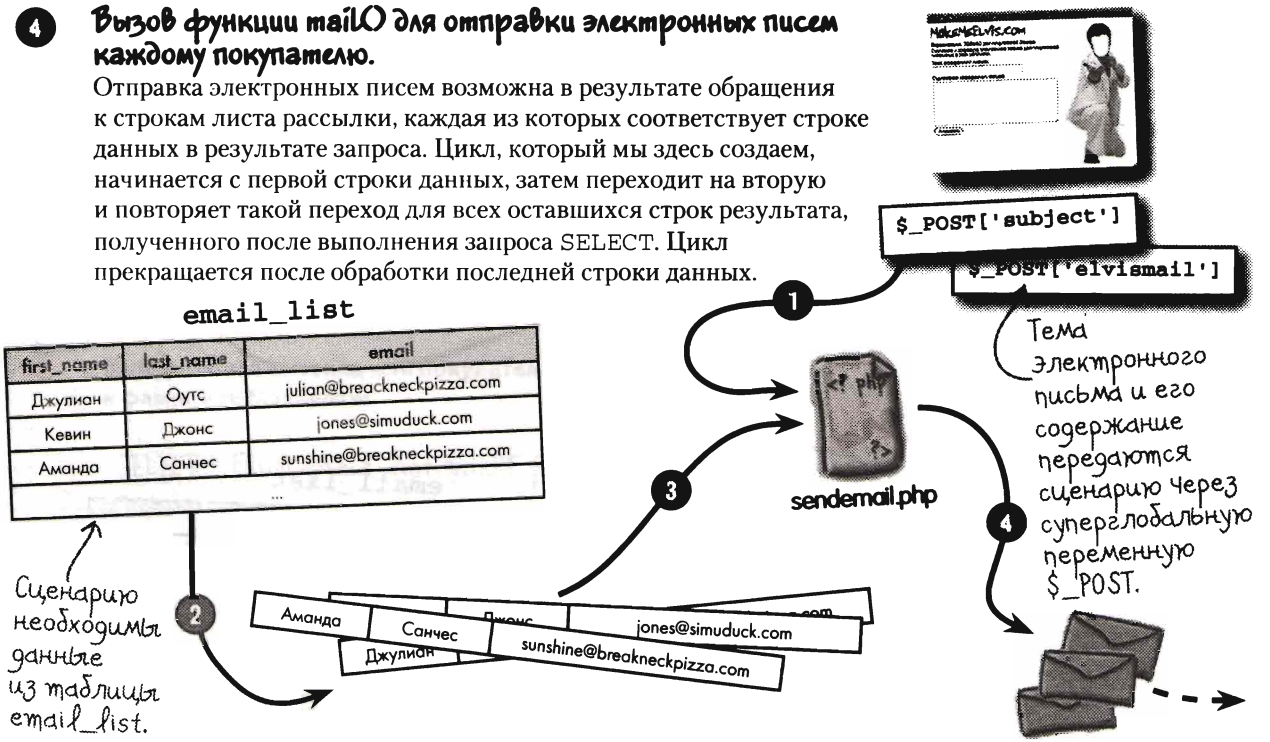
Сценарий `sendemail.php` должен объединить данные из двух разных источников информации для того, чтобы составить и отправить электронные письма. С одной стороны, сценарий должен взять имена и адреса электронной почты получателей из таблицы `email_list` базы данных `elvis_store`. С другой стороны, он также должен прочитать тему и содержание электронного письма, введенные Элмером в форму «Отправка электронной почты» (`sendemail.html`). Давайте разобьем все это на этапы.

1. **Считывание темы и содержания электронного письма в форме с использованием переменной $_POST.**
   Ничего нового здесь нет. После нажатия кнопки «Отправить» в форме «Отправка электронной почты» (`sendemail.html`) все данные передаются сценарию `sendemail.php`, в котором мы присваиваем их значения переменным с небольшой помощью массива `$_POST`.

2. **Выполнение запроса SELECT к таблице email_list.**
   PHP-функция `mysqli_query()` выполняет запрос `SELECT`, в результате чего получает данные из листа рассылки. Так как нам необходимы все данные, имеющиеся в таблице, мы используем запрос вида `SELECT *`.

3. **Выборка данных о покупателях из результата запроса.**
   Выполнение запроса само по себе не предоставляет доступа к данным. Мы должны считать каждую строку данных в результате запроса, чтобы получить доступ к имени, фамилии и адресу электронной почты каждого покупателя.

4. **Вызов функции mail() для отправки электронных писем каждому покупателю.**
   Отправка электронных писем возможна в результате обращения к строкам листа рассылки, каждая из которых соответствует строке данных в результате запроса. Цикл, который мы здесь создаем, начинается с первой строки данных, затем переходит на вторую и повторяет такой переход для всех оставшихся строк результата, полученного после выполнения запроса `SELECT`. Цикл прекращается после обработки последней строки данных.



В первую очередь — извлечение данных

У вас уже богатый опыт по извлечению данных из форм в PHP, поэтому для вас нет ничего нового в выполнении первого этапа. Просто используйте суперглобальную переменную `$_POST` для того, чтобы сохранить значения темы электронного письма и его содержания в переменных PHP. Пока вы здесь, давайте будем двигаться дальше и присвоим эти значения переменным, которые понадобятся нам позднее для отправки электронных писем.

Адрес электронной почты Элмера сохранен в переменной, чтобы мы точно знали, где он находится, если когда-нибудь возникнет необходимость его изменить.

```
$from = 'elmer@makemeelvis.com';
$subject = $_POST['subject'];
$text = $_POST['elvismail'];
```

Данные формы «Отправка электронной почты» также сохранены в переменных.

Остальные данные, необходимые сценарию `sendemail.php`, будут получены из базы данных MySQL Элмера. Для извлечения данных покупателей из таблицы `email_list` сценарию необходимо сделать запрос `SELECT`. В отличие от того, как мы делали это прежде, когда передавали запрос `SELECT` из MySQL-терминала и просто смотрели данные таблицы, на этот раз мы сделаем это из сценария `sendemail.php` вызовом функции `mysqli_query()`.

Переменная $query содержит SQL-запрос в виде строки текста.

Это наш запрос, в результате выполнения которого будут извлечены значения всех колонок таблицы email_list.

```
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
```

Функция mysqli_query() использует переменную, содержащую ссылку на соединение с базой данных ($dbc) и строку запроса ($query).

Ссылка на соединение с базой данных необходима для передачи запроса на выполнение. Детали, касающиеся этого соединения, содержатся в переменной $dbc.

Нет. В действительности переменная $result непосредственно не содержит никаких данных таблицы.

Если вы попытаетесь вывести значение переменной `$result` непосредственно с помощью команды `echo`, вы увидите что-нибудь подобное:

```
Resource id #3
```

Переменная `$result` содержит номер идентификатора ресурса MySQL, а не непосредственные данные, полученные в результате выполнения запроса. Происходит это следующим образом: MySQL-сервер временно сохраняет результат выполнения вашего запроса и присваивает ему номер ресурса, по которому его можно идентифицировать. Затем вы передаете этот идентификатор функции PHP `mysqli_fetch_array()`, которая извлекает данные по одной записи за раз.

Функция mysqli_fetch_array() извлекает результаты запроса

Как только ваш запрос выполнен, вы можете получить доступ к его результатам, используя переменную `$result`. Функция `mysqli_fetch_array()` использует эту переменную для того, чтобы получить данные таблицы по одной записи за один раз. Каждая запись данных возвращается в виде массива, значение которого мы можем сохранять в переменной `$row`.

Эта функция извлекает запись данных из результатов запроса и сохраняет эти данные в виде массива.

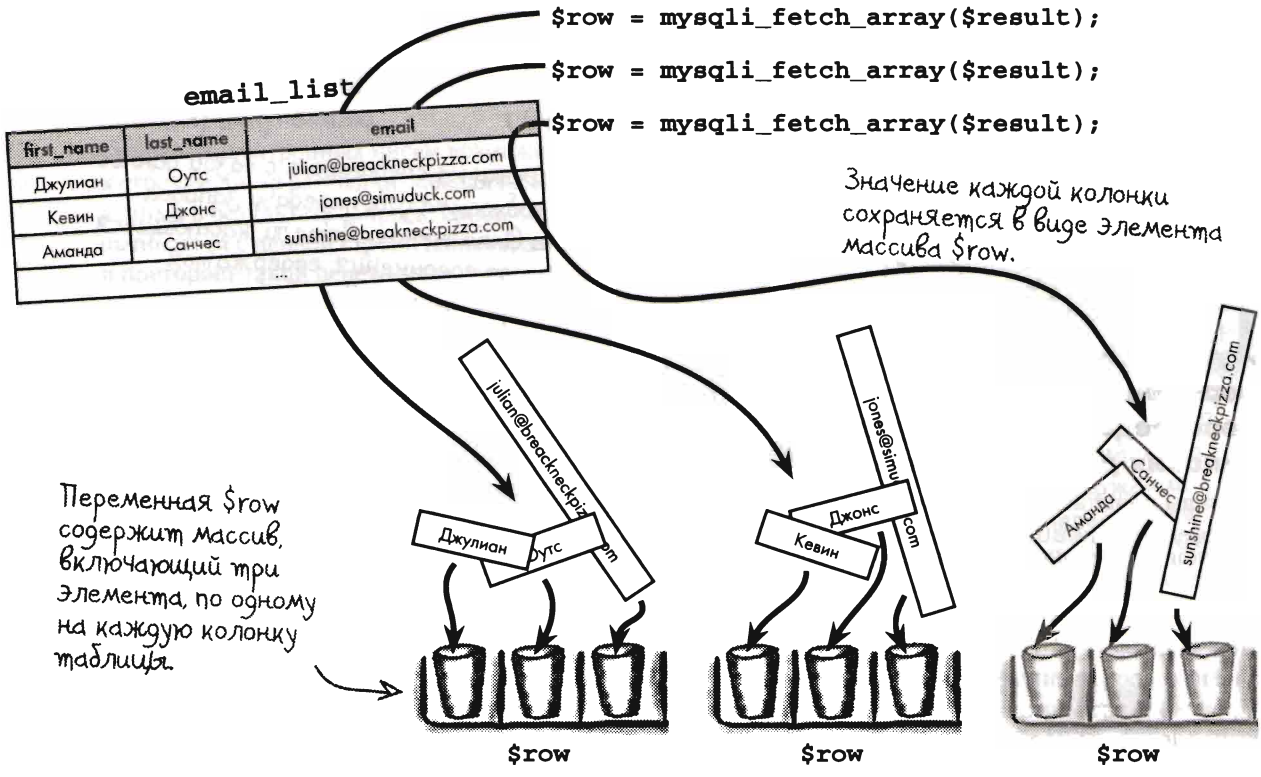
```
$row = mysqli_fetch_array($result);
```

Переменная $row — это массив, который первоначально содержит первую запись данных нашего результата.

Каждый SQL-запрос имеет свой собственный идентификационный номер, который используется для доступа к данным, ассоциированным с результатом этого запроса.

Каждый раз, когда этот код выполняется на веб-сервере, запись данных из результатов запроса сохраняется в массиве `$row`. Вы многократно вызываете функцию `mysqli_fetch_array()` для доступа к данным каждой записи результатов вашего запроса. Таким образом, первые три вызова функции `mysqli_fetch_array()` извлекут первые три записи таблицы, сохраняя значения колонок в виде элементов массива `$row`.

Функция mysqli_fetch_array() сохраняет запись данных из результатов запроса в массив.



Время поработать

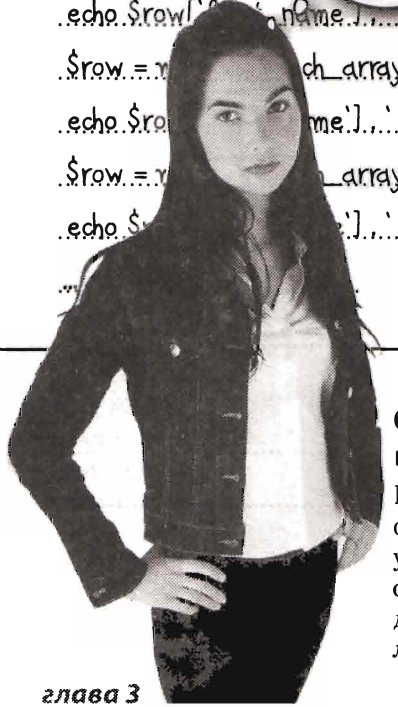
Для того чтобы убедиться, что вы действительно можете извлечь данные по одной записи за раз, закончите код PHP так, чтобы, используя команду `echo`, вывести имя, фамилию и адрес электронной почты каждого покупателя, содержащегося в таблице `email_list`.

```
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
$row = mysqli_fetch_array($result);
```

Решение задачи

Для того чтобы убедиться, что вы действительно можете извлечь данные по одной записи за раз, закончите код PHP так, чтобы, используя команду `echo`, вывести имя, фамилию и адрес электронной почты каждого покупателя, содержащегося в таблице `email_list`.

```
$query = "SELECT * FROM email_list";

$result = mysqli_query($dbc, $query);

$row = mysqli_fetch_array($result);

echo $row['first_name'].' '.$row['last_name'].': '.$row['email'].'<br />';
 $row = mysqli_fetch_array($result);
echo $row['first_name'].' '.$row['last_name'].': '.$row['email'].'<br />';
$row = mysqli_fetch_array($result);
echo $row['first_name'].' '.$row['last_name'].': '.$row['email'].'<br />';
$row = mysqli_fetch_array($result);
echo $row['first_name'].' '.$row['last_name'].': '.$row['email'].'<br />';
$row = mysqli_fetch
echo $row['first_
$row = mysqli_fet
echo $row[' name'].
$row = r ch_array($result);
echo $ro me'].' '.$row['last_name'].': '.$row['email'].'<br />';
$row = r _array($result);
echo $ ].' '.$row['last_name'].': '.$row['email'].'<br />';
```



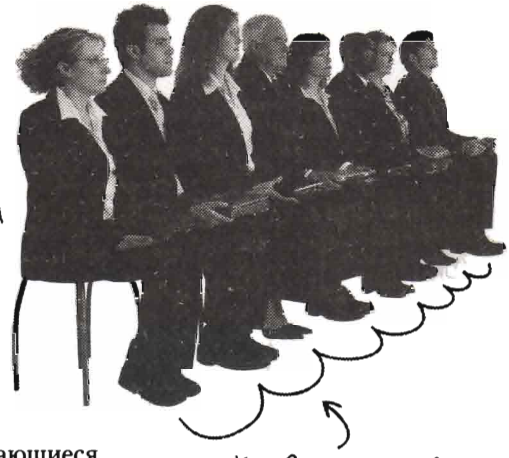
Существует более приемлемый метод: нам необходим цикл.

Цикл — это механизм в языке PHP, который повторяет фрагмент кода до тех пор, пока соблюдается определенное условие, например до того момента, пока есть данные для обработки. Таким образом, код в цикле будет повторяться для каждой записи данных в результатах запроса, выполняя любые необходимые нам действия в процессе этих переходов.

Повторение операции в цикле while

Цикл `while` («пока») — это цикл, в котором код повторяется снова и снова в течение всего времени, **пока соблюдается какое-либо условие**. Например, в вашем приложении по техническому обслуживанию может быть переменная с именем `$got_customers`, значение которой зависит от того, нуждается ли кто-нибудь из покупателей в обслуживании или нет. Если `$got_customers` имеет значение `true` («да», «истина»), это говорит о том, что у вас есть покупатели, нуждающиеся в обслуживании, и поэтому вам необходимо вызвать функцию `next_customer()`, чтобы принять следующего покупателя и обслужить его. Вот так этот сценарий может быть запрограммирован с использованием цикла `while`:

Цикл while повторяет код, пока (while) соблюдается какое-либо условие.



Код в цикле while повторяется для каждого покупателя до тех пор, пока не останется ни одного необслуженного.

Цикл повторяется снова и снова, пока есть хотя бы один покупатель.

```
while ($got_customers) {
  next_customer();
  ...
}
```

Этот код будет повторяться при каждом повторении цикла.

Заключение кода цикла в фигурные скобки дает возможность выполнять столько строк кода, сколько вам необходимо.

Когда мы пытаемся определить, имеются ли еще покупатели, нуждающиеся в обслуживании, мы **проверяем условие**. **Условие** — это код в скобках, следующий за ключевым словом `while`, и его значение или результат его интерпретации всегда должен выглядеть как ответ «Да»/«Нет» на поставленный вопрос. Если ответ «Да», или **true («истина»)**, то код выполняется, если же «Нет», или **false («ложь»)**, цикл прекращается и управление передается коду, следующему за ним.

Когда мы вызываем функцию `next_customer()` для обслуживания покупателей, мы **выполняем операцию**. В данном случае операция — это код, заключенный внутри фигурных скобок, который выполняется снова и снова, пока условие сохраняет значение `true`. Как только условие примет значение `false`, цикл прекращается, что приводит также и к прекращению выполнения операции.

Результат проверки условия всегда либо true, либо false — продолжение цикла (true) или прекращение цикла (false).

```
while (test_condition) {
  action
}
```

Операция выполняется один раз за один проход цикла.

Как вы думаете, цикл `while` может быть использован для рассылки электронных писем покупателям из таблицы Элмера `email_list`?

Обработка данных в цикле while

Применяя цикл `while` к данным электронной почты, давайте будем обрабатывать данные запись за записью без дублирования кода. Мы знаем, что функция `mysqli_fetch_array()` может извлекать одну запись таблицы и сохранять значения колонок в виде элементов массива `$row`, но функция сама не будет перемещаться по записям таблицы — она просто сохранит первую запись и на этом закончит обработку. Цикл `while` может вызывать функцию `mysqli_fetch_array()`, перемещаясь по каждой записи данных за раз, пока не достигнет последней.

Условием цикла while является значение, возвращаемое функцией mysqli_fetch_array(): true («истина») — если данные есть и false («ложь») — если их нет.

```
while($row = mysqli_fetch_array($result)) {
  echo $row['first_name'] . ' ' . $row['last_name'] .
    ' : ' . $row['email'] . '<br />';
}
```

Операция выполняется при каждом проходе цикла.

Операция цикла состоит из нескольких команд echo, соединяющих данные строки результата запроса, с добавлением символа новой строки в конце.

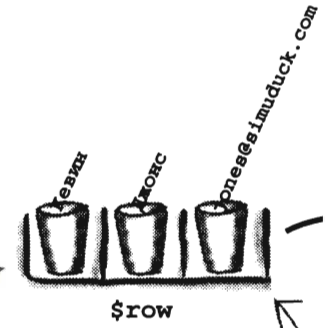
При первом проходе цикла массив $row содержит данные первой записи таблицы email_list.

email_list

first_name	last_name	email
Джулиан	Оутс	julian@breackneckpizza.com
Кевин	Джонс	jones@simuduck.com
Аманда	Санчес	sunshine@breakneckpizza.com
...		

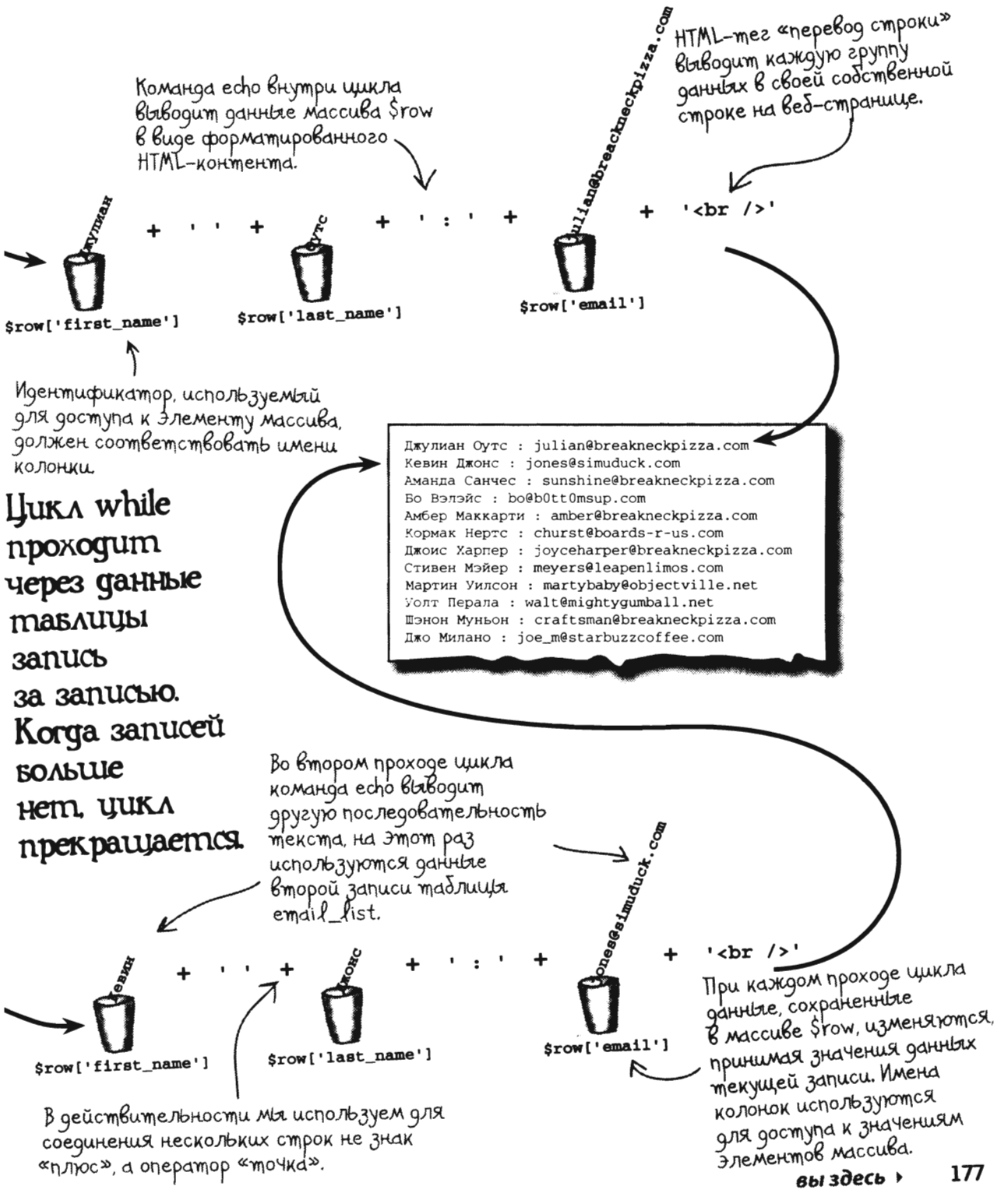
1-й проход цикла

2-й проход цикла



Во втором проходе цикла массив $row содержит данные второй записи таблицы email_list... Видите схему?

следующие проходы цикла...


Команда echo внутри цикла выводит данные массива $row в виде форматированного HTML-контента.
HTML-тег «перевод строки» выводит каждую группу данных в своей собственной строке на веб-странице.
Джулиан
Оутс
julian@breakneckpizza.com
+ ' ' + + ' : ' + + '<br />'
$row['first_name']
$row['last_name']
$row['email']
Идентификатор, используемый для доступа к элементу массива, должен соответствовать имени колонки.
Джулиан Оутс : julian@breakneckpizza.com
Кевин Джонс : jones@simuduck.com
Аманда Санчес : sunshine@breakneckpizza.com
Бо Вэлэйс : bo@b0tt0msup.com
Амбер Маккарти : amber@breakneckpizza.com
Кормак Нертс : churst@boards-r-us.com
Джоис Харпер : joyceharper@breakneckpizza.com
Стивен Мэйер : meyers@leapenlimos.com
Мартин Уилсон : martybaby@objectville.net
Уолт Перала : walt@mightygumball.net
Шэнон Муньон : craftsman@breakneckpizza.com
Джо Милано : joe_m@starbuzzcoffee.com
Цикл while проходит через данные таблицы запись за записью. Когда записей больше нет, цикл прекращается.
Во втором проходе цикла команда echo выводит другую последовательность текста, на этот раз используются данные второй записи таблицы email_list.
Кевин
Джонс
jones@simuduck.com
+ ' ' + + ' : ' + + '<br />'
$row['first_name']
$row['last_name']
$row['email']
В действительности мы используем для соединения нескольких строк не знак «плюс», а оператор «точка».
При каждом проходе цикла данные, сохраненные в массиве $row, изменяются, принимая значения данных текущей записи. Имена колонок используются для доступа к значениям элементов массива.

Не бывает глупых вопросов

В: **Каким образом цикл while определяет, что проход необходимо повторить? Я имею в виду тот факт, что цикл проверяет, какое из значений, true или false, принимает условие перед каждым проходом, в то время как функция mysqli_fetch_array() возвращает что-то вроде идентификатора ресурса, в котором содержится массив $row… Не похоже, чтобы это выглядело как проверка условия true/false.**

О: Хорошее наблюдение. Дело в том, что PHP достаточно либерален в вопросах, касающихся интерпретации условия `true`. Если опустить подробности, при проверке условия любое значение, которое не является нулем (0) или `false`, считается равным `true`. Поэтому, когда функция `mysqli_fetch_array()` возвращает массив данных, переменная `$row` (в контексте проверки условия) интерпретируется как `true`, так как она не является ни нулем (0), ни `false`.

В: **Значит, я могу использовать для управления циклом while данные любого типа, а не только те, которые имеют непосредственное значение true или false?**

О: Это так. Но имейте в виду, что в конечном итоге цикл `while` интерпретирует эти данные как `true` или `false`. Поэтому при интерпретации данных других типов очень важно понимать, что именно определяет значение проверяемого условия как `true` или `false`. И простой ответ заключается в том, что любое значение, которое не является нулем (0) или `false`, всегда считается равным `true`.

В: **Что произойдет с циклом while, если функция mysqli_fetch_array() возвратит пустой массив данных?**

О: Если в результатах запроса нет данных, это будет интерпретироваться как то, что функция `mysqli_fetch_array()` возвратила значение `false`. В результате цикл `while` не совершит ни одного прохода, что означает, что код внутри фигурных скобок не будет выполнен ни одного раза.

В: **Значит, возможен цикл, который не совершит ни одного прохода?**

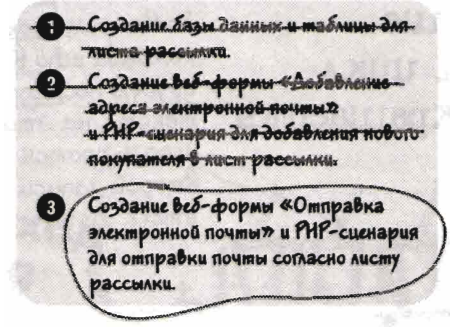
О: Это действительно так. Возможен также цикл, который будет повторять и повторять проходы, никогда не остановившись. Посмотрите на этот цикл `while`:

```
while (true) {
```

Такой цикл известен под именем бесконечный цикл, потому что проверка условия никогда не позволит ему прекратить проходы и продолжить ход выполнения программы. Бесконечный цикл — очень неприятная вещь.

КЛЮЧЕВЫЕ МОМЕНТЫ

- База данных — это контейнер для сохранения данных в высоко структурированной форме.
- Данные сохраняются в таблице в виде колонок и записей.
- SQL-запрос `CREATE DATABASE` используется для создания новой базы данных.
- SQL-запрос `CREATE TABLE` используется для создания таблицы внутри базы данных.
- Вы можете удалить таблицу из базы данных с помощью SQL-запроса `DROP TABLE`.
- Функция `mysqli_fetch_array()` извлекает строку данных из результатов запроса к базе данных.
- Цикл `while` повторяет фрагмент кода PHP до тех пор, пока соблюдается определенное условие.



Не забывайте: у нас все еще не выполнен этот последний этап для полного завершения работы.

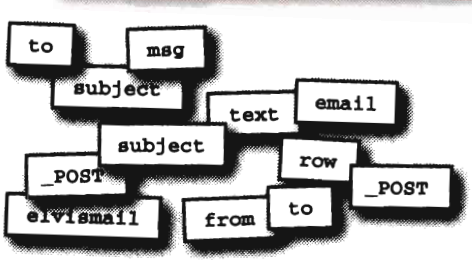
Магниты PHP и MySQL

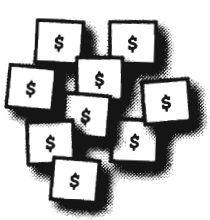
Используйте магниты, изображенные ниже, чтобы закончить код для сценария «Отправка электронной почты», что позволит Элмеру начать отправку электронных писем покупателям, включенным в его лист рассылки. Ниже — напоминание, как работает функция `mail()`:

```
mail(Кому, Тема, Содержание, 'From: ' . Обратный_адрес);
```

```
<?php
  $from = 'elmer@makemeelvis.com';

  $subject = ..... ;

  $text = ..... ;

  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    or die('Ошибка соединения с MySQL-сервером.');

  $query = "SELECT * FROM email_list";
  $result = mysqli_query($dbc, $query)
    or die('Ошибка при выполнении запроса к базе данных.');

  while($row = mysqli_fetch_array($result)) {
    $first_name = $row['first_name'];
    $last_name = $row['last_name'];

    $msg = "Уважаемый $fist_name $last_name,\n .....";

    $to = ..... ;

    mail( ..... , ..... , ..... , 'From:' . ..... );

    echo 'Электронное письмо отправлено: ' . ..... '<br

  }

  mysqli_close($dbc);
?>
```

sendemail.php





Магниты PHP и MySQL

Используйте магниты, изображенные ниже, чтобы закончить код для сценария «Отправка электронной почты», что позволит Элмеру начать отправку электронных писем покупателям, включенным в его лист рассылки. Ниже — напоминание, как работает функция `mail()`:

```
mail(Кому, Тема, Содержание, 'From: ' . Обратный_адрес);
```

Не забудьте изменить этот адрес электронной почты на свой собственный.

```
<?php
  $from = 'elmer@makemeelvis.com';

  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    or die('Ошибка соединения с MySQL сервером.');

  $query = "SELECT * FROM email_list";
  $result = mysqli_query($dbc, $query)
    or die('Ошибка при выполнении запроса к базе данных.');

  while($row = mysqli_fetch_array($result)) {
    $first_name = $row['first_name'];
    $last_name = $row['last_name'];

    $msg = "Уважаемый $fist_name $last_name,\n" . $text;

    $to = $row['email'];

    mail($to, $subject, $msg, 'From:' . $from);

    echo 'Электронное письмо отправлено: ' . $to . '<br />';

  }

  mysqli_close($dbc);
?>
```

Поле ввода темы на форме имеет имя subject. Это же самое имя используется для доступа к соответствующему элементу массива $_POST.

Поле ввода содержания электронного письма на форме имеет имя elvismail.

Полное содержание электронного письма состоит из имени покупателя и содержания электронного письма, введенного в соответствующее поле формы.

Функции mail() передаются в качестве аргументов адрес электронной почты получателя, тема и содержание электронного письма вместе с обратным адресом Элмера.

sendemail.php

Колонка таблицы с именем email содержит адрес электронной почты покупателя, которому должно быть отправлено электронное письмо.

На веб-страницу выводится сообщение о подтверждении отправки электронного письма с адресом электронной почты, на который оно было отправлено.

Обычно считается достаточно опасной практика передавать данные, введенные посетителем сайта, функции mail() непосредственно, без их предварительной проверки. В главе 6 излагаются некоторые технические приемы решения этой проблемы.

Тест-драйв

Отправка электронных писем покупателям, перечисленным в листе рассылки, с использованием формы «Отправка электронной почты».

Загрузите код веб-страницы «Отправка электронной почты» с сайта «Лаборатория "Очертя голову"» по адресу `www.headfirstlab.com/books/hfphp`. Он в каталоге `chapter03`. Так же как и код веб-страницы «Добавление адреса электронной почты», этот код включает веб-форму Элмера `sendemail.html`, каскадные таблицы стилей (`style.css`) и два файла изображений (`elvislogo.gif` и `blankface.jpg`).

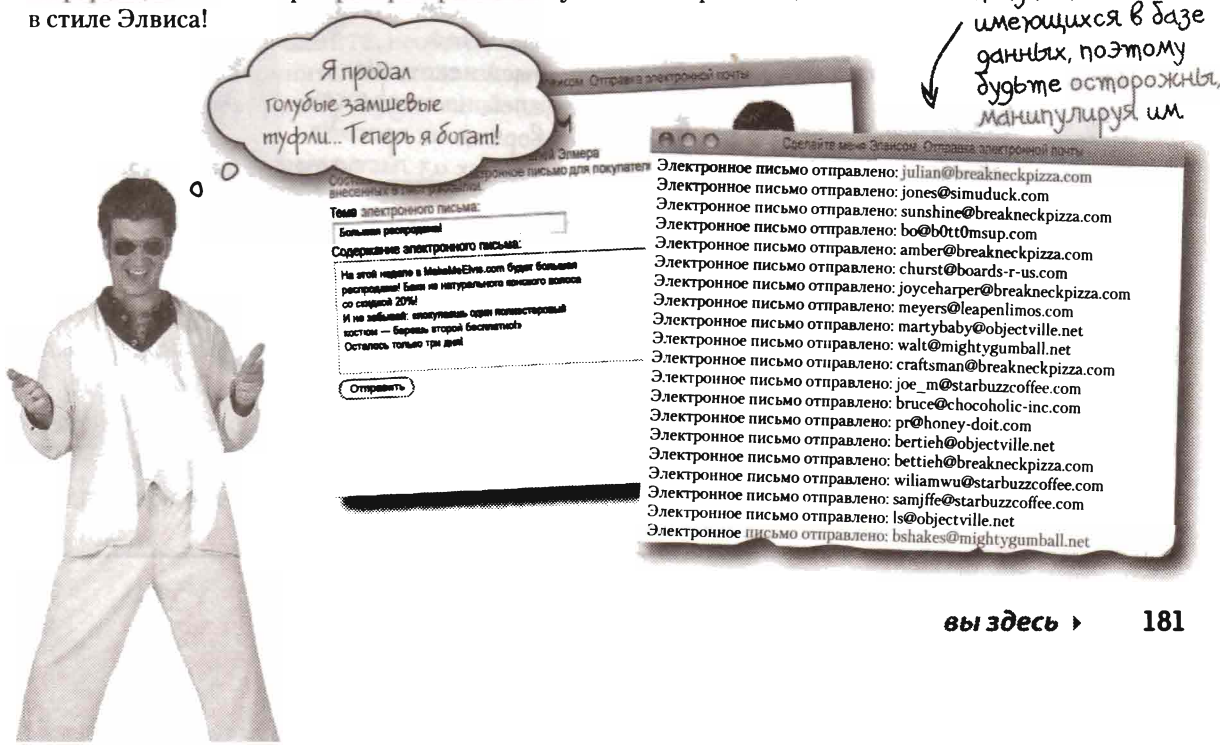
Имейте в виду, что ваш адрес электронной почты должен быть в листе рассылки, чтобы вы получили электронное письмо.

Создайте новый текстовый файл с именем `sendemail.php` и введите весь код, приведенный на предыдущей странице книги. Загрузите все файлы на ваш веб-сервер и откройте страницу `sendemail.html` в браузере.

Вы получили письмо... от Элмера!

Наконец Элмер может отправлять свои электронные письма, содержащие объявления о распродажах в магазине `MakeMeElvis.com` любому покупателю, включенному в его лист рассылки, используя свою новую веб-форму «Отправка электронной почты» и PHP-сценарий. Этот же сценарий, кроме того, используется для вывода подтверждения о том, что каждое письмо было успешно отправлено. Каждый раз, когда выполняется код в цикле `while`, он видит сообщение: «Электронное письмо отправлено кому-то@куда-то.com» с адресом электронной почты покупателя из его базы данных. В результате информация о его товарах распространяется лучше и — хорошо ли, плохо ли — в стиле Элвиса!

PHP-сценарий «Отправка электронной почты» действительно отправляет электронные письма на адреса покупателей, имеющихся в базе данных, поэтому будьте осторожны, манипулируя им.



Иногда люди хотят удалиться

На дороге любого процветающего бизнеса могут встречаться ухабы. Похоже, некоторые фанаты Элвиса дезертировали с корабля и хотят, чтобы их исключили из листа рассылки Элмера. Элмер хочет удовлетворить их желание, но это означает, что ему необходима возможность удалять записи о покупателях из базы данных.

Уважаемый Элмер!

Я не хочу больше получать электронные письма о распродажах в магазине Элвиса. Я остаюсь поклонником Элвиса, но не могу больше принимать участие в вашем предприятии. Удалите меня, пожалуйста, из вашего листа рассылки. Мой адрес электронной почты:

cbriggs@boards-r-us.com

Благод

Экс-и

Уважаемый друг Бедро-вращатель!

Хотя я все еще обожаю неповторимые движения Элвиса, но так сильно он меня уже не впечатляет. Сейчас я предпочитаю неповторимые способности Либерейс в организации концертов и ее мастерство пианистки. Это мой адрес электронной почты (удалите, пожалуйста, меня из списка рассылки):

ndy@tikibeanlounge.com

кренне Ваша,

Уважаемый сэр!

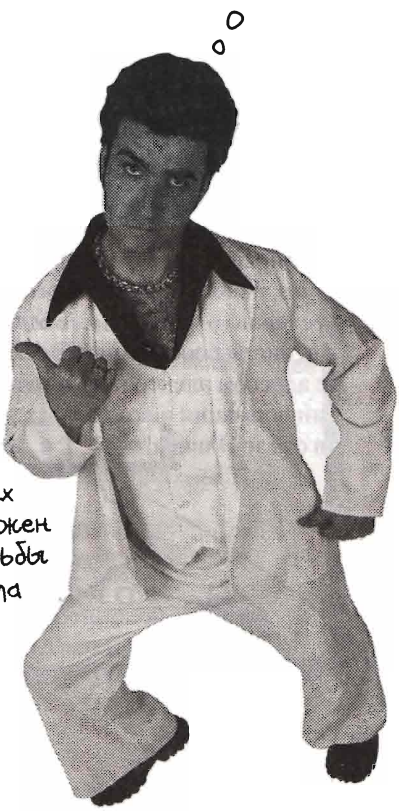
После нескольких аллергических реакций на ваши подлинные баки из конского волоса я решил, что, видимо, подражание внешнему виду Элвиса — это не мой стиль. Мне нравится наряжаться, но баки — это, пожалуй, слишком. Удалите, пожалуйста, меня из списка рассылки.

Искренне Ваш,

Брайан Пауэрс

bp@honet-doit.com

Я полагаю, не каждый способен имитировать короля. Мне необходимо удалить этих людей из списка рассылки и сосредоточиться на настоящих фанах.



Элмер не слишком доволен тем, что он лишается некоторых покупателей, но он должен удовлетворять их просьбы на удаление их из листа рассылки.

Это факт MySQL-жизни: иногда вам приходится удалять данные из своей базы. Элмер должен усовершенствовать свое приложение и найти возможность удалять покупателей из таблицы `email_list`.

Запишите новые компоненты приложения, которые, как вы думаете, будут необходимы Элмеру для обеспечения функции «Удаление адреса электронной почты».

Удаление данных с помощью запроса DELETE

Для того чтобы удалить данные из таблицы, нам необходимо выполнить запрос DELETE («удалить»). Мы будем использовать запрос DELETE в новом сценарии «Удаление адреса электронной почты», который удаляет данные покупателя из листа рассылки Элмера. Фактически нам необходимы новый сценарий и новая веб-форма, чтобы приводить этот сценарий в действие... но сначала мы должны удалить (DELETE) данные.

1. ~~Создание базы данных и таблицы для листа рассылки.~~
2. ~~Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.~~
3. ~~Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.~~
4. Создание веб-формы «Удаление адреса электронной почты» и PHP-сценария для удаления покупателя из листа рассылки.

Похоже, нам необходима новая ступень... Иногда планы меняются!

SQL-запрос DELETE удаляет запись из таблицы. Вы должны соблюдать осторожность при использовании этого запроса, так как в результате его выполнения в мгновение ока таблица может оказаться совершенно пустой. С учетом этого ниже приведена наиболее опасная форма запроса DELETE, в результате выполнения которого удаляются все записи из таблицы.

```
DELETE FROM имя_таблицы
```

Это имя таблицы, из которой вы хотите удалить записи.

Без какого-либо ограничивающего условия в результате выполнения запроса DELETE таблица будет очищена от всех содержащихся в ней данных.

Значит, мы никак не можем удалить из таблицы что-либо без того, чтобы удалить все?

Нет, совершенно не так. Запрос DELETE можно использовать с точным указанием того, какая конкретно запись должна быть удалена.

Для того чтобы точно указать, какую конкретно запись или записи вы хотите удалить с помощью запроса DELETE, необходимо добавить ограничивающее условие WHERE («где»). Помните, как использование этого выражения в запросе SELECT позволяло нам выделить определенные записи в запросе?

Время поработать

Предположим, у Элмера есть 23 покупателя с именем Анна, 11 покупателей с фамилией Паркер и один покупатель с именем и фамилией Анна Паркер. Напишите, сколько записей будет удалено при выполнении каждого из этих запросов:

```
DELETE FROM email_list WHERE first_name = 'Анна';

DELETE FROM email_list WHERE first_name = 'Анна' OR last_name = 'Паркер';

DELETE FROM email_list WHERE last_name = Паркер;
```

Решение задачи

Предположим, у Элмера есть 23 покупателя с именем Анна, 11 покупателей с фамилией Паркер и один покупатель с именем и фамилией Анна Паркер. Напишите, сколько записей будет удалено при выполнении каждого из этих запросов:

```
DELETE FROM email_list WHERE first_name = 'Анна';
```
23...

```
DELETE FROM email_list WHERE first_name = 'Анна' OR last_name = 'Паркер';
```
33...

```
DELETE FROM email_list WHERE last_name = Паркер;
```
0.....

Вопрос с подвохом! Фамилия покупателя не заключена в кавычки, поэтому ни одной записи удалено не будет. Все текстовые данные должны заключаться в кавычки.

Использование ограничивающего условия WHERE для удаления определенных данных

Используя ограничивающее условие `WHERE` в запросе `DELETE`, мы определяем конкретный перечень записей, которые должны быть удалены, вместо того чтобы опустошать всю таблицу. Выражение `WHERE` позволяет нам сфокусироваться только на тех записях, которые мы хотим удалить. В данном случае — на данных одного из покупателей Элмера, который пожелал быть удаленным из листа рассылки.

```
DELETE FROM email_list
  WHERE email = 'pr@honey-doit.com'
```

Значение, которому должен соответствовать адрес электронной почты, чтобы запись была удалена.

Имя колонки таблицы.

Эта часть выражения WHERE производит проверку каждой записи таблицы на соответствие заданному значению.

Фактически проверка на соответствие условию `WHERE` производится путем сравнения значения колонки с заданным значением для каждой записи в таблице. В этом примере знак равенства (=) говорит о том, что будут удалены только те записи, для которых значение колонки `email` равно `pr@honey-doit.com`.

Напишите, как вы думаете, почему в ограничивающем условии запроса использовалась колонка email, а не first_name или last_name:

.....

Выражение WHERE ограничивает запрос, распространяя его только на определенные записи.

Минимизация риска случайного удаления

Важно понимать, что, хотя для проверки соответствия записи ограничивающему условию в выражении `WHERE` может быть использовано имя любой колонки, существует очень веское основание для выбора колонки `email` в запросе `DELETE` Элмера. Совершенно понятно, что если более одной записи отвечают требованиям ограничивающего условия `WHERE`, то все они будут удалены. Поэтому очень важно, чтобы ограничивающее условие `WHERE` точно соответствовало записи, которую вы хотите удалить.

Выражение WHERE ограничивает запрос, распространяя его только на определенные записи.

```
DELETE FROM email_list
  WHERE email = 'pr@honey-doit.com'
```

Использование колонки email в ограничивающем условии WHERE помогает установить уникальность и снижает риск случайного удаления записи.

В результате выполнения запроса DELETE эта запись будет удалена из таблицы... и вы никогда больше ее не увидите!

email_list

first_name	last_name	email
...		
Джо	Милькно	joe_m@starbuzzcoffee.com
Брюс	Спенс	bruce@chocoholic-inc.com
Пат	Райс	pr@honey-doit.com
Берти	Хендерсон	bertieh@objectville.net
Грег	Эскштеин	gregeck@breakneckpizza.com
Вильма	Ву	wilmawu@starbuzzcoffee.com
Сэм	Джеффи	samjaffe@starbuzzcoffee.com
Луи	Шеффер	ls@objectville.net
Буба	Шекспир	bshakes@mightygumball.net
Джон	До	johndoe@tikibeanlounge.com
Пэт	Громмет	grommetp@simuduck.com
...		

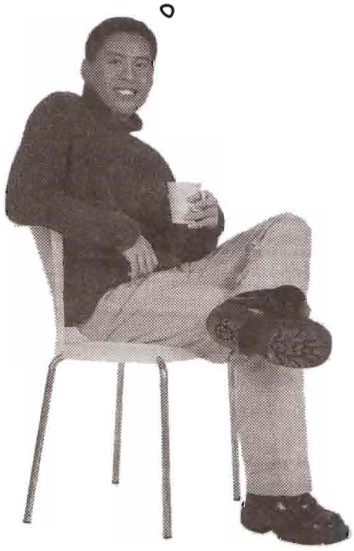
Если бы мы использовали в ограничивающем условии WHERE колонку first_name вместо колонки email, этот покупатель был бы также удален.

```
Файл Правка Окно Помощь Прощай
mysql> DELETE FROM email list WHERE email = 'pr@honey-doit.com';
1 запись удалена (0.005 сек)
```

Тест-драйв

Попробуйте применить запрос DELETE к базе данных Элмера.

Загрузите инструментальную программу MySQL и попробуйте удалить несколько записей из таблицы `email_list` с помощью запроса `DELETE`. Идентифицируйте удаляемые записи, основываясь на адресах электронной почты покупателей. Не забывайте включать ограничивающее условие `WHERE` в каждый запрос `DELETE`, чтобы случайно не очистить всю таблицу.

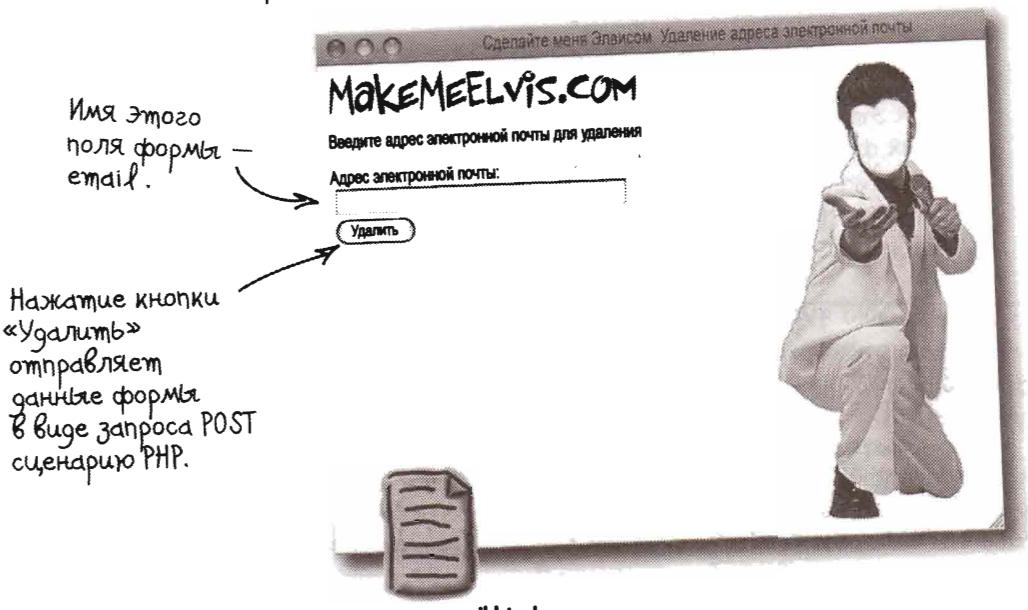


Это верно. Удаление покупателей вручную, с помощью индивидуальных запросов — никуда не годный способ обработки листа рассылки.

Так как Элмер рано или поздно непременно столкнется с ситуацией, когда покупатель пожелает, чтобы его удалили из листа рассылки, имеет смысл разработать пользовательский веб-интерфейс для удаления покупателей. HTML-форма и PHP-сценарий должны решить эту задачу, используя запрос `DELETE FROM` с небольшой помощью ограничивающего условия `WHERE`.

УПРАЖНЕНИЕ

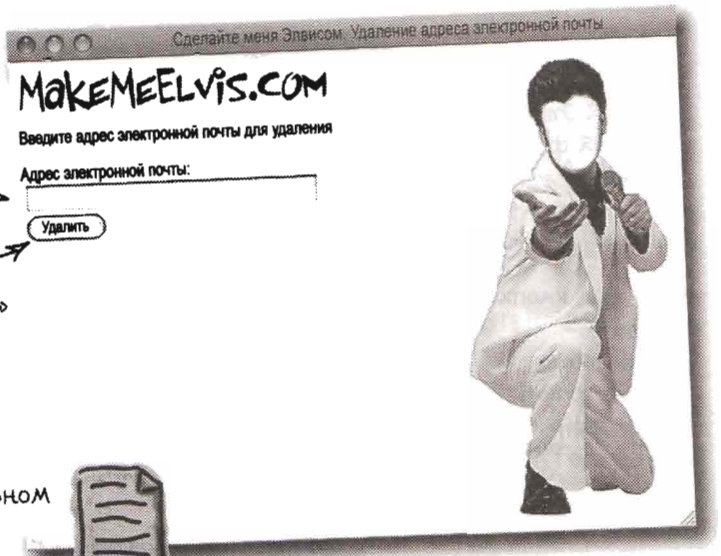
Элмер создал веб-форму (`removeemail.html`) для удаления покупателя из своего листа рассылки. Все, что необходимо при работе с этой формой, — это ввести в поле с именем `email` адрес электронной почты. Закончите код сценария Элмера `removeemail.php`, который вызывается формой для удаления покупателя из листа рассылки.




```
<?php

  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')

    or die('Ошибка соединения с MySQL-сервером.');









  mysqli_close($dbc);

?>
```

removeemail.php

Элмер создал веб-форму (`removeemail.html`) для удаления покупателя из своего листа рассылки. Все, что необходимо при работе с этой формой, — это ввести в поле с именем `email` адрес электронной почты. Закончите код сценария Элмера `removeemail.php`, который вызывается формой для удаления покупателя из листа рассылки.



Имя этого поля формы — email.

Нажатие кнопки «Удалить» отправляет данные формы в виде запроса POST сценарию PHP.

Адрес электронной почты, содержащийся в суперглобальном массиве $_POST, сохраняется в переменной и используется затем в запросе DELETE.

removeemail.html

```
<?php

  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    or die('Ошибка соединения с MySQL-сервером.');

  $email = $_POST['email'];

  $query = "DELETE FROM email_list WHERE email = '$email'";
  mysqli_query($dbc, $query)
    or die('Ошибка выполнения запроса к базе данных');

  echo 'Покупатель удален: ' . $email;

  mysqli_close($dbc);
?>
```

Будьте внимательны с двойными и одинарными кавычками. Двойные кавычки ограничивают здесь весь SQL-запрос, в то время как одинарные ограничивают адрес электронной почты, сохраненный накануне в переменной $email.

Никогда не вредно вывести подтверждение выполнения операции, особенно когда это касается удаления данных из базы.

Не забывайте убрать все после себя, закрыв соединение с базой данных.

removeemail.php

Вот и все, мы наконец-то закончили!

1. ~~Создание базы данных и таблицы для листа рассылки.~~
2. ~~Создание веб-формы «Добавление адреса электронной почты» и PHP-сценария для добавления нового покупателя в лист рассылки.~~
3. ~~Создание веб-формы «Отправка электронной почты» и PHP-сценария для отправки почты согласно листу рассылки.~~
4. ~~Создание веб-формы «Удаление адреса электронной почты» и PHP-сценария для удаления покупателя из листа рассылки.~~

Тест-драйв

Удалите покупателя из листа рассылки, используя форму «Удаление адреса электронной почты».

Возникает чувство, что это вам уже немного знакомо, а? Загрузите код для веб-страницы «Удаление адреса электронной почты» с сайта «Лаборатория “Очертя голову”» по адресу `www.headfirstlabs.com/books/hfphp`. Этот код находится в каталоге `chapter03` и включает веб-форму Элмера `removeemail.html`, каскадные таблицы стилей (`style.css`) и два файла изображений (`elvislogo.gif` и `blankface.jpg`).

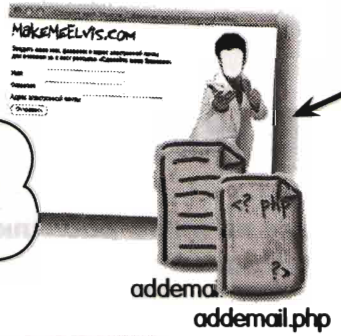
А теперь создайте новый текстовый файл `removeemail.php` и введите весь код предыдущей страницы.

Загрузите все файлы на ваш веб-сервер и откройте страницу `removeemail.html` в веб-браузере. Введите в форму адрес электронной почты пользователя и нажмите кнопку «Удалить» для того, чтобы удалить пользователя из базы данных.

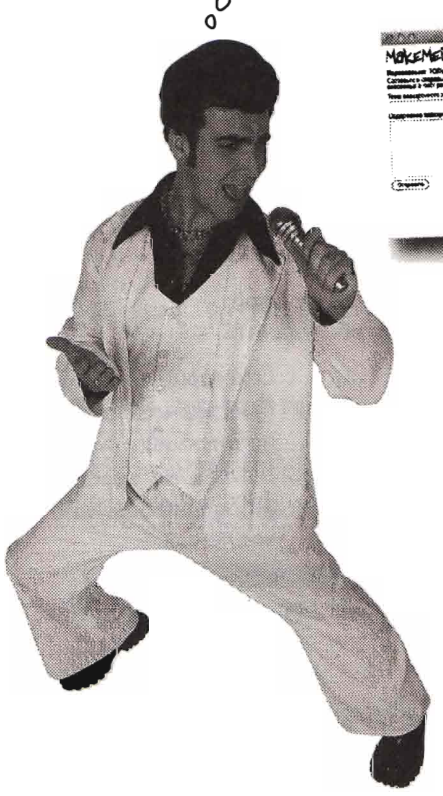
Сценарий делает всю черновую работу по отправке запроса DELETE на сервер и подтверждению удаления.

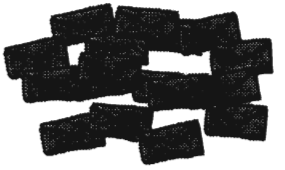
MakeMeElvis.com — это веб-приложение

Это официально. Благодаря PHP и MySQL сайт Элмера `MakeMeElvice.com` заслуживает того, чтобы называться приложением. Элмер теперь может как угодно долго хранить данные в базе данных MySQL и обмениваться с ней данными через веб-форму. Совокупность HTML-страниц, PHP-сценариев и встроенных SQL-запросов предоставляет Элмеру возможность добавлять покупателей в лист рассылки и удалять их из него (они также могут добавлять себя в лист рассылки самостоятельно). Кроме того, Элмер может отправлять электронные письма всем покупателям, перечисленным в листе рассылки.

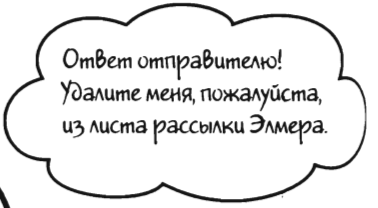


Страница «Добавление адреса электронной почты» добавляет нового покупателя в лист рассылки Элмера.





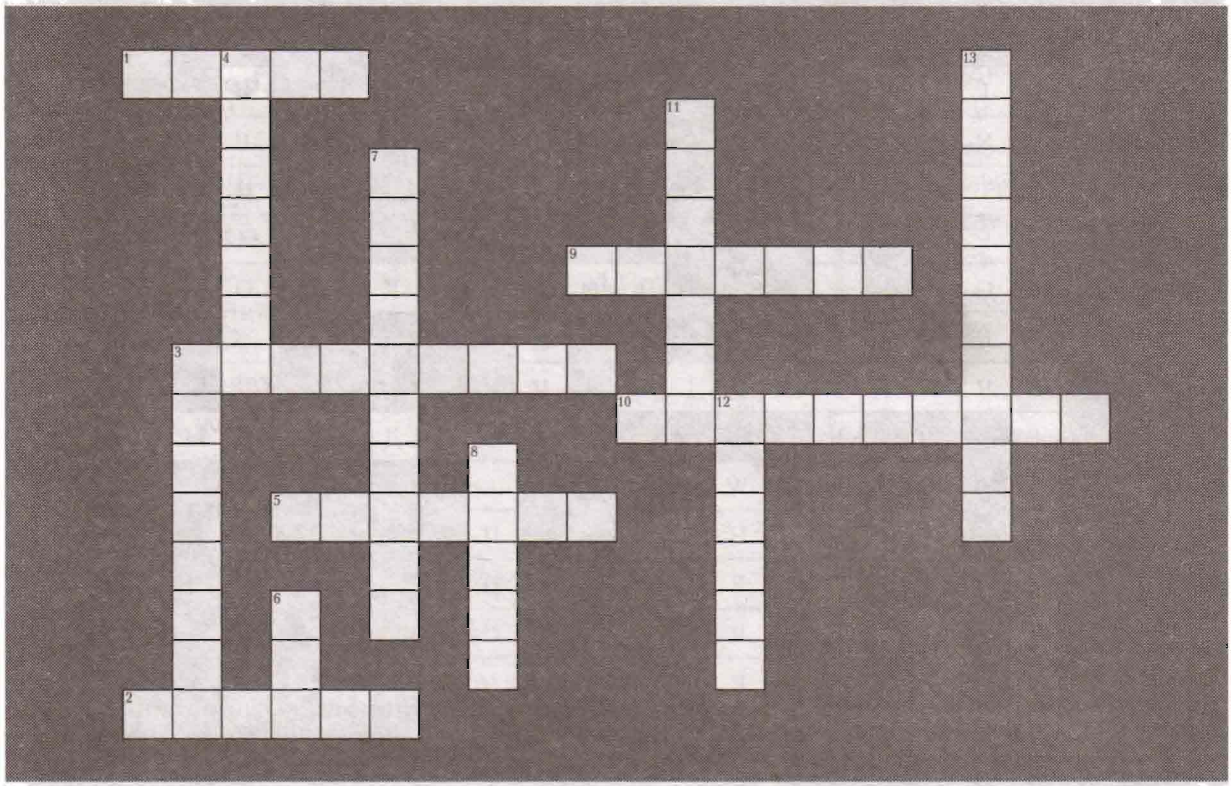
Страница «Отправление электронной почты» отправляет электронные письма каждому покупателю из перечисленных в листе рассылки простым нажатием одной кнопки.



Страница «Удаление адреса электронной почты» удаляет покупателя из листа рассылки.

Кроссворд PHP и MySQL

Как только вы закончите совершенствовать танцевальные движения Элмера, попробуйте решить этот кроссворд.



По горизонтали

1. Продолжает выполнять какие-либо операции, пока определенное условие остается истинным.

2. Используйте этот SQL-запрос для извлечения записей данных из таблицы.

3. Этот SQL-запрос удаляет всю таблицу из базы данных.

5. Используйте этот тип данных MySQL для сохранения текста переменной длины.

9. Здесь в таблице содержатся данные определенного типа.

10. Постоянная, высоко систематизированная и структурированная совокупность данных, которая обычно сохраняется в файле на жестком диске.

По вертикали

3. Используйте этот SQL-запрос для просмотра структуры таблицы.

4. Тип данных MySQL для сохранения целых чисел.

6. После создания новой базы данных в MySQL-терминале вы должны выполнить этот запрос, прежде чем сможете делать какие-либо манипуляции с базой данных.

7. Используйте этот SQL-запрос для удаления записей из таблицы.

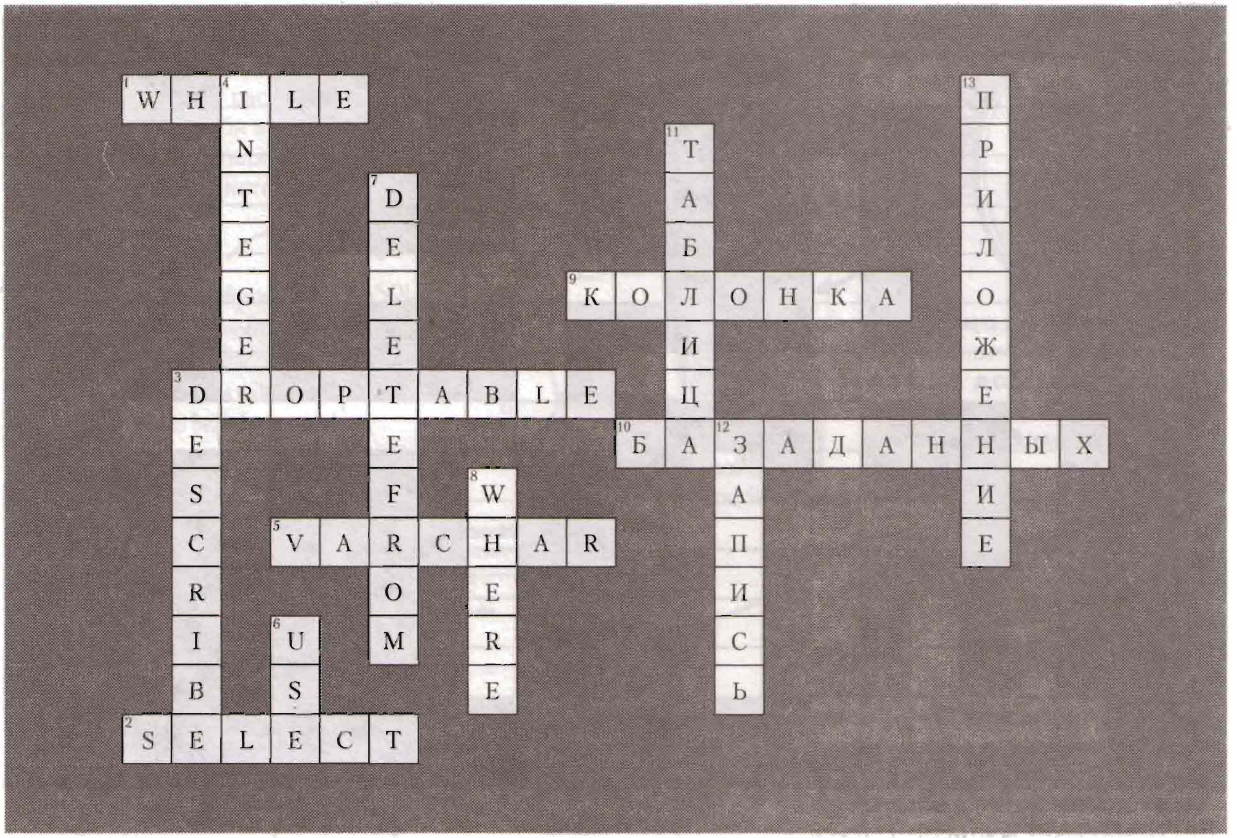
8. Ограничивающее условие; может быть добавлено к SQL-запросу для указания записей, на которые распространяется данный запрос.

11. Из них состоит база данных MySQL.

12. Полный набор данных в таблице, содержащий по одному значению для каждой колонки.

13. После добавления к веб-странице динамической функциональности она становится _____ .

Кроссворд PHP и MySQL. Решение



Ваш инструментарий PHP и MySQL

Вы не только помогли Элмеру усовершенствовать его веб-приложение, но и приобрели при чтении этой главы определенные ценные навыки по использованию PHP и MySQL. Например...

while

Циклическая конструкция PHP. Она позволяет вам многократно выполнять один и тот же фрагмент кода до тех пор, пока выполняется определенное условие. Одной из особенно удобных областей использования этой конструкции является последовательное извлечение данных запись за записью из результатов SQL-запроса.

DROP TABLE *имя_таблицы*

В результате выполнения этого SQL-запроса таблица полностью удаляется из базы данных. Это означает, что таблица удаляется вместе со всеми данными, которые были в нее помещены.

mysqli_fetch_array()

Эта встроенная PHP-функция извлекает одну строку данных из результатов запроса к базе данных. Вы можете вызывать эту функцию многократно, чтобы последовательно извлекать данные строка за строкой.

WHERE

Это SQL-выражение используется в дополнение к SQL-запросам в качестве ограничивающего условия. Такое сочетание позволяет составить запрос, который распространяется не на все записи в таблице, а только на вполне определенные. Например, вы можете выделить только те записи, для которых данные колонок соответствуют какому-либо значению.

DESCRIBE *имя_таблицы*

Если вам необходимо узнать структуру таблицы, этот запрос — то, что нужно. В результате выполнения этого SQL-запроса вы не увидите никаких данных, содержащихся в таблице, но узнаете имена всех колонок и типы данных, которые могут быть в них сохранены.

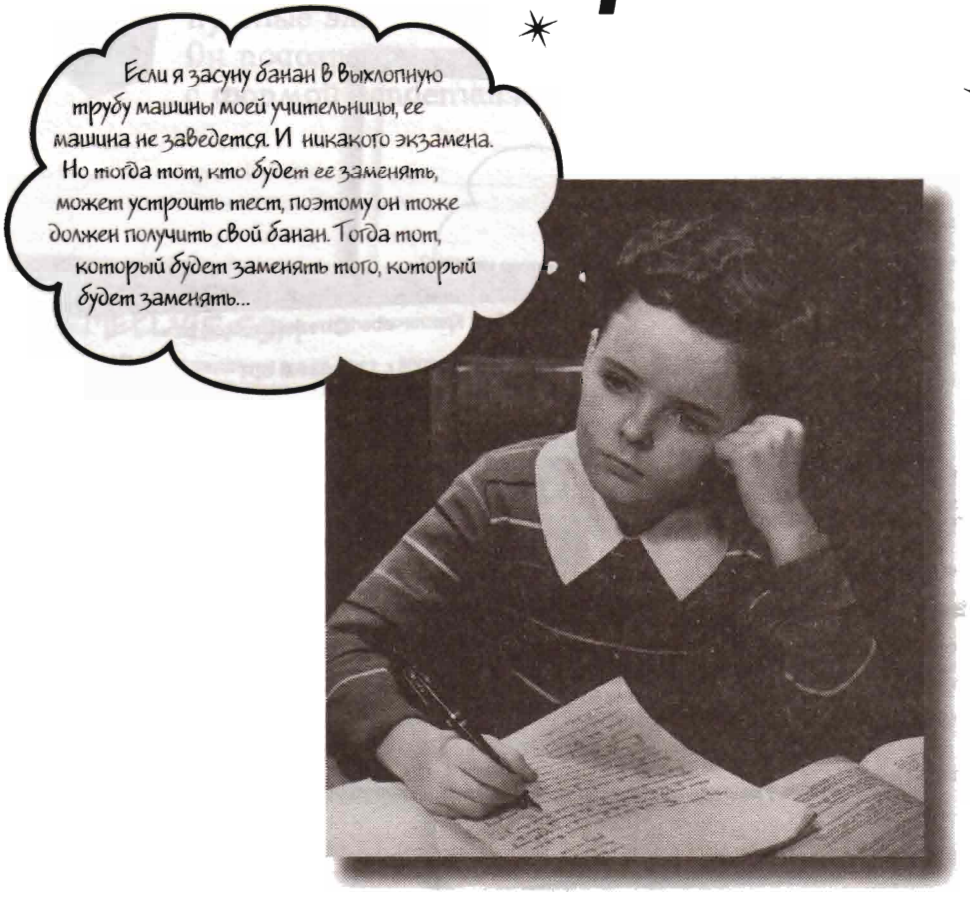
DELETE FROM *имя_таблицы*

Используйте этот SQL-запрос для удаления записей из таблицы. В зависимости от того, как вы используете этот запрос, вы можете удалить одну или несколько записей.

SELECT * FROM *имя_таблицы*

В результате выполнения этого SQL-запроса извлекаются данные из таблицы. При использовании символа звездочки (*) извлекаются данные для всех колонок. Если вам не нужны данные из всех колонок, можете сузить круг этих колонок, перечисляя их имена вместо символа звездочки.

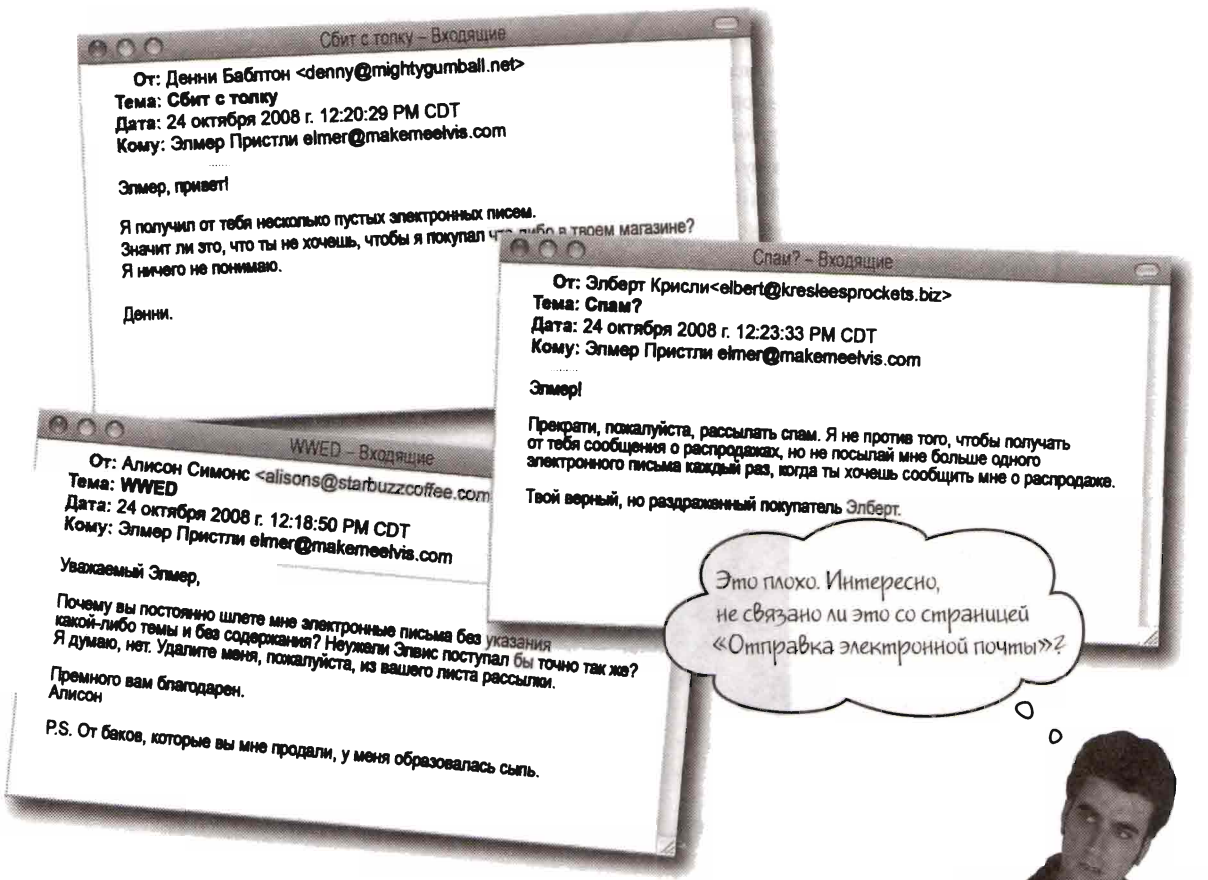
Ваше веб-приложение



Иногда надо реально смотреть на вещи и менять свои планы. Или планировать более внимательно с самого начала. После загрузки приложения на веб-сервер вы, возможно, обнаружите, что спланировали его недостаточно хорошо. Некоторые решения, которые, как вы предполагали, будут работать, в реальной жизни оказались не слишком удачными. В этой главе мы рассмотрим некоторые *проблемы реальной жизни*, которые могут возникнуть, после того как **ваше приложение выйдет из режима тестирования и переместится в Интернет**. По пути мы познакомим вас с примерами более важного кода PHP и SQL.

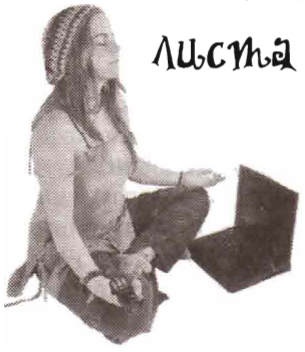
У Элмера появились раздраженные покупатели

Лист рассылки Элмера стремительно разросся, но его электронные письма в ряде случаев повлекли за собой некоторые жалобы. Они носили различный характер, но, похоже, все сводились к тому, что покупатели получали пустые электронные письма или сразу по нескольку писем, ни одно из которых не содержало ничего полезного. Элмеру необходимо выяснить причину этого явления и исправить положение. От этого зависит его бизнес.



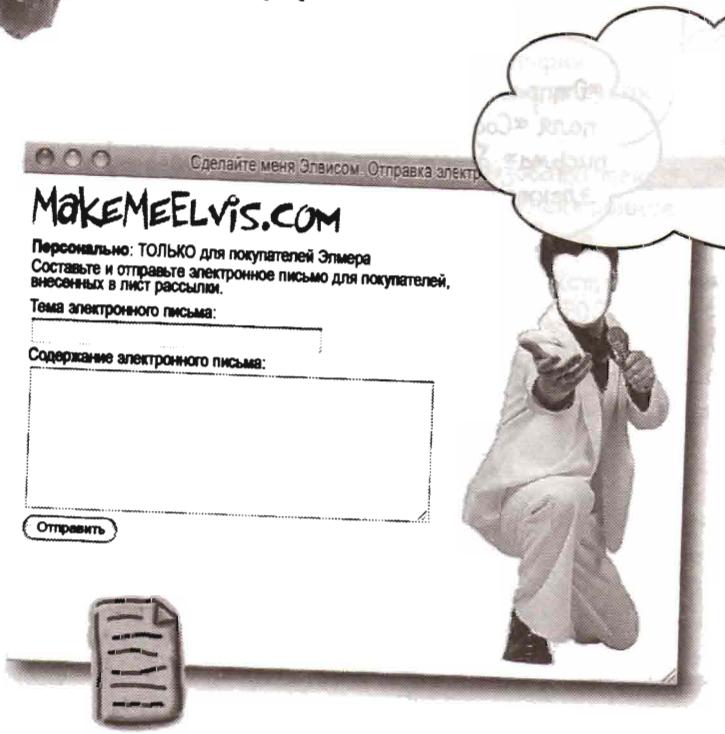
Элмер знает, что столкнулся с проблемой и ему будет нужна помощь в том, чтобы понять, что же происходит в действительности.

Станьте администратором листа рассылки Элмера



Ваша задача — сыграть роль Элмера и выяснить, как отправляются пустые электронные письма. Он подозревает, что все это связано с формой sendemail.html.

Напишите, в чем заключается проблема по мнению Элмера.



Станьте администратором листа рассылки Элмера

Ваша задача — сыграть роль Элмера и выяснить, как отправляются пустые электронные письма. Он подозревает, что все это связано с формой sendemail.html.

Напишите, в чем заключается проблема по мнению Элмера.

Если нажать кнопку «Отправить» без заполнения поля «Содержание электронного письма», будет отправлено пустое электронное письмо.

Если нажать кнопку «Отправить» в момент, когда в поле ввода «Содержание электронного письма» ничего нет, будет отправлено пустое электронное письмо. То же самое относится к полю ввода «Тема электронного письма».

Защита Элмера от... Элмера

Итак, основная проблема здесь заключается в «человеческом факторе». Элмер по оплошности нажал кнопку «Отправить» до того, как ввел содержание электронного письма, и пустое письмо отправилось всем покупателям, включенным в лист рассылки. Достаточно небезопасно считать, что веб-форма всегда будет использоваться строго так, как это предусмотрено ее создателем. Вот почему это исключительно ваша, бдительный создатель PHP-сценария, задача — протестировать и устранить проблемы подобного рода, пытаясь предугадать все неправильные действия любого пользователя, имеющего дело с вашей формой.

Давайте взглянем на код текущей версии нашего сценария `sendemail.php`, чтобы понять, как у Элмера получается пустое электронное письмо.

Наш сценарий sendemail.php пытается использовать текст, имеющийся в форме, для того, чтобы создать электронное письмо, даже если пользователь ничего не ввел в нее.

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    or die('Ошибка соединения с MySQL-сервером.');

  $query = "SELECT * FROM email_list";
  $result = mysqli_query($dbc, $query)
    or die('Ошибка при выполнении запроса к базе данных.');

  while ($row = mysqli_fetch_array($result)){
    $to = $row['email'];
    $first_name = $row['first_name'];
    $last_name = $row['last_name'];
    $msg = "Уважаемый $fist_name $last_name,\n$text";
    mail($to, $subject, $msg, 'From:' . $from);
    echo 'Электронное письмо отправлено: ' . _$to_ . '<br />';
  }

  mysqli_close($dbc);
?>
```

Текст полей формы извлекается из переменных $_POST['subject'] и $_POST['elvismail'] и сохраняется в переменных $subject и $text соответственно...

Проблема заключается в том, что мы используем переменную $text независимо от того, содержит она какой-либо текст или нет...

...точно так же мы используем и переменную $subject — независимо от того, содержит она какой-либо текст или нет.

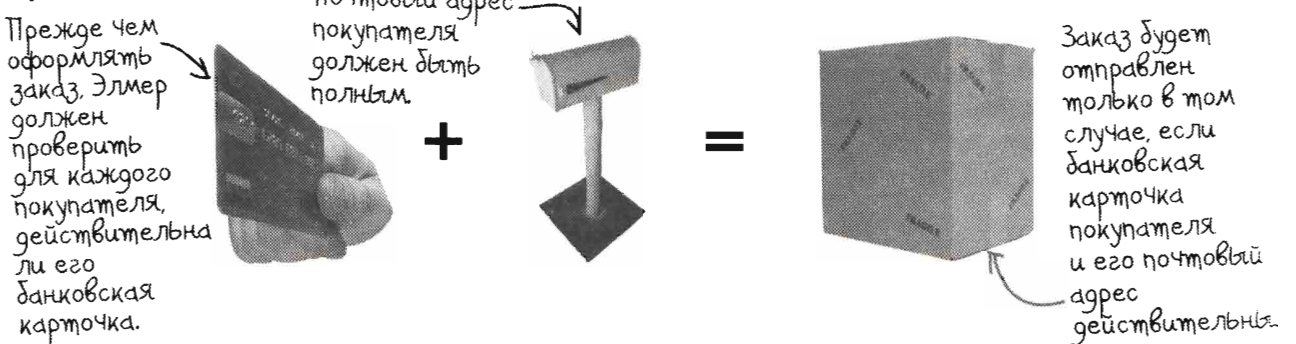
Напишите, какие, по вашему мнению, нужно внести изменения в код сценария sendemail.php, чтобы решить проблему пустых электронных писем:

Требуйте достоверных данных, вводимых в форму

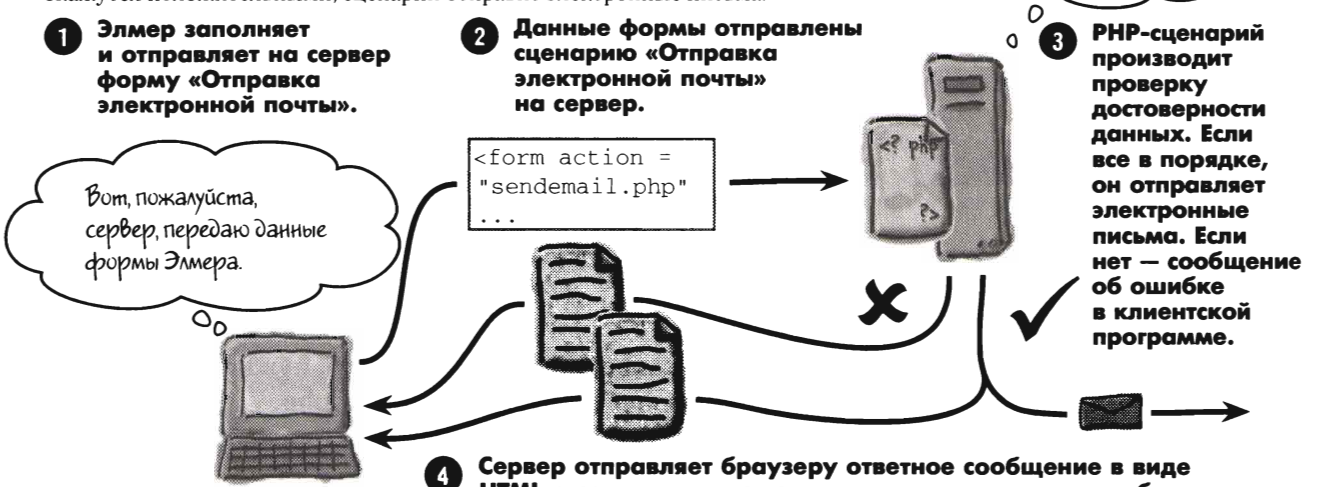
Контроль достоверности проводится, чтобы убедиться в том, что данные, которые вы получаете, — это те данные, которые вы ожидаете.

Форме Элмера «Отправка электронной почты» необходим **контроль достоверности** вводимых в форму данных. В результате такого контроля должно быть установлено, что данные в форме отвечают требованиям, предъявляемым к ним, и могут быть использованы для дальнейших операций. Элмер уже использует контроль достоверности, хотя прямо так его и не называет. Каждый раз, когда он получает заказ на отправку товаров, связанных с жизнью Элвиса, он не выполняет заказ немедленно... Сначала он проверяет достоверность каждого заказа!

В каждом случае вначале он проверяет, **действительна** ли банковская карточка покупателя, оформившего заказ. Если с карточкой все в порядке, он оформляет заказ и готовит его к отправке. После этого он должен обязательно проверить, все ли в порядке с почтовым адресом покупателя. При отсутствии каких-либо проблем с адресом Элмер отправляет заказ. Успешное выполнение заказа для магазина Элмера тесно связано с контролем достоверности всех данных, приведенных в этом заказе.



Для того чтобы решить проблему Элмера с электронными письмами, нам необходимо проводить контроль достоверности данных, передаваемых сценарию `sendemail.php`. Это означает, что все данные, отправленные с веб-формой на сервер сценарию `sendemail.php`, должны быть проверены в нем на достоверность и полноту, прежде чем будут использованы. Мы можем добавить в сценарий `sendemail.php` код, который проведет такую проверку. И только в том случае, если результаты проверки окажутся положительными, сценарий отправит электронные письма.



Логика в проверке на достоверность данных формы «Отправка электронной почты»

Элмеру необходимо **проконтролировать достоверность** данных, полученных от формы `sendemail.html`, прежде чем отправить любое электронное письмо. Фактически отправка электронных писем должна полностью полагаться на проверку достоверности данных. Чего мы в действительности хотим от PHP, так это принятия решения, основанного на достоверности данных, полученных сценарием `sendemail.php`. Нам необходим код с таким алгоритмом: «**Если** данные **достоверны**, отправляйте электронные письма».

Чтобы данные считались достоверными, необходимо, чтобы соблюдались два таких условия.

ЕСЛИ в поле формы «Тема» содержится текст
И в поле формы «Содержание» содержится текст,
ТОГДА отправить электронное письмо

Если соблюдаются оба эти условия, все отлично и мы можем отправлять электронные письма.

Мы посылали электронные письма, нисколько не беспокоясь о том, что именно введено в эти поля, да и вообще введено ли туда хоть что-нибудь.

С помощью проверки на достоверность мы можем быть уверенными, что ни одно письмо не будет отправлено, если для него нет данных хотя бы в одном из этих полей ввода.

Не бывает глупых вопросов

В: Я слышал также о том, что существуют способы проверки данных на достоверность на клиенте, а не на сервере. Как это происходит?

О: Клиентской программой в данном случае является браузер. Поэтому проверкой данных на достоверность на клиентской стороне будет любая проверка, осуществляемая до отправки данных на сервер PHP-сценарию. Такой язык, как JavaScript, может проводить проверку данных на достоверность на клиентской стороне. Если вы заинтересованы в более глубоком изучении этого вопроса, обратитесь к книге Head First JavaScript.

В: Зачем вообще осуществлять проверку данных на достоверность на сервере вместо того, чтобы делать все на клиенте?

О: При проверке данных на достоверность на клиентской стороне мы решаем только часть проблемы. В принципе, Элмер может открыть `sendemail.php` непосредственно в браузере и отправить пустые электронные письма. Но когда вы осуществляете проверку данных на достоверность на сервере, это решает обе проблемы. Отсутствие данных будет определено как в случае передачи их с формой, так и в случае загрузки непосредственно PHP-сценария. Нельзя сказать, что осуществлять проверку данных на достоверность на клиентской стороне неправильно. Фактически это очень хорошая практика. Но нужно всегда помнить, что сервер — это последняя линия обороны в перехвате недостоверных данных, поэтому проверка данных на достоверность на серверной стороне не должна игнорироваться.

Ваш код может принимать решения, используя управляющую конструкцию if

Управляющая конструкция PHP `if` (если) позволяет вашему коду принимать решение **на основании проверки истинности некоторого выражения**. Рассмотрим опять заказы Элмера. Прежде чем оформить заказ, Элмер планирует получить деньги в размере стоимости этого заказа, что означает, что он должен снять необходимую сумму с банковской карточки покупателя. Если покупатель даст Элмеру неправильный номер карточки, он не сможет оформить заказ. Поэтому в реальной жизни Элмер проводит проверку каждого заказа по такому сценарию:

Если оплата по банковской карточке покупателя происходит без проблем, он оформляет заказ.

Мы можем перевести этот сценарий в PHP-код, используя управляющую конструкцию `if`, которая служит для принятия решений подобного типа.

Основа управляющей конструкции if состоит из трех частей:

1. **Ключевое слово `if`**

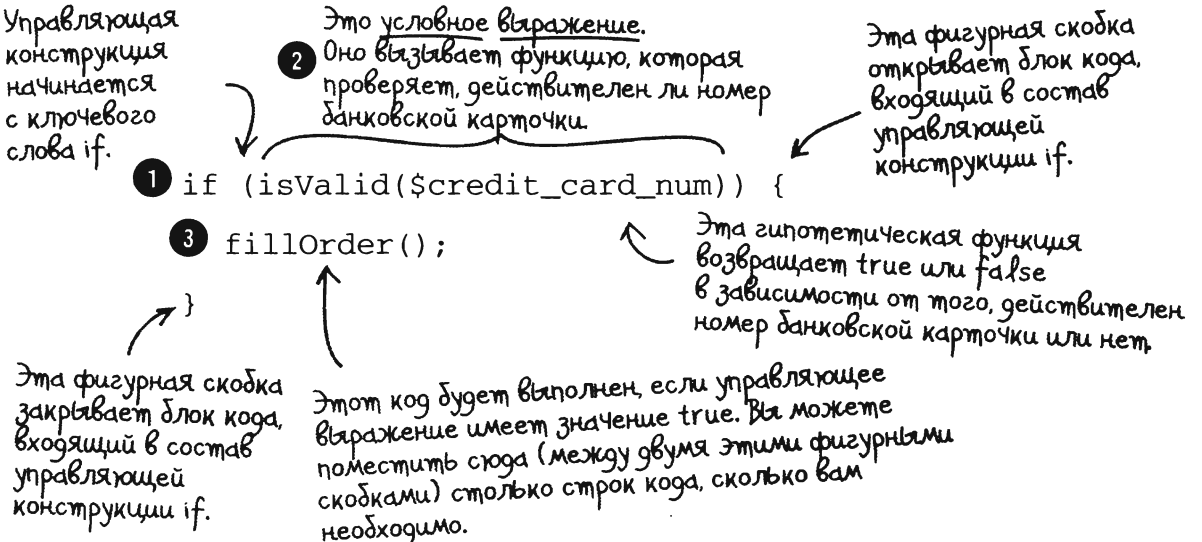
   Начало управляющей конструкции.

2. **Условное выражение**

   Условное выражение располагается в круглых скобках сразу за ключевым словом `if`. Здесь находится утверждение, истинность которого нам предстоит проверить.

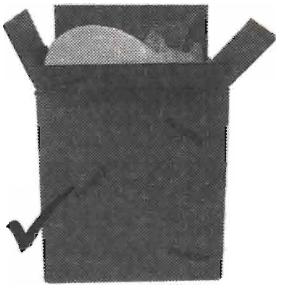
3. **Блок кода**

   Блок кода следует непосредственно за условным выражением и заключается в фигурные скобки. Он выполняется только в том случае, если условное выражение имеет значение `true` (истина).



Проверка на достоверность

Центральной частью управляющей конструкции `if` является ее условное выражение, которое всегда интерпретируется только как `true` или `false`. Условное выражение может быть переменной, вызовом функции или результатом сравнения какого-либо объекта с другим объектом. Элмер использует вызов функции, которая производит проверку достоверности номера банковской карточки и возвращает либо `true`, либо `false` в зависимости от результатов этой проверки.



Условное выражение принимает значение либо true, либо false.

ЕСЛИ номер банковской карточки действителен

ТОГДА оформить заказ

```
if (isValid($credit_card_num)) {
  fillOrder();
}
```

Если условное выражение принимает значение true, тогда выполняется блок кода, ограниченный фигурными скобками.

Достаточно распространенной практикой является использование в качестве условного выражения сравнения. Обычно оно сводится к сравнению значения переменной с какой-либо величиной. Например, Элмер хочет предоставить скидку тем своим покупателям, которые живут в Неваде. Он может создать управляющую конструкцию, которая будет проверять часть почтового адреса покупателя таким образом:

Этот оператор возвращает true, если значение переменной $shipping_state тождественно текстовой строке 'Невада'.

ЕСЛИ покупатель живет в Неваде

ТОГДА предоставить скидку

```
if ($shipping_state == 'Невада') {
  $total = $total * 0.9;
}
```

Если условное выражение принимает значение true, тогда выполняется блок кода по предоставлению 10%-й скидки.

Это условное выражение осуществляет проверку на тождество, которое производится оператором == (два знака равенства, следующие друг за другом без пробела). Оператор тождества можно использовать не только для сравнения переменных с текстовыми значениями. Вы можете сравнивать переменные с числами, переменные с переменными и даже с результатами вычислений.

Не заключайте числа в кавычки.

```
($num_items == 10)
```

Вы можете проверить, тождественно ли значение одной переменной значению другой.

```
($shipping_address == $billing_address)
(2 + 2 == 4)
```

Вы можете использовать арифметические операторы в условном выражении.

В управляющей конструкции if может осуществляться проверка не только на тождественность данных

Вполне допустимо записывать всю управляющую конструкцию полностью в одной строке

В управляющей конструкции `if` может осуществляться проверка данных более, чем просто на их тождественность. Условное выражение в вашей управляющей конструкции `if` может также проверить, не превышает ли заданное вами значение какой-либо величины. Если превышает, то результат проверки будет `true`, что приведет к выполнению блока кода, заключенного в фигурные скобки. Ниже приведено еще несколько видов проверки, которые могут применяться в условных выражениях управляющей конструкции `if`.

Начните с этих двух переменных.

```
$small_number = 2;
$big_number = 98065;
```

Результат обеих этих проверок — истина.

Имеется два оператора для проверки **нетождественности** двух величин: `<>` и `!=`. Результат такой проверки будет противоположным результату проверки на тождественность (`==`).

```
if ($small_number <> $big_number) { echo 'True'; }
if ($small_number != $big_number) { echo 'True'; }
```

Символ **«больше, чем»** (`>`) используется в качестве оператора, выполняющего проверку, больше ли значение величины объекта, расположенного слева от него, значения величины объекта, расположенного справа.

Результат этой проверки — ложь.

```
if ($small_number > $big_number) { echo 'True'; }
```

Символ **«меньше, чем»** (`<`) используется в качестве оператора, выполняющего проверку, меньше ли значение величины объекта, расположенного слева от него, значения величины объекта, расположенного справа.

Результат этой проверки — истина.

```
if ($small_number < $big_number) { echo 'True'; }
```

Следующие друг за другом без пробела символы **«больше, чем»** и **«равно»** (`>=`) используются в качестве оператора, выполняющего проверку, аналогичную проверке, которую проводит оператор, выраженный символом «больше, чем» (`>`), за исключением того, что результат будет `true` также и в том случае, если два проверяемых объекта равны между собой.

Результат этой проверки — ложь.

```
if ($small_number >= $big_number) { echo 'True'; }
```

Следующие друг за другом без пробела символы **«меньше, чем»** и **«равно»** (`<=`) используются в качестве оператора, выполняющего проверку, аналогичную проверке, которую проводит оператор, выраженный символом «меньше, чем» (`<`), за исключением того, что результат будет `true` также и в том случае, если два проверяемых объекта равны между собой.

Результат этой проверки — истина.

```
if ($small_number <= $big_number) { echo 'True'; }
```

А как будет обстоять дело с переменными, содержащими текст? Можем ли мы составить вот такое условное выражение: ("dog" > "cat")?

Да, вы можете сравнивать строки в условном выражении управляющей конструкции if.

Такое сравнение основывается на интерпретации алфавитных символов, при которой символ a считается меньше, чем символ z. Использование операторов «больше, чем» и «меньше, чем» может помочь вам, например, если возникнет необходимость расположить информацию в алфавитном порядке.

Станьте условным выражением в управляющей конструкции if

Ваша задача — играть роль условного выражения в управляющей конструкции if и решать, истина вы или ложь, в каждом из случаев, когда вам будут предлагаться для проверки следующие переменные:

```
$my_name = 'Бастер';
$a_number = 3;
$a_decimal = 4.6;
$favorite_song = 'Беспокойство';
$another_number = 0;
$your_name = $my_name;
```

`($a_number == 3)`	**true или false**
`($another_number == "")`	**true или false**
`($favorite_song == "Беспокойство")`	**true или false**
`($my_name == '$your_name')`	**true или false**
`($my_name == "$your_name")`	**true или false**
`($your_name == $my_name)`	**true или false**
`($favorite_song == 'Беспокойство')`	**true или false**
`($a_number > 9)`	**true или false**
`($favorite_food = 'hamburger')`	**true или false**

Станьте условным выражением в управляющей конструкции if

Ваша задача — играть роль условного выражения в управляющей конструкции if и решать, истина вы или ложь, в каждом из случаев, когда вам будут предлагаться для проверки следующие переменные:

```
$my_name = 'Бастер';
$a_number = 3;
$a_decimal = 4.6;
$favorite_song = 'Беспокойство';
$another_number = 0;
$your_name = $my_name;
```

0 (ноль) и пустая строка интерпретируются как тождественные друг другу.

```
($a_number == 3)                          true или false
($another_number == "")                   true или false
($favorite_song == "Беспокойство")        true или false
($my_name == '$your_name')                true или false
($my_name == "$your_name")                true или false
($your_name == $my_name)                  true или false
($favorite_song == 'Беспокойство')        true или false
($a_number > 9)                           true или false
($favorite_food = 'hamburger')            true или false
```

Раз мы заключили слово $your_name в одинарные кавычки, значит это слово и есть значение объекта, расположенного справа от оператора ==. Значение же переменной $my_name, расположенной слева от оператора ==, — «Бастер». Мы сравниваем разные значения, следовательно, результат сравнения — ложь.

Значение переменной $a_number — 3, не больше 9.

Это пример с подвохом. Так как здесь использован только один знак равенства, значит это оператор присваивания (=), а не сравнения (тождества, ==). А результат присваивания — всегда истина, так как все, кроме 0, NULL или false, интерпретируется PHP как true.

Если вы хотите определить, тождественны ли эти два объекта друг другу, здесь должен быть оператор тождества (==), а не присваивания (=).

Не бывает глупых вопросов

В: Является ли условное выражение управляющей конструкции if тем же самым, что и условное выражение в цикле while?

О: Это совершенно одно и то же. И хотя в главе 3 мы использовали его только для того, чтобы узнать, есть ли еще записи в результате запроса, мы можем разработать более интересные условные выражения для цикла `while` путем использования сравнений различного вида. Вы увидите это позднее.

Логика в проверке на достоверность данных формы «Отправка электронной почты»

Элмеру необходимо проконтролировать достоверность данных, полученных от формы `sendemail.html`, прежде чем отправить любое электронное письмо. Фактически отправка электронных писем должна полностью полагаться на проверку достоверности данных. Чего мы в действительности хотим от PHP, так это принятия решения, основанного на достоверности данных, полученных сценарием `sendemail.php`. Нам необходим код с таким алгоритмом: «Если данные достоверны, отправляйте электронные письма».

Но вначале мы должны извлечь данные из формы и присвоить их значения паре переменных.

```
$subject = $_POST['subject'];

$text = $_POST['elvismail'];
```

sendemail.html

Нам необходимо проверить, содержатся ли данные в каждом из этих полей ввода формы. Логика такой проверки должна выглядеть примерно так:

ЕСЛИ в поле формы «Тема» содержится текст
И в поле формы «Содержание» содержится текст,
ТОГДА отправить электронное письмо

Или можем попробовать противоположный способ и проверить, не являются ли оба эти поля ввода формы пустыми. В таком случае мы можем вывести пользователю предупреждение об ошибке:

ЕСЛИ в поле формы «Тема» не содержится текста
И в поле формы «Содержание» не содержится текста,
ТОГДА вывести сообщение об ошибке

Проблема этих двух примеров заключается в том, что их логика требует от нас проводить два сравнения в одной управляющей конструкции `if`. Одно из возможных решений — использовать две управляющие конструкции `if`.

Время поработать

Напишите две управляющие конструкции `if`, в которых проверяется, являются ли оба поля ввода формы «Тема электронного письма» и «Содержание электронного письма» пустыми. Выведите с помощью команды PHP `echo` сообщение об ошибке.

Решение задачи

Напишите две управляющих конструкции if, в которых проверяется, являются ли оба поля ввода формы «Тема электронного письма» и «Содержание электронного письма» пустыми. Выведите с помощью команды PHP echo сообщение об ошибке.

Две следующие подряд одинарные кавычки представляют пустую строку.

```
if ($subject == '') {
  if ($text == '') {
    echo 'Вы забыли указать тему и содержание электронного письма.<br />';
  }
}
```

Помещая вторую управляющую конструкцию if внутри первой (внутри ее фигурных скобок), мы фактически формируем код, говорящий о том, что условные выражения обеих управляющих конструкций if должны быть true, чтобы была выполнена команда echo.

Смещение текста помогает понять, где заканчивается внутренняя управляющая конструкция if, а где — внешняя.

PHP-функции для проверки переменных

Использование оператора == для проверки пустых строк работает, но существует более удобный метод, основанный на использовании встроенных PHP-функций. Функция `isset()` проверяет, существует ли переменная, то есть присвоено ли ей какое-либо значение. Функция `empty()` идет на шаг дальше и определяет, присвоено ли переменной пустое значение, что определяется в PHP как 0 (ноль), пустая строка (`''` или `""`), или значения `false` или `NULL`.

Давайте посмотрим, как эти функции работают:

Переменной $v1 присвоено значение.

Обе переменные $v1 и $v2 существуют, хотя только переменной $v1 присвоено конкретное значение.

Значение переменной $v2 — пустая строка.

Исполняется только код, выделенный серым цветом.

Значение переменной $v2 — не пустая строка, она содержит текст. Поэтому это условное выражение имеет значение false.

```
$v1 = 'aloha';
$v2 = '';
if (isset($v1)) {echo 'Переменная $v1 существует<br />';}
if (empty($v1)) {echo 'Переменная $v1 содержит пустое значение<br />';}
if (isset($v2)) {echo 'Переменная $v2 существует<br />';}
if (empty($v2)) {echo 'Переменная $v2 содержит пустое значение<br />';}
if (isset($v3)) {echo 'Переменная $v3 существует<br />';}
if (empty($v3)) {echo 'Переменная $v3 содержит пустое значение<br />';}
```

Переменная $v2 существует, хотя и имеет значение пустой строки.

Переменная $v3 не существует.

Переменная $v2 пустая, так как содержит пустую строку.

Переменная $v3 считается пустой, хотя она и не существует.

Я поняла. Мы можем использовать функции isset() и empty() для того, чтобы проверить достоверность данных полей «Тема электронного письма» и «Содержание электронного письма».

Это верно, но только отчасти. В действительности нам необходимо только убедиться в том, что эти поля формы не пусты, а для такой проверки функции empty() вполне достаточно.

Переменным `$subject` и `$text` присваиваются значения элементов `$_POST['subject']` и `$_POST['elvismail']` суперглобального массива `$_POST`. Если вы будете проверять их с помощью функции `isset()`, она всегда будет возвращать вам значение `true` независимо от того, содержат они какие-либо полезные данные или нет. Иначе говоря, функция `isset()` не покажет вам разницы между пустым полем ввода и полем, в которое введены какие-либо данные. Функция `empty()` проверяет, действительно ли поле ввода пусто или нет, что как раз нам и нужно для контроля достоверности.

Функция isset() проверяет, существует ли переменная и присвоено ли ей какое-либо значение.
Функция empty() проверяет, содержит ли переменная какие-либо непустые данные.

Не бывает глупых вопросов

В: Зачем вообще тогда нужна функция isset()?

О: Функция `isset()` крайне необходима, когда вам нужно узнать, **существуют** ли интересующие вас данные. Например, вы можете проверить, переданы ли вам данные формы в результате запроса `POST`, отправляя функции `isset()` в качестве аргумента переменную `$_POST`. Как вы убедитесь позднее в данной главе, это оказывается исключительно удобным на практике.

Время поработать

Перепишите две управляющих конструкции `if`, в которых проверяется, являются ли оба поля ввода формы «Тема электронного письма» и «Содержание электронного письма» пустыми. Но на этот раз используйте в условном выражении функцию `empty()` вместо оператора тождества (`==`).

Решение задачи

Перепишите две управляющие конструкции `if`, в которых проверяется, являются ли оба поля ввода формы «Тема электронного письма» и «Содержание электронного письма» пустыми. Но на этот раз используйте в условном выражении функцию `empty()` вместо оператора тождества (==).

Оператор тождества (==) в обоих условных выражениях заменяется на вызов функции empty().

```
if (empty($subject)) {
  if (empty($text)) {
    echo 'Вы забыли указать тему и содержание электронного
письма.<br />';
  }
}
```

Остальной код не изменился.

А вдруг мы должны выполнить некоторые действия, если поле ввода формы **не** пустое? Существует ли функция notemptyO?

Нет, зато существует простой способ реверсировать логику любого условного выражения... оператор отрицания.

Мы знаем, что условное выражение, которое управляет конструкцией `if`, может иметь значение либо `true` либо `false`. Но как быть, если логика диктует нам проверку значения, обратного тому, которое дает условное выражение? Например, было бы полезно знать, что поля ввода формы Элмера не пусты, перед тем как отправлять огромное количество электронных писем с данными формы. Проблема в том, что никакой функции `notempty()` не существует. Решение заключается в использовании оператора отрицания (!), который превращает `true` в `false`, а `false` в `true`. Поэтому выражение `!empty()` вызывает функцию `empty()`, а затем меняет результат, который она возвращает, на противоположный.

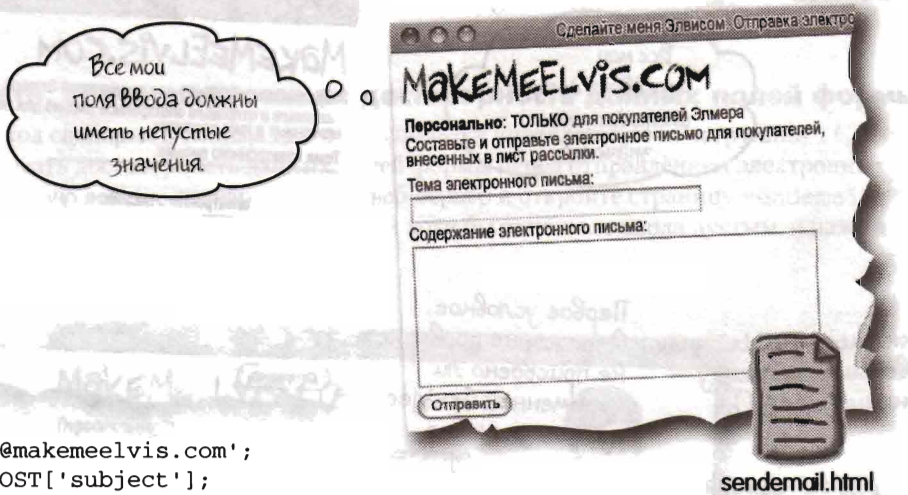
Оператор отрицания (!) изменяет true на false, а false на true.

```
if (!empty($subject)) {

}
```

Это условное выражение спрашивает: «Поле ввода формы 'Тема электронного письма' не пустое?» или «Содержит ли это поле какие-либо данные?»

УПРАЖНЕНИЕ

Заполните пустые строки в коде сценария `sendemail.php` так, чтобы электронные письма **отправлялись только** тогда, когда обеим переменным `$subject` и `$text` присвоены непустые значения. Используйте управляющую конструкцию `if` и функцию `empty()`.




```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  if .....

    if .....

      $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
        or die('Ошибка соединения с MySQL-сервером.');

      $query = "SELECT * FROM email_list";
      $result = mysqli_query($dbc, $query)
        or die('Ошибка при выполнении запроса к базе данных.');


      while ($row = mysqli_fetch_array($result)) {
        $to = $row['email'];
        $first_name = $row['first_name'];
        $last_name = $row['last_name'];
        $msg = "Уважаемый $first_name $last_name,\n$text";
        mail($to, $subject, $msg, 'From:' . $from);
        echo 'Электронное письмо отправлено: ' . $to . '<br />';
      }
      mysqli_close($dbc);



?>
```

Заполните пустые строки в коде сценария `sendemail.php` так, чтобы электронные письма отправлялись только тогда, когда обеим переменным `$subject` и `$text` присвоены непустые значения. Используйте управляющую конструкцию `if` и функцию `empty()`.

Все мои поля ввода должны иметь непустые значения.

sendemail.html

Восклицательный знак реверсирует логику функции empty().

Первое условное выражение проверяет, не присвоено ли переменной $subject пустое значение...

...если нет — отлично! А сейчас проверим, не присвоено ли переменной $text пустое значение.

Мы должны записать одну управляющую конструкцию if в блок кода другой, чтобы осуществить такую проверку. Это называется вложением.

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  if (!empty($subject)) {
    if (!empty($text)) {
      $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
        or die('Ошибка соединения с MySQL-сервером.');

      $query = "SELECT * FROM email_list";
      $result = mysqli_query($dbc, $query)
        or die('Ошибка при выполнении запроса к базе данных.');

      while ($row = mysqli_fetch_array($result)) {
        $to = $row['email'];
        $first_name = $row['first_name'];
        $last_name = $row['last_name'];
        $msg = "Уважаемый $first_name $last_name,\n$text";
        mail($to, $subject, $msg, 'From:' . $from);
        echo 'Электронное письмо отправлено: ' . $to . '<br />';
      }
      mysqli_close($dbc);
    }
  }
?>
```

Если хотя бы одна из переменных содержит пустое значение, условное выражение с ее участием примет значение false и блок кода, заключенный в фигурные скобки, не будет выполнен. Это означает, что электронное письмо с пустыми темой или содержанием отправлено не будет — как раз то, что нам необходимо!

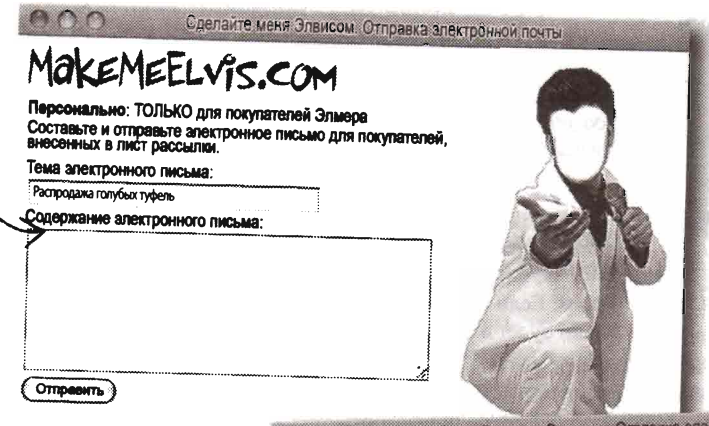
Нам необходимо закрыть оба блока кода каждой из управляющих конструкций if. Первая фигурная скобка закрывает блок кода внутренней управляющей конструкции if, вторая — внешней.

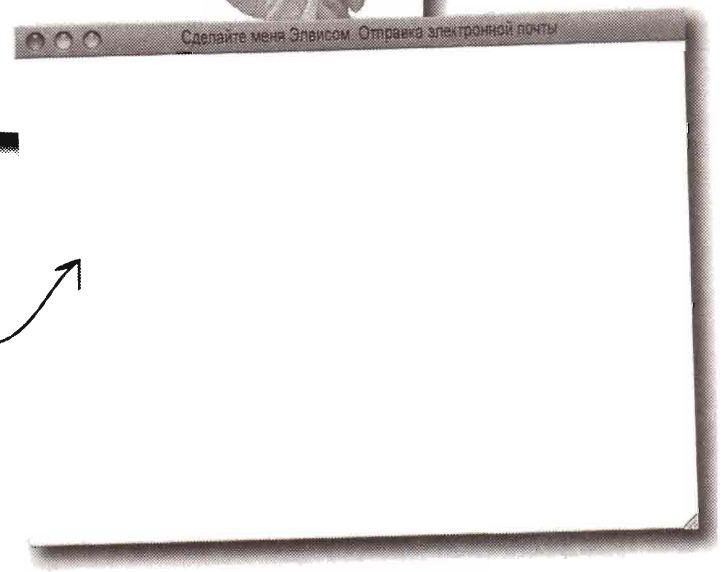
Тест-драйв

Проверьте, контролируется ли достоверность данных полей формы.

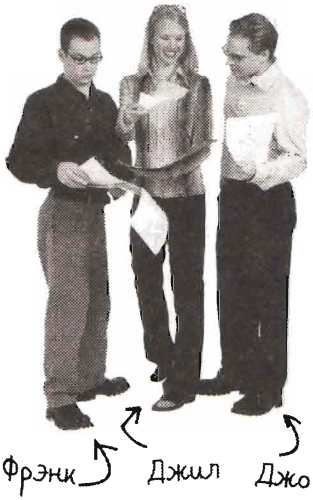
Измените код сценария `sendemail.php`, используя управляющие конструкции `if`, чтобы контролировать достоверность данных полей формы перед отправлением электронных писем. Загрузите новую версию сценария на ваш веб-сервер и откройте страницу `sendemail.hnml` в браузере. Убедитесь в том, что вы оставили хотя бы одно поле ввода пустым, и нажмите кнопку «Отправить».

Поле ввода содержания электронного письма пустое. Это означает, что данные формы не пройдут процедуру проверки на достоверность.

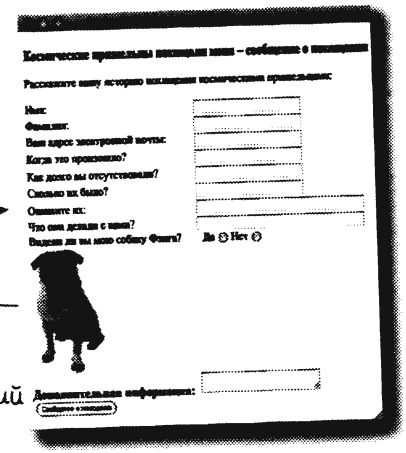




Только пустая страница вместо подтверждения того факта, что электронное письмо отправлено не было. Но какое-либо сообщение об ошибке было бы полезнее пустой страницы.



А что если в нашей форме будет много полей ввода? Мы должны будем вкладывать множество управляющих конструкций if одна в другую, чтобы проверить достоверность их всех?



Форма Оуэна, которую мы рассматривали ранее в этой книге, — пример того, как множество полей ввода приведет к запутанному множеству управляющих конструкций if, вложенных одна в другую.

```
if (!empty($first_name)) {
  if (!empty($last_name)) {
    if (!empty($when_it_happened)) {
      if (!empty($how_long)) {
        if (!empty($how_many)) {
          ...
```

Такое большое количество вложенных управляющих конструкций if затрудняет отслеживание соответствия открывающих и закрывающих фигурных скобок.

Джо: Я думаю, ты права. Если мы хотим убедиться, что все эти поля ввода данных не пусты, мы должны вложить управляющую конструкцию `if` для каждого поля.

Фрэнк: Если мы сместим каждую строку кода для каждой управляющей конструкции `if`, все ли будет в порядке?

Джил: Технически — да. Я хочу сказать, что код определенно будет работать независимо от того, сколько управляющих конструкций `if` мы вложим одна в другую, но меня беспокоит то, насколько это затруднит чтение и понимание кода с такими многократными вложениями. Простой поиск соответствия открывающих и закрывающих фигурных скобок может оказаться серьезной проблемой.

Фрэнк: Это точно. Я думаю, даже простое смещение кода может серьезно усложнить дело... Представьте себе, что десять полей ввода данных повлекут за собой десять управляющих конструкций `if`, вложенных одна в другую, с десятью уровнями смещения кода. Даже если мы будем смещать каждую конструкцию `if` всего на два пробела, это приведет к смещению блока кода в ней на 20 пробелов. Кошмар!

Джо: Но если мы будем смещать код, используя символ табуляции вместо символов пробела, это сократит количество символов вдвое: 10 символов табуляции против 20 пробелов. Не так уж и плохо.

Джил: Ребята! В конце концов проблема заключатся не в том, как смещать строки кода при вложении друг в друга множества управляющих конструкций `if`. Просто такое множественное вложение управляющих конструкций `if` само по себе не слишком хороший стиль программирования. Рассматривайте это в таком аспекте, что в действительности мы говорим об одном логическом условном выражении, а именно: «Все ли наши поля ввода данных формы не пусты?». Проблема состоит в том, что условное выражение включает проверку десяти различных данных, вынуждая нас разбить весь процесс на десять управляющих конструкций `if`.

Фрэнк: Я понял. Все, что нам необходимо, — это найти способ проверки всех десяти данных формы в одной управляющей конструкции `if`, так?

Джил: Точно.

Джо: Тогда мы смогли бы написать одно большое условное выражение, которое проверяло бы все данные формы за раз. Отлично!

Джил: Да, но мы пока что не разгадали головоломку, которая позволит нам объединить множество проверок данных в одно условное выражение.

Множественная проверка на достоверность с помощью логических операторов AND и OR

Вы можете создать условное выражение для управляющей конструкции `if` с проверкой на достоверность множества данных, объединяя их с помощью логических операторов. Давайте посмотрим на то, как это действует в случае двух уже знакомых нам проверок на достоверность: `!empty($subject)` и `!empty($text)`. В первом примере два этих выражения объединяются с помощью логического оператора AND, код которого выглядит как &&.

Логические операторы дают возможность более элегантно создавать управляющие конструкции if.

Логический оператор AND.

Дополнительные круглые скобки дают понять, что оператор отрицания относится только к функции empty().

```
if ((!empty($subject)) && (!empty($text))) {
```

Это условное выражение тогда и только тогда будет иметь значение true, когда обе переменных $subject И $text имеют непустые значения.

Логический оператор AND (и) кодируется как && а логический оператор OR (или) — как ||.

Логический оператор AND рассматривает две величины, которые могут принимать значения `true` или `false`, и возвращает `true` тогда и только тогда, когда обе из рассматриваемых величин имеют значения `true`; во всех остальных случаях он возвращает `false`. Следовательно, оба поля ввода данных формы должны иметь непустые значения для того, чтобы условное выражение приняло значение `true` и был выполнен блок кода управляющей конструкции `if`.

Логический оператор OR, код которого выглядит как ||, также рассматривает две величины, которые могут принимать значения `true` или `false` , и возвращает `true` тогда, когда любая из рассматриваемых величин имеет значение `true`, и только в случае, если обе они имеют значение `false`, он возвращает `false`.

```
if ((!empty($subject)) || (!empty($text))) {
```

Это не число одиннадцать, это две вертикальные черты, следующие одна за другой. Символ вертикальной черты выводится нажатием клавиши обратной наклонной черты при нажатой клавише Shift.

Это условное выражение будет иметь значение true тогда, когда любая из переменных $subject ИЛИ $text имеет непустое значение.

Таким образом, блок кода этой управляющей конструкции `if` будет выполнен тогда, когда любое из полей ввода будет иметь непустое значение. Можно привести более интересный пример, когда вы хотите, чтобы блок кода исполнялся при условии, что в одном поле ввода имеются непустые данные, а в другом — нет. Например:

```
if (empty($subject) && (!empty($text))) {
```

Поле ввода формы «Тема электронного письма» должно быть пустым, а поле формы «Содержание электронного письма» — непустым, чтобы все условное выражение имело значение true.

Так как в этом условном выражении используется логический оператор AND, то для того, чтобы был выполнен блок кода управляющей конструкции `if`, оба составных условных выражения, расположенных как слева, так и справа от оператора AND, должны иметь значение `true`. В данном случае это означает, что поле формы «Тема электронного письма» должно быть пустым, а поле формы «Содержание электронного письма» — непустым. Вы можете реверсировать подобную проверку перемещением оператора отрицания (!) от правой функции `empty()` к левой.

```
if ((!empty($subject)) && empty($text)) {
```

Это условное выражение будет иметь значение true только в том случае, если поле ввода формы «Тема электронного письма» будет непустым, а поле формы «Содержание электронного письма» — пустым.

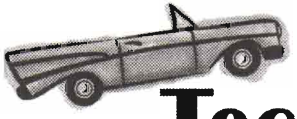
Логические операторы AND (&&) и OR (||) позволяют создавать очень мощные условные выражения, реализация логики которых без их использования потребовала бы дополнительных, часто очень запутанных управляющих конструкции `if`.

Перепишите выделенные серым цветом фрагменты кода сценария `sendemail.php` **так, чтобы в них использовались логические операторы вместо вложенных управляющих конструкций** `if`.

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  if (!empty($subject)) {
    if (!empty($text)) {

  ..........

  ..........

      $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
        or die('Ошибка соединения с MySQL-сервером.');

      $query = "SELECT * FROM email_list";
      $result = mysqli_query($dbc, $query)
        or die('Ошибка при выполнении запроса к базе данных.');


      while ($row = mysqli_fetch_array($result)) {
        $to = $row['email'];
        $first_name = $row['first_name'];
        $last_name = $row['last_name'];
        $msg = "Уважаемый $first_name $last_name,\n$text";
        mail($to, $subject, $msg, 'From:' . $from);
        echo 'Электронное письмо отправлено: ' . $to . '<br />';
      }

      mysqli_close($dbc);
    }
  }


?>
```

Это наши вложенные управляющие конструкции if. Перепишите код с использованием логических операторов.

Эти фигурные скобки закрывают блоки кода двух управляющих конструкций if.



Тест-драйв

Убедитесь в том, что логические операторы в сценарии «Отправка электронной почты» работают так же, как и вложенные управляющие конструкции if.

Измените код сценария `sendemail.php` так, чтобы для контроля достоверности данных полей перед отправлением электронных писем использовать логические операторы в условном выражении единственной управляющей конструкции `if`. Перепроверяйте решение этого упражнения на следующей странице, пока не убедитесь, что ваши изменения работают.

Загрузите новую версию сценария на ваш веб-сервер и откройте страницу `sendemail.hnml` в браузере. Убедитесь в том, что вы оставили хотя бы одно поле ввода пустым, и нажмите кнопку «Отправить». Сценарий все еще не позволяет отправлять электронные письма, если какое-либо из полей ввода формы не заполнено?

Не бывает глупых вопросов

В: Имеет ли значение, в каком порядке записаны условные выражения, объединяемые логическими операторами && или || в управляющей конструкции if?

О: Да. Смысл здесь заключается в том, что эти два оператора заканчивают процесс проверки так скоро, как это только возможно. Это означает, что если результата проверки левого условного выражения достаточно для принятия правильного решения, то процесс проверки на этом прекращается и правое условное выражение просто игнорируется. Например, если при проверке двух условных выражений, объединенных логическим оператором &&, левое условное выражение имеет значение `false`, нет никакого смысла продолжать проверку, так как при любом значении правого условного выражения общее значение будет `false`, поэтому процесс проверки на этом прекращается, и правое условное выражение игнорируется. Аналогично принимается решение в случае, если левое условное выражение в операторе OR имеет значение `true`.

В: Я видел PHP-код, в котором используются логические операторы AND и OR вместо && и ||. Как они работают?

О: Логические операторы AND и OR — практически то же самое, что и логические операторы && и ||. Они немного отличаются друг от друга в приоритетах относительно других операторов, но если вы будете внимательными с использованием скобок при составлении своих условных выражений, то не увидите никакой разницы между ними.

Перепишите выделенные серым цветом фрагменты кода сценария `sendemail.php` так, чтобы в них использовались логические операторы вместо вложенных управляющих конструкций `if`.

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  if (!empty($subject)) {
    if (!empty($text)) {

  if (((!empty($subject)) && (!empty($text))) {
  .....

    $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
      or die('Ошибка соединения с MySQL-сервером.');

    $query = "SELECT * FROM email_list";
    $result = mysqli_query($dbc, $query)
      or die('Ошибка при выполнении запроса к базе данных.');;


    while ($row = mysqli_fetch_array($result)) {
      $to = $row['email'];
      $first_name = $row['first_name'];
      $last_name = $row['last_name'];
      $msg = "Уважаемый $fist_name $last_name,\n$text";
      mail($to, $subject, $msg, 'From:' . $from);
      echo 'Электронное письмо отправлено: ' . $to . '<br />';
    }

    mysqli_close($dbc);
  }
  }
  .....

?>
```

Оператор отрицания, или NOT (!), используется для проверки непустых полей ввода данных.

Мы можем использовать оператор AND для проверки обоих условных выражений в одной управляющей конструкции if.

Не забывайте: два символа &&, написанных без пробела, — это то, как мы кодируем логический оператор AND.

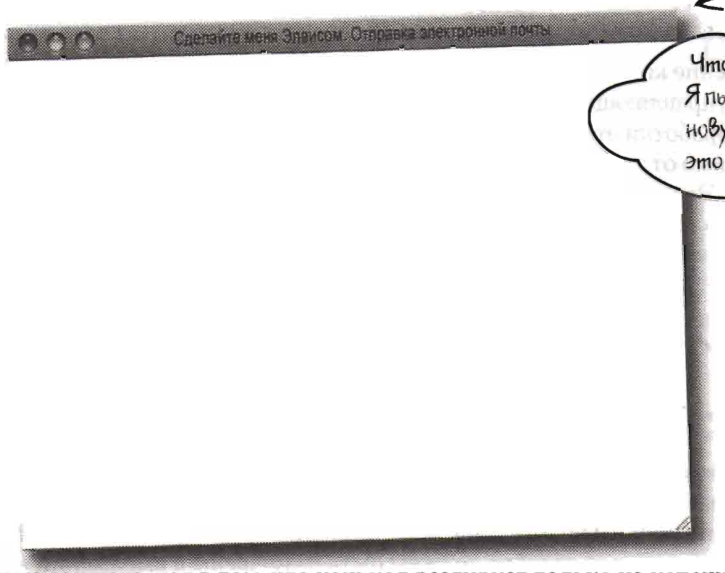
Весь код внутри фигурных скобок должен быть сдвинут влево на один уровень смещения, так как сейчас он расположен в одной управляющей конструкции if вместо двух, как это было прежде.

Так как сейчас у нас одна управляющая конструкция if, то и ее закрывающая фигурная скобка тоже одна.

Форме ввода данных необходима обратная связь

Код нашего сценария `sendemail.php` делает очень полезную работу, проводя проверку данных на достоверность, так что не будет отправлено ни одного электронного письма, если в полях «Тема электронного письма» либо «Содержание электронного письма» не введено никаких данных. Но когда проверка данных на достоверность не удается, сценарий ничего не сообщает Элмеру о причинах этого. Все, что он получает, — это пустая веб-страница.

Элмер видит эту пустую страницу после отправки формы на сервер... И он понятия не имеет, почему так происходит!



Проблема заключается в том, что наш код реагирует только на успешную проверку данных. Но если результат проверки окажется `false` (неверные данные), ничего не произойдет и Элмер останется в неведении по поводу того, было ли отправлено письмо, и если нет, то почему. Ниже приведен сокращенный код, объясняющий проблему пустой страницы:

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  if ((!empty($subject)) && (!empty($text))) {
    $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    ...
    mysqli_close($dbc);
  }
?>
```

Ничего не происходит, если управляющей конструкции if не удается выполнить блок кода. Именно по этой причине и генерируется пустая страница, когда какие-либо из необходимых данных пропущены.

Нам необходимо дать Элмеру понять, в чем заключается проблема, в идеале — сообщая ему, какие из полей ввода данных формы пусты, чтобы он мог попробовать ввести все необходимые данные опять.

Это не проблема. Нужно просто поместить команду echo сразу после закрывающей фигурной скобки управляющей конструкции if.

Это не будет работать, потому что код, помещенный непосредственно после закрывающей фигурной скобки управляющей конструкции if, будет выполняться всегда.

Размещение команды `echo` непосредственно после закрывающей фигурной скобки управляющей конструкции `if` означает, что она будет выполняться после обработки этой конструкции, но это будет происходить всегда, независимо от того, выполнялся ли блок кода управляющей конструкции `if` или нет. Это не то, что нам необходимо. Мы хотим, чтобы команда `echo` выводила сообщение об ошибке только тогда, когда условное выражение управляющей конструкции `if` принимает значение `false`.

ЕСЛИ в поле формы «Тема» содержится текст
И в поле формы «Содержание» содержится текст,
✓ **ТОГДА** отправить электронное письмо
✗ **ИНАЧЕ** вывести сообщение об ошибке

Управляющая конструкция `if` предлагает необязательное ключевое слово `else` (иначе), за которым следует код, выполняемый в случае, если условное выражение принимает значение `false`. Следовательно, наши команды на вывод сообщения об ошибке должны быть размещены в выражении `else`. В этом случае сообщение об ошибке будет выведено, только если одно из полей ввода данных формы окажется пустым. Просто напишите ключевое слово `else` непосредственно после закрывающей скобки управляющей конструкции `if` и затем альтернативный блок кода, ограничив его фигурными скобками.

Напишите ключевое слово else непосредственно после закрывающей скобки управляющей конструкции if.

```
if ((!empty($subject)) && (!empty($text))) {
    ...
}
else {
    echo 'Вы забыли ввести тему и/или содержание электронного
письма.<br />';
}
```

Здесь должен быть блок кода, в результате выполнения которого будет отправлено электронное письмо.

Так же как и в основном блоке кода управляющей конструкции if, блок кода, следующий за ключевым словом else, ограничивается фигурными скобками.

Этот блок кода будет выполняться только в том случае, если условное выражение примет значение false.

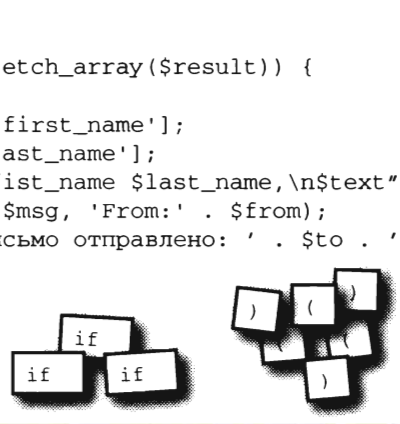
Код, следующий за ключевым словом else, выполняется тогда, когда условное выражение управляющей конструкции if принимает значение false.

УПРАЖНЕНИЕ

Ниже приведен новый код сценария Элмера `sendemail.php`, использующий управляющую конструкцию `if` с ключевым словом `else` для обеспечения обратной связи, но часть кода удалена. Используя магниты, поместите нужный код в нужное место.

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  .....

  }
  else {

  // Мы знаем, что ИЛИ переменная $subject, ИЛИ переменная $text пуста. Давайте узнаем, какая

        echo 'Вы забыли ввести тему электронного письма.<br />';
      }
      else {

        echo 'Вы забыли ввести содержание электронного письма.<br />';
      }
    }
    else {

      ...
      while ($row = mysqli_fetch_array($result)) {
        $to = $row['email'];
        $first_name = $row['first_name'];
        $last_name = $row['last_name'];
        $msg = "Уважаемый $fist_name $last_name,\n$text";
        mail($to, $subject, $msg, 'From:' . $from);
        echo 'Электронное письмо отправлено: ' . $to . '<br />';
      }

      mysqli_close($dbc);
    }
  }
?>
```

```
// Мы знаем, что и переменная $subject, И переменная $text пусты
// Переменная $subject пуста
// Все отлично, отправляйте электронное письмо
// Переменная $text пуста
```



РЕШЕНИЕ К УПРАЖНЕНИЮ

Ниже приведен новый код сценария Элмера `sendemail.php`, использующий управляющую конструкцию `if` с ключевым словом `else` для обеспечения обратной связи, но часть кода удалена. Используя магниты, поместите нужный код в нужное место.

Внешняя управляющая конструкция if проверяет пустое ли значение имеют обе переменные $subject и $text. Если нет, существует только три альтернативы: обе содержат текст, тема электронного письма пуста, содержание электронного письма пусто.

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];

  if ( empty($subject) && empty($text) ) {
    // Мы знаем, что и переменная $subject, И переменная $text пусты
  }
  else {
    if ( empty($subject) || empty($text) ) {
      // Мы знаем, что ИЛИ переменная $subject, ИЛИ переменная $text пуста. Давайте
узнаем, какая
      if ( empty($subject) ) {
        // Переменная $subject пуста
        echo 'Вы забыли ввести тему электронного письма.<br />';
      }
      else {
        // Переменная $text пуста
        echo 'Вы забыли ввести содержание электронного письма.<br />';
      }
    }
    else {
      // Все отлично, отправляйте электронное письмо
      while ($row = mysqli_fetch_array($result)) {
        $to = $row['email'];
        $first_name = $row['first_name'];
        $last_name = $row['last_name'];
        $msg = "Уважаемый $fist_name $last_name,\n$text";
        mail($to, $subject, $msg, 'From:' . $from);
        echo 'Электронное письмо отправлено: ' . $to . '<br />';
      }
      mysqli_close($dbc);
    }
  }
?>
```

К этому моменту после проверки всех иных возможностей мы точно знаем, что все поля данных формы заполнены.

Все эти вложенные управляющие конструкции if с ключевым словом else затрудняют чтение и понимание сценария. Я не хотел бы когда-либо иметь дело с подобными сценариями! Его необходимо упростить, пока кто-нибудь не вывихнул себе мозги.

Хороший стиль программирования выражается также и в том, что код должен быть как можно более простым для чтения и понимания. Особенно это касается случаев вложения различных конструкций одна в другую.

Слишком большое количество инвертированных условных выражений `else` во вложенных управляющих конструкциях `if` делают код трудным для чтения и понимания. Возможно, это не имело бы значения, если бы вам не приходилось возвращаться к нему позднее, но это маловероятно. Если вам когда-нибудь понадобится изменить форму ввода данных и добавить новое поле ввода, написание кода проверки достоверности может превратиться в достаточно сложную задачу, так как вам будет трудно понять свой прежний код и определить, в какую его часть необходимо внести соответствующие изменения.

Станьте чистильщиком кодом управляющей конструкции IF

Ваша задача — играть роль управляющей конструкции if и расчищать запутанные вложенные конструкции IF и ELSE. Перепишите код так, чтобы избавиться от вложенных конструкций, но убедитесь, что он работает правильно.

Совет: вероятно, вы можете вообще обойтись без else!

```
if (empty($subject) && empty($text)) {
  echo 'Вы забыли ввести тему и содержание электронного письма.<br />';
} else {
  if (empty($subject) || empty($text)) {
    if (empty($subject) {
      echo 'Вы забыли ввести тему электронного письма.<br />';
    } else {
      echo 'Вы забыли ввести содержание электронного письма.<br />';
    }
  } else {
    // Все отлично, отправляйте электронное письмо
  }
}
```

Перепишите этот код, чтобы в нем не было вложенных конструкций.

Тест-драйв

Испытайте упрощенный код управляющей конструкции if, чтобы убедиться в том, что он выполняется как положено.

Измените код сценария `sendemail.php` так, чтобы в нем использовались управляющие конструкции `if` подобно тому, как было написано ранее, но с упрощенными вложениями управляющих конструкций `if`. Обратитесь к следующей странице, если вы сомневаетесь в том, какие изменения необходимо внести.

Загрузите новую версию сценария на ваш веб-сервер и откройте страницу `sendemail.hnml` в браузере. Поэкспериментируйте со сценарием, отправляя форму на сервер с заполненными и пустыми полями ввода данных. Выводит ли сценарий сообщения об ошибках?

Не бывает глупых вопросов

В: Неужели небольшое количество уровней вложения имеет такое уж огромное значение?

О: Это зависит от обстоятельств. Если вы разрабатываете код, который только вы и будете видеть, и считаете, что сможете точно помнить в течение ближайших шести месяцев, если вам вдруг понадобится обратиться к нему для внесения изменений, за что отвечает каждая его строка, то пишите с вложениями этих конструкций, раз вам так проще. С другой стороны, если вы хотите, чтобы ваш код был как можно более строгим и логичным, можете использовать любые из логических операторов, которые мы уже рассматривали.

В: Как работает else?

О: В управляющей конструкции `if ... else` инвертированное условное выражение соответствует тому, чему не соответствует прямое условное выражение.

В: Гм... отлично. Означает ли это, что я могу вкладывать одни управляющие конструкции if ... else в другие?

О: В принципе, вы можете это делать. Но со всеми этими вложениями вы очень быстро усложните свой код, а мы здесь как раз пытаемся избежать этого.

Станьте чистильщиком кода управляющей конструкции IF

Ваша задача — играть роль управляющей конструкции if и расчищать запутанные вложенные конструкции IF и ELSE. Перепишите код так, чтобы избавиться от вложенных конструкций, но убедитесь, что он работает правильно.

Совет: вероятно, вы можете вообще обойтись без else!

```
if (empty($subject) && empty($text)) {
  echo 'Вы забыли ввести тему и содержание электронного письма.<br />';
} else {
  if (empty($subject) || empty($text)) {
    if (empty($subject) {
      echo 'Вы забыли ввести тему электронного письма.<br />';
    } else {
      echo 'Вы забыли ввести содержание электронного письма.<br />';
    }
  } else {
    // Все отлично, отправляйте электронное письмо
  }
}
```

Здесь мы проверяем, являются ли значения переменных $subject и $text пустыми.

```
if (empty($subject) && empty($text)) {
  echo 'Вы забыли ввести тему и содержание электронного письма.<br />';
}
```

Здесь мы проверяем, является ли значение переменной $subject пустым, а переменной $text — непустым.

Здесь мы проверяем, является ли значение переменной $text пустым, а переменной $subject — непустым.

```
if (empty($subject) && (!empty($text))) {
  echo 'Вы забыли ввести тему электронного письма.<br />';
}
if ((!empty($subject)) && empty($text)){
  echo 'Вы забыли ввести содержание электронного письма.<br />';
}
```

И здесь мы проверяем, являются ли значения переменных $subject и $text непустыми.

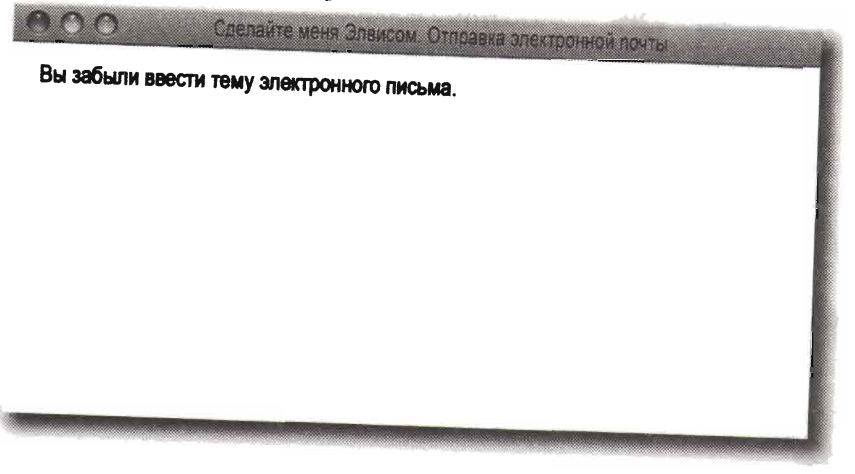
```
if ((!empty($subject)) && (!empty($text))) {
  // Все отлично, отправляйте электронное письмо
}
```

Логический оператор NOT (!) позволяет проверить, являются ли значения переменных $subject и $text непустыми.

Если бы мы не использовали логический оператор AND (&&) для того, чтобы изолировать непустые значения переменных $subject/$text, мы получали бы дополнительные сообщения. То же самое касается случая с непустым значением переменной $subject и пустым значение переменной $text.

Я потрясен. Когда я забыл ввести в форму тему электронного письма, то получил эту страницу. Но когда я нажал кнопку «Назад», мне пришлось вводить все электронное письмо заново.

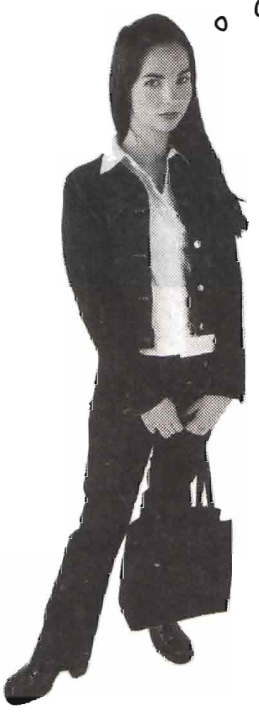
Эта страница сообщает Элмеру, что он сделал неправильно, но ничего более.



Проверка данных на достоверность в сценарии Элмера «Отправка электронной почты» работает, но сценарий мог бы делать более полезные вещи.

Когда сценарий `sendemail.php` видит недостоверные данные формы, он выводит сообщение об этом, но этим и ограничивается. Например, в нем нет ссылки на форму, которая его вызвала. Более того, когда Элмер возвращается к странице формы, используя средства своего браузера, информации, которую он только что вводил, там больше нет. Он вынужден повторно вводить тему электронного письма и его содержание.

Что, по вашему мнению, может улучшить обработку ошибок в сценарии «Отправка электронной почты», сделав его более полезным?



Было бы здорово показать форму вместе с сообщением об ошибке. Разве мы не можем вывести форму вновь, используя команду echo, если тема электронного письма или его содержание пусты?

Было бы действительно полезно вывести форму вновь, так как это избавило бы Элмера от необходимости возвращаться к ней, используя средства браузера.

Поэтому в дополнение к выводу сообщения об ошибке, когда одно или несколько полей формы остались незаполненными, мы должны также восстановить HTML-код формы из PHP-сценария, используя для этого команду echo. Этот код показывает, что PHP имеет возможность генерировать достаточно сложный HTML-код:

Этот PHP-код генерирует HTML-форму полностью, начиная с тега <form>.

```
echo '<form method="post" action="sendemail.php">';

echo '    <label for="subject">Тема электронного письма:</
label><br />';

echo '    <input id="subject" name="subject" type="text" size="30"
/><br />';

echo '    <label for="elvismail">Содержание электронного письма:</
label><br />';

echo '    <textarea id="elvismail" name="elvismail" rows="8"
cols="40" ></textarea><br />';

echo '    <input type="submit" name="Submit" value="Отправить"
/>';

echo '</form>';
```

Эти смещения не обязательны, но позволяют лучше видеть структуру HTML-кода.

Так как значения параметров HTML-кода ограничены двойными кавычками, проще использовать одинарные кавычки для ограничения каждой полной строки HTML-кода.

Если этот код кажется вам слегка загроможденным, то это оттого, что он действительно такой. Только лишь потому, что вы можете делать что-то в PHP, не значит, что вы должны это делать. В данном случае дополнительная сложность написания HTML-кода — это проблема. Генерировать такое большое количество строк кода с использованием команды `echo` — в действительности не слишком удачный выбор...

Необходим простой вход в PHP и выход из него

Многие часто забывают, что PHP-сценарий — это просто HTML-страница, содержащая PHP-код. Любой код в PHP-сценарии, не заключенный между тегами <?php и ?>, рассматривается как HTML-код. Это означает, что при необходимости вы можете закрыть блок PHP-кода и написать блок кода на HTML, а затем вернуться к кодированию на PHP. Это является исключительно удобной техникой для вывода фрагмента HTML-кода, который при выводе с использованием команды PHP echo получится слишком громоздким... как наш код формы «Отправить электронную почту».

В PHP-сценарии вы можете закрывать блоки PHP-кода для вывода фрагментов HTML-кода, а затем открывать их вновь.

```
<?php
   $from = 'elmer@makemeelvis.com';
   $subject = $_POST['subject'];
   $text = $_POST['elvismail'];

   if (empty($subject) && empty($text)) {
      // Мы знаем, что и тема электронного письма, и его содержание пусты
      echo 'Вы забыли ввести тему и содержание электронного письма.<br />';
?>

<form method="post" action="sendemail.php">
   <label for="subject">Тема электронного письма:</label><br />
   <input id="subject" name="subject" type="text" size="30" /><br />
   <label for="elvismail">Содержание электронного письма:</label><br />
   <textarea id="elvismail" name="elvismail" rows="8" cols="40" ></textarea><br
/>
   <input type="submit" name="Submit" value="Отправить" />
</form>

<?php
      }
   if (empty($subject) && (!empty($text))) {
      echo 'Вы забыли ввести тему электронного письма.<br />';
   }
   if ((!empty($subject)) && empty($text)) {
      echo 'Вы забыли ввести содержание электронного письма.<br />';
   }
   if (empty($subject) && empty($text)) {
      // Все отлично, отправляйте электронное письмо
      ...
   }
?>
```

Форма кодируется как нормальный HTML-код, так как этот код расположен вне PHP-тегов.

Этот тег ?> закрывает блок PHP-кода, возвращая нас к HTML.

Так как мы все еще внутри блока кода управляющей конструкции if, HTML-код будет выводить форму, если только оба поля формы останутся незаполненными.

Тег <?php открывает новый PHP-блок. Так как мы все еще внутри блока кода управляющей конструкции if, мы должны закрыть его, перед тем как идти дальше.

Напишите, какие, по вашему мнению, имеются ограничения в этом коде. Как бы вы устранили их?

Используйте флаг для предотвращения ~~повторения~~ повторения кода

Проблема предыдущего кода заключается в том, что он должен выйти из PHP-интерпретатора и повторить HTML-код по созданию формы в трех различных местах (по одному разу на каждый случай ввода недостоверных данных). Мы можем использовать переменную, принимающую значения `true/false`, как **флаг** (сигнал), чтобы отслеживать необходимость повторного вывода формы. Позднее в коде мы можем проверить значение этой переменной и вывести форму повторно, если оно будет `true` (флаг установлен).

Поэтому мы должны начать выполнение сценария, установив значение `false` для переменной `$output_form` (сбросить флаг), и затем изменить это значение на `true` (установить флаг) в том случае, если какое-либо поле формы останется незаполненным и нам необходимо вывести форму для заполнения повторно.

Первоначальное присваивание переменной `$output_form` значения `false` означает, что форма не будет выводиться, если не возникнет проблем с проверкой данных на достоверность.

Инициализация переменной `$output_form` **значением** `false`

ЕСЛИ в поле формы «Тема» не содержится текста
И в поле формы «Содержание» не содержится текста
ТОГДА вывести сообщение об ошибке,
присвоить значение `true` переменной `$output_form`

Эти сообщения об ошибках будут слегка отличаться в зависимости от того, какие поля ввода формы остались незаполненными.

ЕСЛИ в поле формы «Тема» не содержится текста
И в поле формы «Содержание» содержится текст
ТОГДА вывести сообщение об ошибке,
присвоить значение `true` переменной `$output_form`

ЕСЛИ в поле формы «Тема» содержится текст
И в поле формы «Содержание» не содержится текста
ТОГДА вывести сообщение об ошибке,
присвоить значение `true` переменной `$output_form`

Если какие-либо поля формы останутся незаполненными переменной `$output_form` будет присвоено значение `$true`, но форма выведена не будет... пока!

ЕСЛИ в поле формы «Тема» содержится текст
И в поле формы «Содержание» содержится текста
ТОГДА отправить электронное письмо

Если оба поля ввода формы заполнены, электронное письмо отправляется.

ЕСЛИ значение переменной `$output_form` **равно** `true`
ТОГДА вывести форму

На последнем этапе контроля достоверности данных проверяется значение `$output_form`, чтобы выяснить, нужно ли выводить форму для исправления ввода. В любом случае нам необходим HTML-код для повторного вывода формы, но только один раз.

Используйте повторный код HTML-формы только один раз

Создание нового кода проверки достоверности данных в PHP требует создания и инициализации новой переменной `$output_form` и затем изменения ее значения в зависимости от результатов контроля достоверности данных. Наиболее важным здесь является то, что форма выводится повторно (для исправления недостоверных данных) только в том случае, если значение переменной `$output_form` равно `true`.

Определяя зависимость вывода HTML-кода от флага, устанавливаемого в управляющих конструкциях if, мы предотвращаем повторение кода в нашем сценарии.

```
<?php
  $from = 'elmer@makemeelvis.com';
  $subject = $_POST['subject'];
  $text = $_POST['elvismail'];
  $output_form = false;

  if (empty($subject) && empty($text)) {
    // Мы знаем, что и тема электронного письма, и его содержание пусты
    echo 'Вы забыли ввести тему и содержание электронного письма.<br />';
    $output_form = true;
  }

  if (empty($subject) && (!empty($text))) {
    echo 'Вы забыли ввести тему электронного письма.<br />';
    $output_form = true;
  }

  if ((!empty($subject)) && empty($text)) {
    echo 'Вы забыли ввести содержание электронного письма.<br />';
    $output_form = true;
  }

  if ((!empty($subject)) && (!empty($text))) {
    // Все отлично, отправляйте электронное письмо
    ...
  }

  if ($output_form) {
?>
<form method="post" action="sendemail.php">
    <label for="subject">Тема электронного письма:</label><br />
    <input id="subject" name="subject" type="text" size="30" /><br />
    <label for="elvismail">Содержание электронного письма:</label><br />
    <textarea id="elvismail" name="elvismail" rows="8" cols="40" ></textarea><br />
    <input type="submit" name="Submit" value="Отправить" />
</form>
<?php
  }
?>
```

Здесь мы создаем нашу новую переменную и инициализируем ее значением false.

Мы присваиваем переменной значение true, если тема электронного письма и его содержание не введены. При этом форма будет выведена повторно, что даст возможность исправить ошибку.

Также присваиваем переменной значение true, если тема электронного письма не введена.

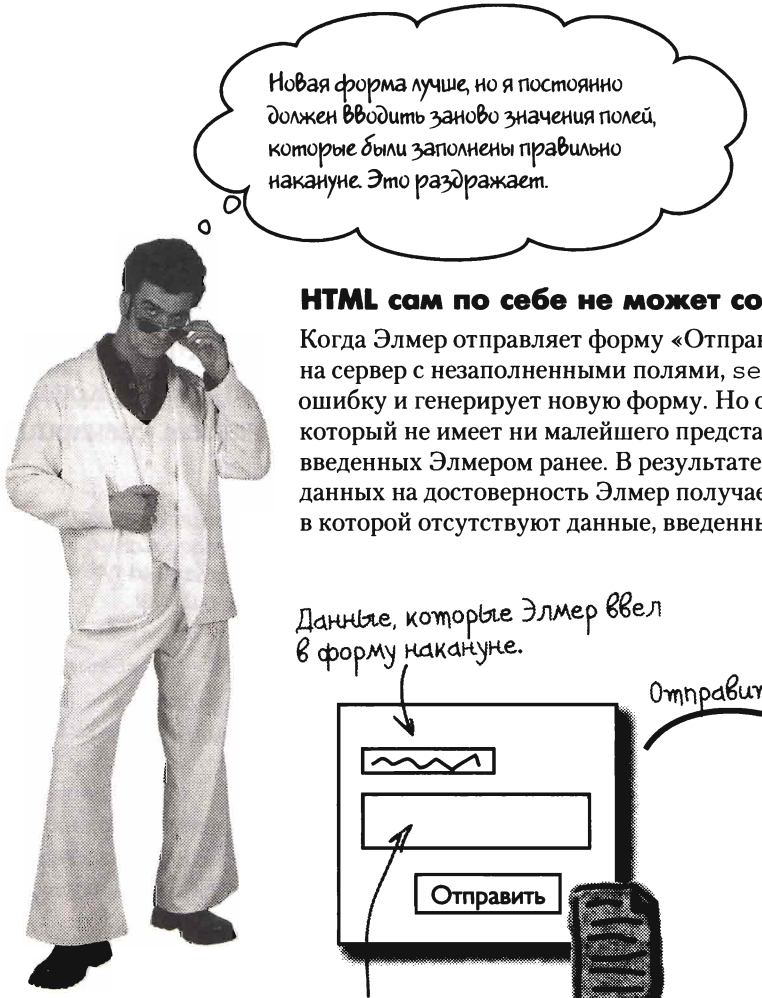
А здесь мы присваиваем переменной значение true, если не введено содержание электронного письма.

Эта управляющая конструкция if проверяет значение переменной $output_form и, если оно равно true, повторно выводит форму для исправления недостоверных данных.

Мы покинули PHP-код, но все, что предшествует фигурной скобке, закрывающей управляющую конструкцию if, считается относящимся к ее блоку кода. В данном случае это HTML-код формы ввода данных.

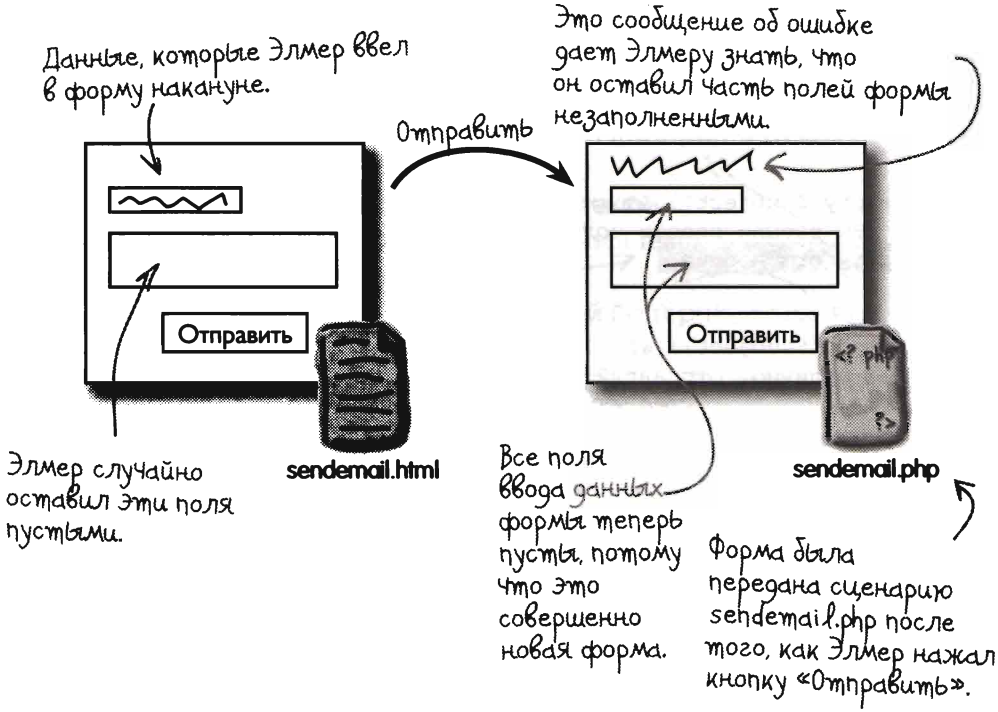
Повторный HTML-код формы ввода данных появляется только один раз, так как мы вместили всю логику, определяющую его появление, в единственную переменную $output_form.

Не забудьте вернуться в PHP-код и закрыть управляющую конструкцию if.



HTML сам по себе не может сохранить данные.

Когда Элмер отправляет форму «Отправка электронной почты» на сервер с незаполненными полями, `sendemail.php` перехватывает ошибку и генерирует новую форму. Но она генерируется HTML-кодом, который не имеет ни малейшего представления о значениях данных, введенных Элмером ранее. В результате как часть процесса проверки данных на достоверность Элмер получает совершенно пустую форму, в которой отсутствуют данные, введенные им ранее.



Мы не можем уйти от того факта, что новая форма должна генерироваться в PHP-сценарии. Но у нас должен быть способ запомнить данные, которые Элмер уже ввел накануне, и ввести их в новую форму, чтобы Элмер сосредоточился на вводе данных только в те поля формы, которые он по ошибке оставил пустыми...

Время поработать

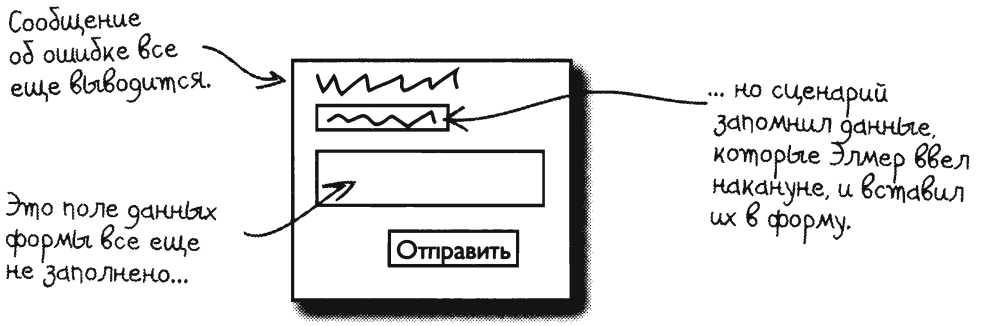
Нарисуйте, как должна выглядеть форма Элмера после того, как он отправил ее на сервер, заполнив только первое поле ввода данных. Затем напишите, какие, по вашему мнению, должны быть внесены изменения в оба файла (HTML и PHP), чтобы обеспечить новую функциональность.

sendemail.php

sendemail.html

Время поработать

Нарисуйте, как должна выглядеть форма Элмера после того, как он отправил ее на сервер, заполнив только первое поле ввода данных. Затем напишите, какие, по вашему мнению, должны быть внесены изменения в оба файла (HTML и PHP), чтобы обеспечить новую функциональность.



sendemail.php

sendemail.html

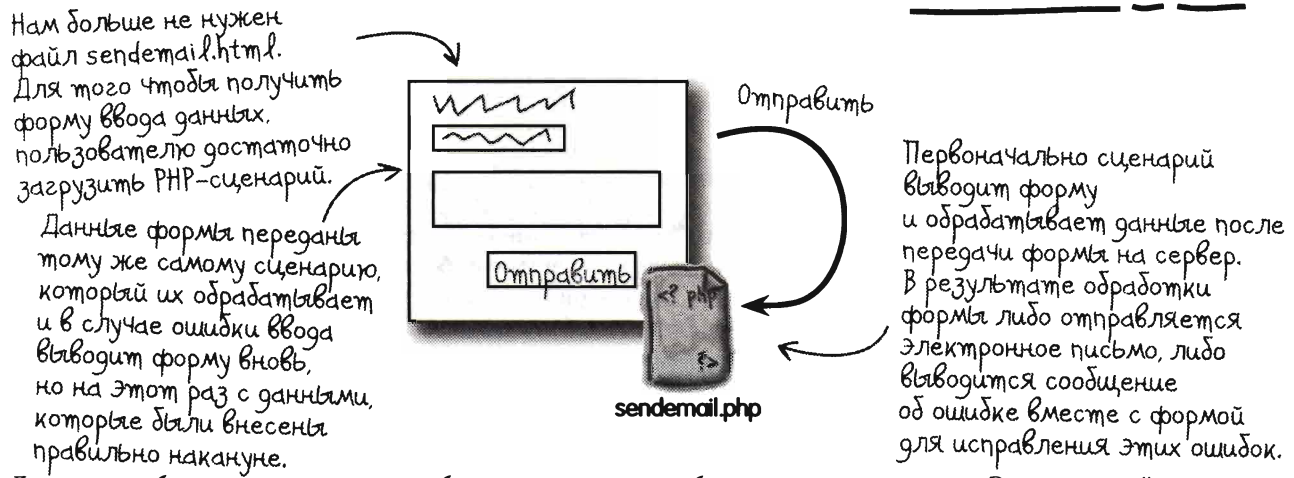
PHP-сценарий берет на себя задачу генерирования формы ввода данных как до, так и после отправки формы на сервер. А так как сценарий имеет доступ к любым данным формы, которые были введены в нее, он может вставить эти данные в соответствующие поля тогда, когда ему придется повторно генерировать форму. Это решает проблему Элмера, который должен был бы повторно вводить данные, уже введенные однажды.

Раз мы будем генерировать форму ввода данных исключительно с использованием PHP-сценария, то можем полностью избавиться от файла sendemail.html и предоставить возможность PHP-сценарию как выводить форму, так и обрабатывать ее данные. В результате PHP-сценарий, имея доступ ко всем данным, введенным в форму, может использовать их в своей работе, чего невозможно добиться средствами чистого HTML-кода.

Форма ввода данных, которая ссылается на себя

Как это возможно: удалить файл `sendemail.html` из уравнения формы «Отправка электронной почты»? Ответ заключается в том, что в действительности мы не убираем никакого HTML-кода, а просто передаем его генерацию PHP-сценарию. Это возможно потому, что в конечном счете результатом интерпретации PHP-кода является HTML-код — точно такой же, какой содержится в HTML-файлах.

HTML-форма ввода данных, генерируемая PHP-сценарием, который к тому же и обрабатывает ее, известна как форма, ссылающаяся на себя.



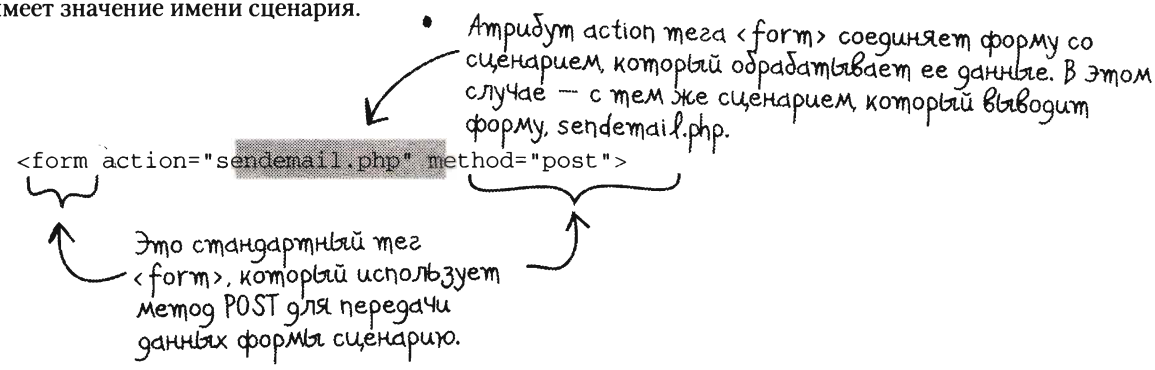
Для того чтобы понять, как все это работает, представьте себе, что происходит, когда Элмер первый раз загружает страницу (сценарий). Генерируется пустая форма в виде HTML-кода и выводится в окне браузера. Элмер заполняет поля ввода данных и нажимает кнопку «Отправить». Сценарий обрабатывает свою собственную форму и выводит сообщение об ошибке, если какие-либо поля ввода остались пустыми. Более важно то, что сценарий выводит форму вновь, но на этот раз она включает все данные, которые Элмер ввел правильно. Можно создать форму, которая будет достаточно интеллектуальной, чтобы запомнить данные, которые были введены в нее в предыдущий раз. В англоязычной литературе такое свойство определяется словом sticky, которому нет аналогов в русском языке в этом контексте.

Формы могут помнить данные, которые пользователь ввел правильно.

Что, по вашему мнению, может улучшить приложение Элмера, чтобы его форма помнила данные, которые он ввел накануне?

Передайте атрибуту action имя сценария

Как мы уже видели несколько раз, значение атрибута `action` тега `<form>` — это имя PHP-сценария, который обрабатывает данные формы. Присваивание имени сценария `sendemail.php` атрибуту `action` работает отлично, позволяя ему обрабатывать свои данные, и это первый шаг на пути придания форме свойств запоминать данные, введенные накануне. Фактически атрибут `action` уже имеет значение имени сценария.



Такой код будет работать при условии, что вы не будете переименовывать сценарий, забыв при этом соответственно изменить значение атрибута `action`. Но существует более надежный метод, который будет работать независимо от подобных обстоятельств, так как он не полагается на имя конкретного сценария. Он основан на использовании встроенной в PHP суперглобальной переменной `$_SERVER['PHP_SELF']`, которая содержит имя текущего сценария. Вы можете заменить имя сценария, которое вы присваиваете атрибуту `action`, на значение переменной `$_SERVER['PHP_SELF']` и больше не беспокоиться по поводу обновления кода в случаях, подобных описанному ранее.

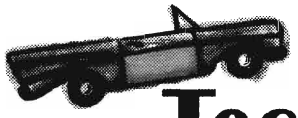
Единственное обстоятельство, про которое нельзя забывать, заключается в том, что `$_SERVER['PHP_SELF']` — это переменная PHP. Это означает, что вы должны использовать команду `echo`, чтобы ее вывод интерпретировался как HTML-, а не PHP-код. Например:

Вместо того чтобы жестко записывать в код имя сценария, вы можете сообщить ему, чтобы он ссылался на себя, используя суперглобальную переменную $_SERVER['PHP_SELF'].

```
<form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
```

Конечно, использование суперглобальной переменной `$_SERVER['PHP_SELF']` вместо имени сценария не является потрясающим достижением, но это один из многих приемов, которые позволяют делать ваши сценарии более удобными в сопровождении.

Суперглобальная переменная $_SERVER['PHP_SELF'] сохраняет имя текущего сценария.



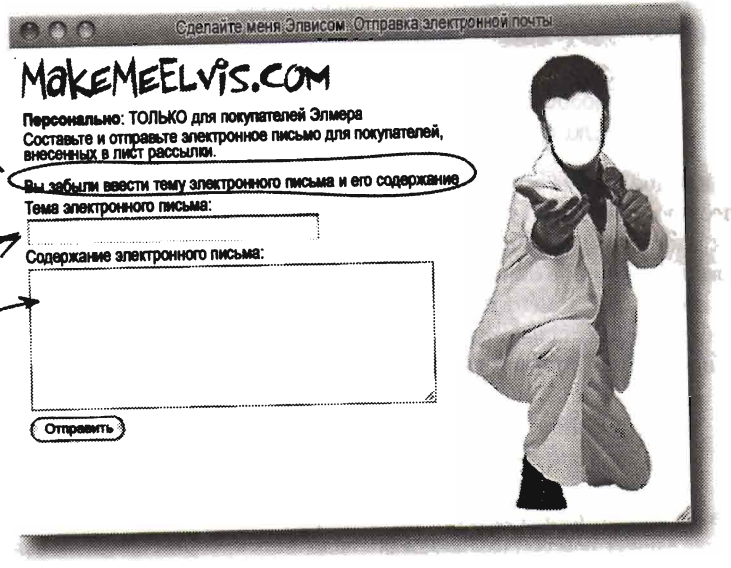
Тест-драйв

Проверьте новый, ссылающийся на самого себя сценарий с улучшенной логикой контроля достоверности.

Измените код сценария `sendemail.php`, используя переменную `$output_form` для избирательного вывода формы ввода данных, как показано несколькими страницами ранее. Измените также значение атрибута `action` тега `<form>` так, чтобы форма ссылалась на себя.

Вам больше не понадобится файл `sendemail.html`, так что можете совершенно спокойно его удалить. Загрузите новую версию сценария `sendemail.php` на ваш веб-сервер и откройте сценарий в браузере. Как он выглядит?

По какой-то причине сценарий выводит сообщение об ошибке, несмотря на то что форма еще даже и не отправлялась на сервер... Ничего хорошего.



Но это еще не все. Сценарий все еще не запоминает прежние данные.

Начнем с главного. Быстро сделаем все, что необходимо, чтобы сценарий запоминал свои данные.

Напишите, почему, по вашему мнению, сценарий выводит сообщение об ошибке вместе с первым выводом формы:

Проверка: была ли форма отправлена на сервер

Проблема в том, что сценарий не видит разницы в том, была форма выведена в результате первоначального вызова сценария или отправлена на сервер с неполными данными. Поэтому при первичной загрузке он выводит сообщение об ошибке, рапортуя, что ни в одно из полей данные не введены. Это вносит путаницу. Вопрос состоит в том, как мы можем узнать, что форма была передана серверу после заполнения полей. Если бы мы знали ответ на него, мы бы проверяли данные на достоверность только в этих случаях.

Помните, что, когда форма передается серверу для обработки с использованием метода `POST`, ее данные сохраняются в массиве `$_POST`? Если форма не была отправлена на сервер после заполнения, то в массиве `$_POST` не будет никаких данных. Говоря другими словами, массив `$_POST` **не будет инициализирован**. Догадываетесь, какую функцию нам необходимо вызвать, чтобы проверить, инициализирован ли массив `$_POST`?

Суперглобальная переменная $_POST позволяет нам проверять, была ли форма передана на сервер после внесения данных.

Функция isset() проверяет, инициализирована ли переменная.

Здесь должно быть имя тега <input> для вашей кнопки «Отправить».

```
if (isset($_POST['submit'])) {
  ...
}
```

Любой код, помещенный здесь, будет выполнен, если форма была отправлена на сервер.

Так как на каждой форме есть кнопка, в результате нажатия которой форма передается на сервер для обработки, то наиболее простой способ убедиться, что форма была передана на сервер, — это проверка, инициализирован ли элемент массива `$_POST` для этой кнопки. В этом случае данные — это всего лишь ярлык, появляющийся на изображении кнопки в окне браузера, и сами по себе никакой ценности они не представляют. Для нас важно, инициализирован ли элемент массива `$_POST['submit']`. Положительный ответ на этот вопрос означает, что форма была передана на сервер, после того как она была выведена для заполнения. Убедитесь только, что значение `'submit'` соответствует атрибуту `name` кнопки «Отправить» вашей формы.

Не бывает глупых вопросов

В: Как знание того, что форма была передана на сервер после внесения данных, поможет нам воспрепятствовать выводу сообщения об ошибке?

О: Причина, по которой сообщение об ошибке выводится неправильно, заключается в том, что сценарий не может определить, была ли форма передана на сервер после внесения данных или она выводится впервые. Поэтому нам необходим способ, который позволит узнать, что форма выводится в первый раз, и в этом случае совершенно нормальным является тот факт, что все поля пустые — это никакая не ошибка. Мы должны проверять данные на достоверность только в тех случаях, когда форма передается на сервер после ее заполнения. Именно поэтому выяснение вопроса, что послужило причиной вывода формы, очень важно.

В: Тогда почему бы нам не проверять на инициализацию действительно важные данные вместо ярлыка кнопки «Отправить»?

О: Было бы вполне приемлемым проверять инициализацию переменной `$_POST['subject']` или `$_POST['elvismail']`, но только для этой конкретной формы. Так как практически каждая форма имеет кнопку, нажатие которой отправляет ее на сервер для обработки, и, как правило, ее атрибуту `name` присваивается значение `'submit'`, то проверка на инициализацию элемента массива `$_POST['submit']` дает вам надежный способ определять, что послужило причиной вывода формы, для всех сценариев.

Сценарий «Отправка электронной почты» в деталях

```
<?php
  if (isset($_POST['submit'])) {
    $from = 'elmer@makemeelvis.com';
    $subject = $_POST['subject'];
    $text = $_POST['elvismail'];
    $output_form = false;

    if (empty($subject) && empty($text)) {
         // Мы знаем, что и тема электронного письма, и его содержание пусты
         echo 'Вы забыли ввести тему и содержание электронного письма.<br />';
         $output_form = true;
      }

      if (empty($subject) && (!empty($text))) {
         echo 'Вы забыли ввести тему электронного письма.<br />';
         $output_form = true;
      }

      if ((!empty($subject)) && empty($text)) {
         echo 'Вы забыли ввести содержание электронного письма.<br />';
         $output_form = true;
      }

      if (empty($subject) && empty($text)) {
        // код для отправки электронного письма
        ...
      }
  }
  else {
    $output_form = true;
  }

  if ($output_form) {
?>

  <form method="post" action="<?php echo $SERVER['PHP_SELF']; ?>">
      <label for="subject">Тема электронного письма:</label><br />
      <input id="subject" name="subject" type="text" size="30" /><br />
      <label for="elvismail">Содержание электронного письма:</label><br />
      <textarea id="elvismail" name="elvismail" rows="8" cols="40" ></textarea><br />
      <input type="submit" name="Submit" value="Отправить" />
   </form>

<?php
  }
?>
```

Мы проверяем значение переменной $_POST['submit']. Если форма не была передана на сервер в результате нажатия кнопки «Отправить», эта переменная не будет инициализирована.

Эта фигурная скобка закрывает первую управляющую конструкцию if, которая управляет процессом в зависимости от того, была ли форма открыта для ввода данных или передана на сервер для обработки после ввода данных.

Если форма не была передана на сервер для обработки после ввода данных, нам определенно необходимо ее открыть.

Отлично. Итак, теперь мы можем определить, когда форма передается на сервер для проверки данных, и вывести правильное сообщение об ошибке. Но мы до сих пор не сделали, чтобы сценарий запоминал уже введенные значения полей формы, так ведь?

Это верно. Определение, когда форма передается на сервер для проверки, а когда выводится впервые для заполнения ее полей, очень важно, но нам все еще необходимо найти метод, позволяющий выводить однажды внесенные данные.

Знание того, что форма передана на сервер для проверки данных, является важным моментом в процессе придания ей способности выводить однажды уже введенные данные, но это только полдела. Другая часть заключается в том, чтобы получить значения данных, которые мы ввели правильно, и занести их в поля выводимой формы. Вы можете занести определенные значения в поля формы, используя атрибут `value` HTML-тега `<input>`. Например, этот код установит значение поля ввода данных формы с использованием атрибута `value`:

Это значение жестко закодировано, и оно всегда будет таким при каждом выводе формы.

```
<input name="subject" type="text" value="Осенняя распродажа!">
```

Но мы не хотим жестко устанавливать определенное значение. Мы хотим ввести значение PHP-переменной. Как это возможно? Вспомните, как в других ситуациях мы использовали команду `echo` для того, чтобы динамически генерировать HTML-код из PHP. В этом случае мы можем использовать команду `echo` для того, чтобы динамически присвоить значение PHP-переменной атрибуту `value`, как показано ниже:

Так как мы переключились на PHP, чтобы вывести значение переменной с помощью команды echo, то должны использовать тег <?php.

Значение переменной выводится с помощью известной команды echo.

```
<input name="subject" type="text" value="<?php echo $subject; ?>">
```

А для возвращения назад к HTML-коду мы закрываем PHP-код с помощью тега ?>.

Форма Элмера может быть модифицирована аналогичным способом для использования всех преимуществ такого запоминания данных.

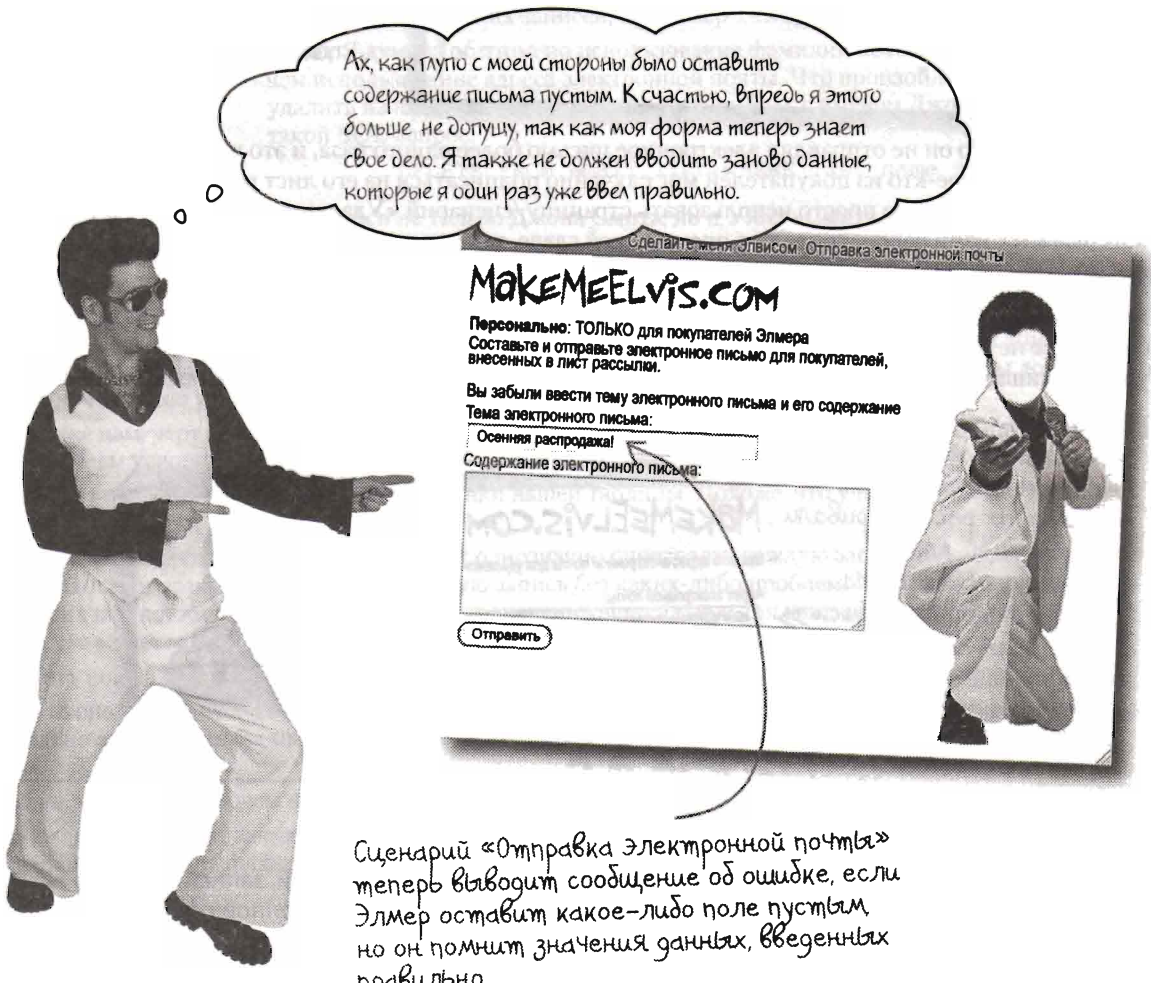
Для поля ввода типа textarea мы не используем атрибут value, а выводим значение переменой $text между тегами <textarea> и </textarea>.

```
<form method="post" action="<?php echo $SERVER['PHP_SELF']; ?>">
    <label for="subject">Тема электронного письма:</label><br />
    <input id="subject" name="subject" type="text" size="30"
  value="<?php echo $subject; ?>" /><br />
    <label for="elvismail">Содержание электронного письма:</
label><br />
    <textarea id="elvismail" name="elvismail" rows="8" cols="40" >
  <?php echo $text; ?></textarea><br />
    <input type="submit" name="Submit" value="Отправить" />
</form>
```

Тест-драйв

Проверьте, насколько данные, однажды введенные Элмером, запоминаются на самом деле.

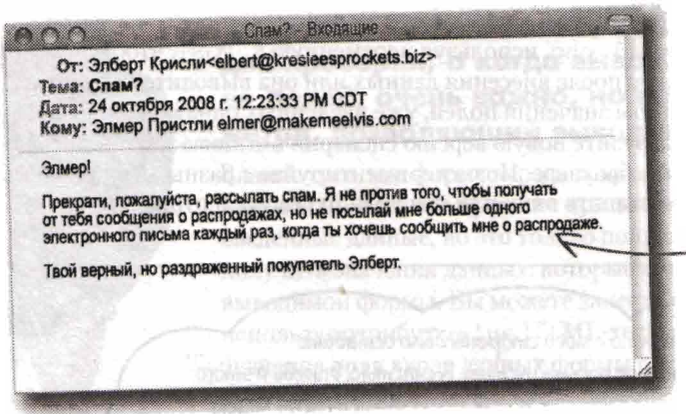
Измените код сценария `sendemail.php`, используя переменную `$_POST`, чтобы проверить, была ли форма передана на сервер после внесения данных или она выводится впервые. Добавьте также код, для того чтобы значения полей, уже введенных однажды, запоминались и помещались на свои места. Загрузите новую версию сценария `sendemail.php` на ваш веб-сервер и откройте сценарий в браузере. Поэкспериментируйте с разными значениями полей, в том числе попробуйте оставить одно или оба поля пустыми, и отправьте их на сервер несколько раз.



Сценарий «Отправка электронной почты» теперь выводит сообщение об ошибке, если Элмер оставит какое-либо поле пустым, но он помнит значения данных, введенных правильно.

Некоторые покупатели все еще сердятся

Проверка данных на достоверность прошла большой путь, чтобы удовлетворять претензии недовольных покупателей, особенно тех из них, которые получали пустые электронные письма. Но не все удовлетворены. Похоже, что некоторое количество людей продолжают получать одни и те же электронные письма по несколько раз... Помните того парня, о котором мы говорили ранее в этой главе?

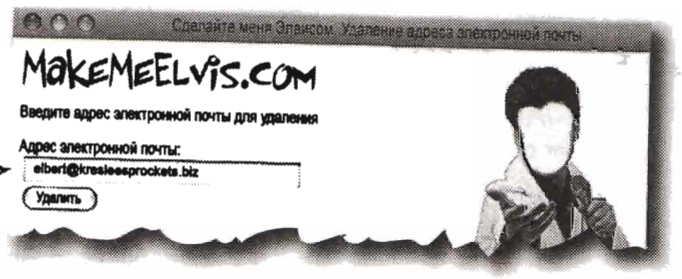


Этот покупатель недоволен, потому что он получает одновременно по несколько копий одного и того же электронного письма Элмера.

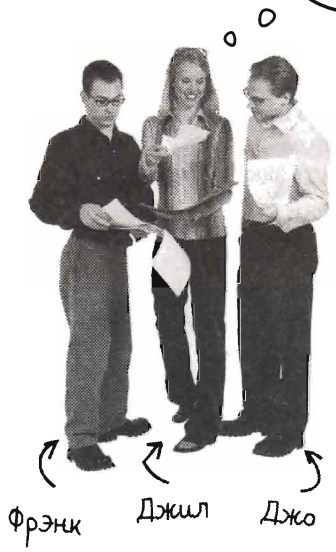
Элмер знает, что он не отправлял электронное письмо более одного раза, и это наводит его на мысль, что, возможно, кое-кто из покупателей мог случайно подписаться на его лист рассылки несколько раз. В чем проблема, нужно просто использовать страницу/сценарий «Удаление электронного адреса», которые мы с вами рассматривали в предыдущей главе, так?

К сожалению, все не так просто. Удалив Элберта с использованием его адреса электронной почты, мы удалим все записи с таким адресом электронной почты из таблицы `email_list`. В результате он больше не получит ни одного электронного письма от Элмера. Нам необходим способ, позволяющий удалять **лишние** записи с адресом электронной почты Элберта, обязательно оставляя при этом одну.

При использовании страницы «Удаление электронного адреса», которую мы рассматривали в предыдущей главе, из базы данных Элмера будут удалены все записи с указанным адресом электронной почты, а это совсем не то, к чему мы стремимся.



Как Элмер сможет удалить все, кроме одной, записи таблицы, которые имеют одинаковые адреса электронной почты?



Джо: Может быть, наша форма «Добавление адреса электронной почты» должна проверять, не существует ли уже в таблице запись с таким адресом электронной почты, перед тем как добавлять запись с данными на нового покупателя? Это бы решило проблему, не так ли?

Фрэнк: Великолепная идея.

Джил: Да, это не допустило бы возникновения таких проблем в будущем, но это никак не поможет нам разобраться с одинаковыми записями, которые уже имеются в таблице.

Фрэнк: Точно. А что если мы попробуем использовать другую колонку таблицы при удалении лишних записей, например `last_name`?

Джил: Я думала об этом, но использование фамилии потенциально даже хуже, чем использование адреса электронной почты. Что произойдет, если мы захотим удалить из нашего листа рассылки кого-нибудь с именем Джон Смит и сделаем такой SQL-запрос:

```
DELETE FROM email_list WHERE last_name = 'Смит'
```

Джо: Мы удалим из нашей таблицы записи не только Джона Смита, но и Уилла Смита, Мэгги Смит, Эмми Смит...

Фрэнк: Ох, я не вижу в этом ничего хорошего. Записи с одинаковыми фамилиями встречаются чаще, чем с одинаковыми адресами электронной почты, а с одинаковыми именами — и того чаще. Мы можем потерять десятки и десятки записей в результате выполнения всего лишь одного запроса.

Джил: Точно. Мы не можем рисковать, используя ограничивающее условие `WHERE`, приводящее к тому, что в результате выполнения запроса будут удалены записи, которые должны быть сохранены. Мы должны быть уверены в том, что точно указываем на ту единственную запись, которую намерены удалить.

Джо: Так как же нам, черт возьми, быть? Мы не можем использовать ни `email`, ни `last_name`, ни `first_name` в ограничивающем условии `WHERE`.

Фрэнк: У нас нет возможности использовать колонки нашей таблицы. Похоже, что у нас вообще нет никаких шансов.

Джил: Не факт. Нам необходимо что-то, что будет однозначно определять каждую запись таблицы. Это что-то и даст нам возможность указать нужную запись без каких-либо проблем. И только лишь то, что у нас до сих пор нет колонки, которая содержит значение, уникальное для каждой записи, совершенно не означает, что мы не можем добавить такую колонку.

Джо: Добавить новую колонку? Так ведь мы уже решили все вопросы, связанные со структурой таблицы.

Фрэнк: Так-то оно так. Да только то, что у нас получилось, не отвечает нашим запросам. Конечно, ты прав, было бы значительно лучше, если бы мы подумали об этом в самом начале и соответственно построили структуру нашей таблицы, но еще не поздно исправить ситуацию.

Джо: Хорошо, но как мы назовем нашу новую колонку? И какие данные будем в нее помещать?

Джил: Раз ее функция будет заключаться в уникальной идентификации каждой записи в таблице, мы могли бы назвать ее identifier (идентификатор), или просто `id` для краткости.

Фрэнк: Отлично, и мы можем заполнить колонку `id` различными числами, уникальными для каждой записи. А когда возникнет необходимость в удалении какой-либо записи, мы сможем составить запрос `DELETE`, основанный на значении этой колонки, а не на адресе электронной почты или фамилии.

Джо: Точно. Это действительно отличная идея, не так ли?

Записи таблицы должны уникально идентифицироваться

Идея сохранения данных в базе заключается в том, что позднее вам может понадобиться извлечь их оттуда для просмотра и/или дальнейшего преобразования. С учетом этого становится очевидным, насколько важно то, чтобы каждая запись в таблице могла быть **уникально идентифицирована**. Это дает вам возможность получить доступ к одной конкретной записи (и только к ней!). Таблица Элмера `email_list` создана с потенциально опасным предположением, что адреса электронной почты являются уникальными величинами. Это предположение подтверждается до тех пор, пока никто ошибочно не подпишется на лист рассылки во второй раз. Но когда кто-нибудь сделает это (а это произойдет обязательно!), его адрес электронной почты будет сохранен в таблице дважды... И на этом вся уникальность заканчивается!

Что таблица Элмера содержит сейчас:

first_name	last_name	email
Денни	Баблтон	denny@mightygumball.net
Ирма	Верлиц	iwer@aliensabductedme.com
Элберт	Крисли	elbert@kresleesprockets.biz
Ирма	Крисли	elbert@kresleesprockets.biz

Ничто в структуре этой таблицы не гарантирует уникальности ее записей.

И хотя в большинстве случаев адреса электронной почты не повторяются, мы не можем рассчитывать, что так будет всегда.

Несколько человек могут иметь одинаковые имена, поэтому эта колонка не может считаться удачным выбором для обеспечения уникальности.

То же самое здесь: фамилия также не обеспечивает уникальности.

Если в вашей таблице нет колонки, содержащей действительно уникальные данные, вам нужно создать ее. Сервер баз данных MySQL предоставляет вам возможность создать колонку, содержащую целые числа, уникальные для каждой записи. Такая колонка также известна под названием **«первичный ключ»**.

Что должна содержать таблица Элмера:

Нам необходима новая колонка, которая должна содержать данные, уникальные для каждой записи.

id	first_name	last_name	email
1	Денни	Баблтон	denny@mightygumball.net
2	Ирма	Верлиц	iwer@aliensabductedme.com
3	Элберт	Крисли	elbert@kresleesprockets.biz
4	Ирма	Крисли	elbert@kresleesprockets.biz

Теперь, когда эта колонка содержит уникальные значения, мы можем быть уверены, что каждая запись в нашей таблице действительно уникальна.

Дублирующиеся данные в других колонках больше не оказывают влияния на уникальность записей, потому что ее обеспечивает новая колонка id.

Эй, гений! Ты же знаешь, что если мы хотим внести изменения в структуру таблицы, то должны удалить ее с помощью запроса DROP TABLE и потом воссоздать с нуля. Данные с адресами электронной почты Элмера испарятся!

Это верно: в результате выполнения запроса DROP TABLE все данные будут утеряны. Но в SQL есть другой запрос, с помощью которого вы можете внести изменения в существующую структуру таблицы без какой-либо потери данных.

Этот запрос начинается с ключевых слов ALTER TABLE (изменить таблицу), и мы можем использовать его, чтобы создать новую колонку без предварительного удаления таблицы и потери данных. Ниже показано, как выглядит общий формат запроса ALTER TABLE, используемого для добавления новой колонки:

Имя таблицы, структура которой должна быть изменена.

Имя колонки, которая должна быть добавлена.

```
ALTER TABLE имя_таблицы ADD имя_колонки тип_колонки
```

Тип данных новой колонки.

Мы можем использовать запрос ALTER TABLE для того, чтобы добавить новую колонку с именем id в таблицу email_list. Мы определим INT как тип данных для колонки id, так как целые числа очень легко использовать для поддержки уникальности. Кроме этого, в запросе необходимо указать некоторую информацию, которая показана ниже:

Имя таблицы, структуру которой мы хотим изменить.

Мы хотим добавить новую колонку с именем id.

Это говорит MySQL увеличивать значение, сохраняемое в данной колонке, на единицу для каждой записи, когда она добавляется к таблице.

```
ALTER TABLE email_list ADD id INT NOT NULL AUTO_INCREMENT FIRST,
  ADD PRIMARY KEY (id)
```

Этот тип данных колонки говорит о том, что в ней будут сохраняться целые числа.

Ключевое слово FIRST (первый) сообщает MySQL, что новая колонка должна быть первой по порядку в структуре таблицы. Это необязательное условие, но хороший стиль программирования предполагает размещение колонки идентификатора первой по порядку.

Небольшой фрагмент кода, сообщающий MySQL о том, что новая колонка является первичным ключом. Более подробно об этом, буквально через секунду!

У запроса ALTER TABLE длинное продолжение потому, что должен быть создан первичный ключ с достаточно специфическими свойствами. Например, ключевые слова NOT NULL (не пусто) сообщают MySQL, что в новой колонке id обязательно должны быть данные — вы не сможете оставить ее пустой. Следующее ключевое слово AUTO_INCREMENT (автоматическое приращение) описывает особенность колонки id, заключающуюся в том, что в нее автоматически будет добавляться уникальное значение при создании каждой новой записи. Как следует из смысла этого ключевого слова, при добавлении в таблицу новой записи сервер баз данных MySQL автоматически прибавляет единицу к значению id последней введенной записи и сохраняет это значение в колонке id добавляемой записи. И наконец, ключевые слова ADD PRIMARY KEY (добавить первичный ключ) говорят серверу баз данных MySQL о том, что значения колонки id являются уникальными. Впрочем, только уникальностью данных не исчерпывается все понятие первичного ключа...

Первичные ключи обеспечивают уникальность

Первичный ключ — это колонка в таблице, которая уникально определяет каждую запись таблицы. В отличие от обычных колонок, которые также могут иметь ограничения на содержание уникальных значений, только одна колонка может быть определена как первичный ключ. Это делается для того, чтобы было понятно, какую колонку использовать в запросе для идентификации конкретной записи.

Первичный ключ — это колонка в вашей таблице, которая делает каждую запись уникальной.

Для обеспечения такой уникальности первичных ключей MySQL налагает ряд ограничений на колонку, которая объявляется как первичный ключ. Вы можете рассматривать эти ограничения как правила, которым необходимо следовать, когда вы имеете дело с первичными ключами.

Пять правил, касающихся первичных ключей:

Данные в первичных ключах не должны повторяться.

Две разные записи не должны иметь одного и того же значения первичного ключа. Никаких исключений в этом правиле не допускается. Первичный ключ в данной таблице всегда должен иметь уникальное значение.

Первичные ключи не могут быть пустыми (иметь значение NULL).

Если первичный ключ будет оставлен пустым (`NULL`), то он может оказаться неуникальным, так как другая запись потенциально может тоже иметь пустое значение первичного ключа. Всегда присваивайте уникальное значение первичному ключу.

Первичным ключам должно быть присвоено значение при добавлении новой записи.

Если бы вы могли добавить запись без первичного ключа, то подверглись бы риску получить запись со значением первичного ключа `NULL`, что, в свою очередь, могло бы привести к нарушению принципа уникальности первичного ключа.

Первичные ключи должны быть максимально эффективными.

Первичный ключ должен содержать ровно столько информации, сколько необходимо для поддержания его уникальности, и не больше. Вот почему целые числа — хорошие кандидаты на роль первичных ключей. Они позволяют очень эффективно обеспечивать уникальность, не требуя большого количества компьютерных ресурсов.

Значения первичного ключа не могут быть изменены.

Если бы вы имели возможность менять значение первичного ключа, то рисковали бы присвоить ему по ошибке значение, которое уже используется для другой записи. Не забывайте: необходимо обеспечивать уникальность первичного ключа любой ценой.

Колонка id в таблице Элмера не содержит повторяющихся данных, имеет значение для каждой записи (эти значения устанавливаются автоматически, когда новая запись добавляется в таблицу), компактна и не изменяется. Великолепно!

id	first_name	last_name	email
1	Денни	Баблтон	denny@mightygumball.net
2	Ирма	Верлиц	iwer@aliensabductedme.com
...			

Тест-драйв

Измените таблицу Элмера и попробуйте добавить новую запись с первичным ключом.

Используя инструментальную программу, такую как MySQL-терминал, или вкладку SQL-программы phpMyAdmin, введите запрос `ALTER TABLE`, чтобы добавить колонку с именем `id`, которую объявите первичным ключом:

```
ALTER TABLE email_list ADD id INT NOT NULL AUTO_INCREMENT FIRST,
  ADD PRIMARY KEY (id)
```

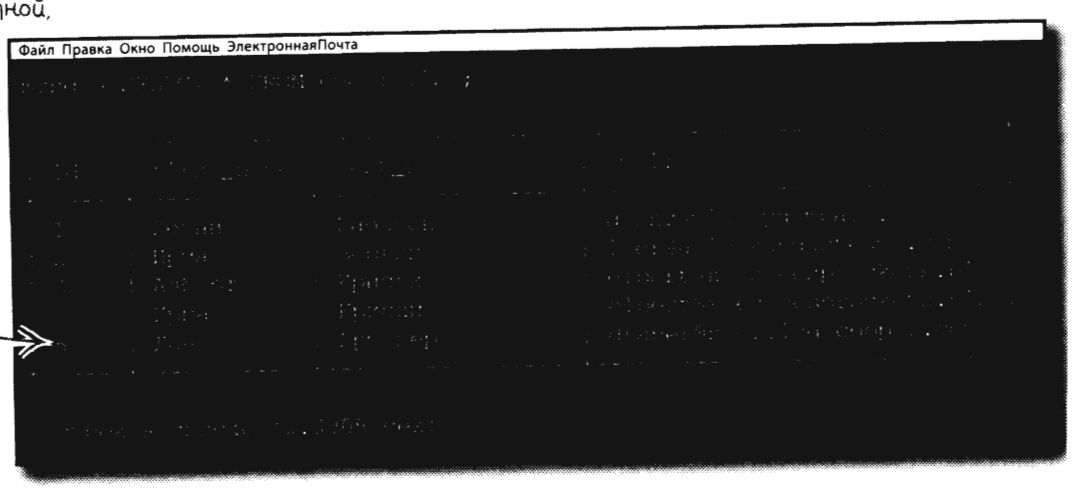
Теперь добавьте в таблицу нового покупателя, чтобы убедиться, что колонка `id` автоматически получит новое значение. Вот пример запроса `INSERT` (обратите внимание на то, что первичный ключ в запросе не упоминается):

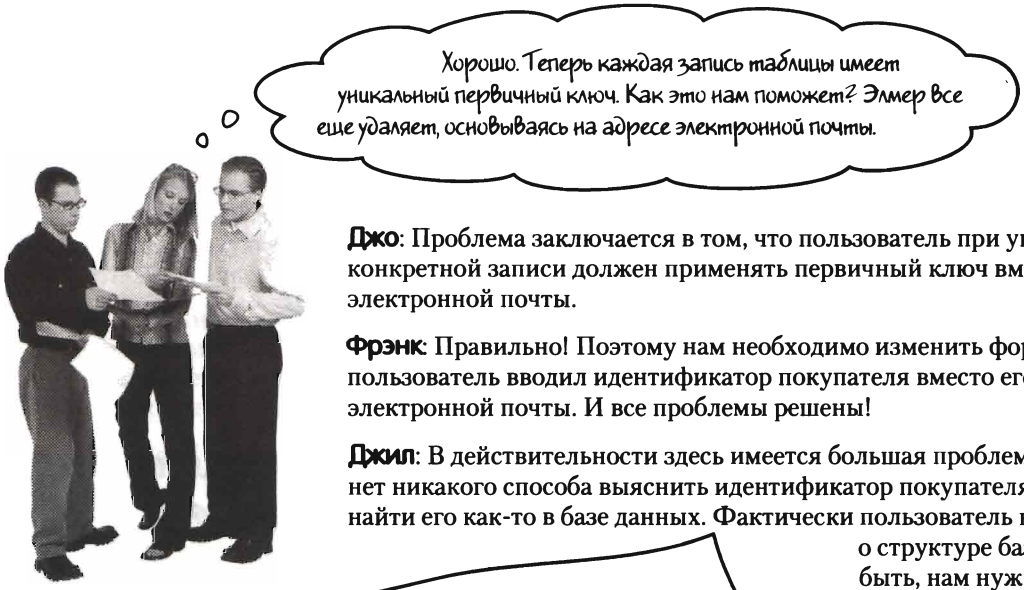
```
INSERT INTO email_list (first_name, last_name, email)
  VALUES ('Дон', 'Дрэппер', 'drapper@sterling-cooper.com')
```

И, наконец, отправьте на сервер запрос `SELECT`, чтобы увидеть содержимое таблицы и наблюдать новый первичный ключ во всей его красе! На всякий случай, если вы забыли, вот запрос `SELECT`:

```
SELECT * FROM email_list
```

Новая колонка id является автоинкрементной, поэтому она остается уникальной для новой записи.

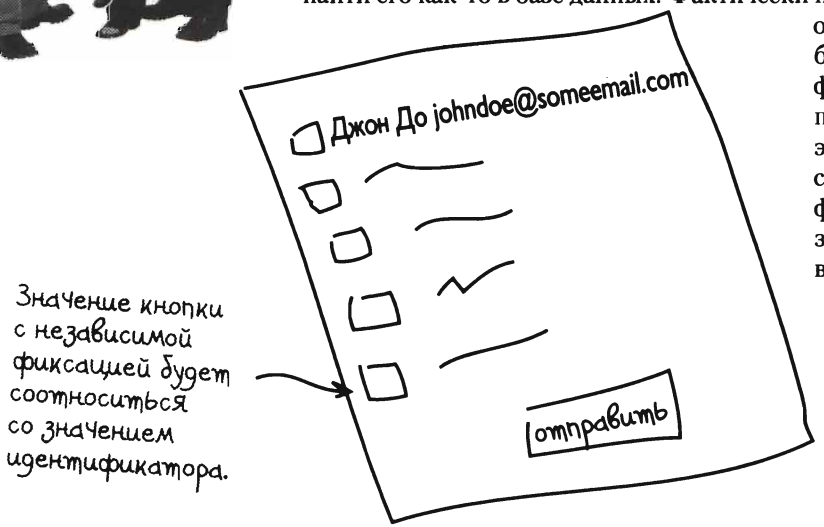




Джо: Проблема заключается в том, что пользователь при указании конкретной записи должен применять первичный ключ вместо адреса электронной почты.

Фрэнк: Правильно! Поэтому нам необходимо изменить форму так, чтобы пользователь вводил идентификатор покупателя вместо его адреса электронной почты. И все проблемы решены!

Джил: В действительности здесь имеется большая проблема. У пользователя нет никакого способа выяснить идентификатор покупателя, кроме как найти его как-то в базе данных. Фактически пользователь не знает ничего о структуре базы данных. Может быть, нам нужно пересмотреть форму так, чтобы на ней перечислялись имена и адреса электронной почты вместе с кнопками с независимой фиксацией напротив каждой записи? Вот здесь я сделала для вас небольшую схему.



Фрэнк: Хороший эскиз, только как все это поможет Элмеру указать запись покупателя, которого он хочет удалить, используя его идентификатор?

Джо: Гм... А что если мы используем идентификатор покупателя в качестве значения кнопки с независимой фиксацией? В этом случае он не будет виден, но сценарий станет иметь к нему доступ.

Джил: Отличная идея. Тогда мы могли бы генерировать форму автоматически, в цикле, используя результаты запроса `SELECT`, чтобы получить все данные, и создавая для каждой записи кнопку с независимой фиксацией.

Джо: Отлично. Но что будет происходить при нажатии кнопки «Отправить»? Какие значения примут элементы массива `$_POST`?

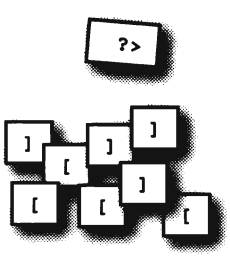
Фрэнк: Подожди, Джо, мы узнаем все это через минуту. Давайте начнем с того, что создадим ту часть сценария, в результате интерпретации которой будут выведены все данные из таблицы вместе с кнопками с независимой фиксацией.

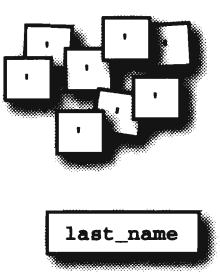
Магниты PHP и MySQL

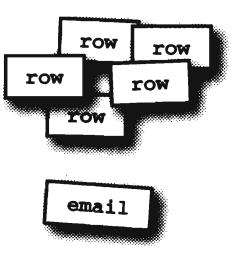
Используйте магниты, изображенные ниже, чтобы закончить код для сценария «Удаление электронной почты», который должен представить перечень покупателей из базы данных Элмера вместе с кнопками с независимой фиксацией. Имейте в виду, что этот код должен только создавать форму. Не думайте пока о коде, который отвечает за непосредственное удаление записей из таблицы.

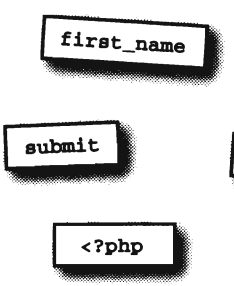
```
<img src="blankface.jpg" width="161" height="350" alt="" style="float:right" />
<img name="elvislogo" src="elvislogo.gif" width="229" height="32" border="0" alt="Сделайте меня Элвисом" />
<p>Выберите, пожалуйста, адреса электронной почты, которые вы хотите удалить из листа рассылки,
а нажмите кнопку «Удалить».</p>

<form method="post" action="..... echo $_SERVER['PHP_SELF']; .....">


<?php
  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    or die('Ошибка соединения с MySQL-сервером.');

  // Ввод записей покупателей вместе с кнопками с независимой фиксацией.
  $query = "SELECT * FROM email_list";
  $result = mysqli_query($dbc, $query);

  while (..... = mysqli_fetch_array($result)) {

    echo '<input type="checkbox" value="' . .....'" name="todelete[]" />';

    echo .....;

    echo ' ' . .....;

    echo ' ' . .....;

    echo '<br />';
  }

  mysqli_close($dbc);
?>

  <input type="submit" name="              " value="Удалить" />

</form>
```

removeemail.php









Магниты PHP и MySQL

Используйте магниты, изображенные ниже, чтобы закончить код для сценария «Удаление электронной почты», который должен представить перечень покупателей из базы данных Элмера вместе с кнопками с независимой фиксацией. Имейте в виду, что этот код должен только создавать форму. Не думайте пока о коде, который отвечает за непосредственное удаление записей из таблицы.

```
<img src="blankface.jpg" width="161" height="350" alt="" style="float:right" />
<img name="elvislogo" src="elvislogo.gif" width="229" height="32" border="0" alt="Сделайте меня Элвисом" />
<p>Выберите, пожалуйста, адреса электронной почты, которые вы хотите удалить из листа рассылки
и нажмите кнопку «Удалить».</p>

<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">

<?php
  $dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
    or die('Ошибка соединения с MySQL-сервером.');

  // Ввод записей покупателей вместе с кнопками с независимой фиксацией.
  $query = "SELECT * FROM email_list";
  $result = mysqli_query($dbc, $query);

  while ( $row = mysqli_fetch_array($result)) {

    echo '<input type="checkbox" value="' . $row['id'] . '" name="todelete[]" />';

    echo $row['first_name'];

    echo ' ' . $row['last_name'];

    echo ' ' . $row['email'];

    echo '<br />';
  }

  mysqli_close($dbc);
?>

  <input type="submit" name="submit" value="Удалить" />

</form>
```

Форма ссылается на себя.

Встроенный PHP-код должен быть заключен между тегами <?php и ?>.

Это место, где значения первичного ключа используются для присвоения значений кнопкам с независимой фиксацией. Эти значения мы сможем использовать далее, когда будем удалять записи покупателей, определяемые этими значениями.

Для каждой записи создается кнопка с независимой фиксацией.

removeemail.php

Пока что в сценарии нет кода для непосредственного удаления записей. На этом этапе он только представляет перечень записей вместе с кнопками с независимой фиксацией для каждой из них.

Вы можете дать этой кнопке любое имя. Не забывайте только, что оно должно соответствовать идентификатору элемента массива $_POST, если вы решите проверять позднее, была ли форма отправлена на сервер для проверки данных или выводится для первоначального их занесения.

От кнопок с независимой фиксацией к идентификаторам покупателей

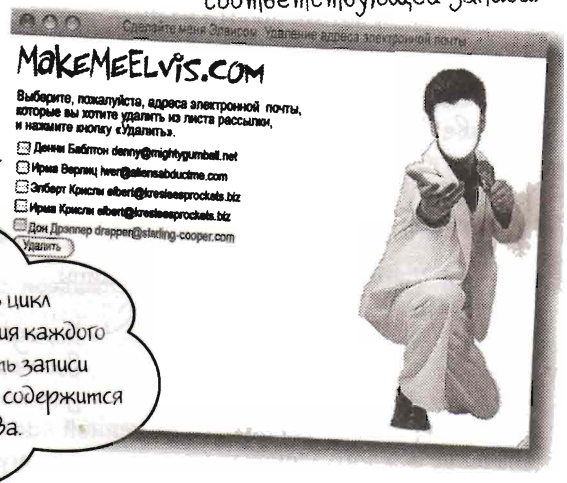
Код, который генерирует кнопку с независимой фиксацией в сценарии «Удалить адрес электронной почты», — это почти обычный HTML-код. При этом в качестве значения атрибута `value` тега `<input>` используется значение первичного индекса соответствующей записи. Однако для кнопки с независимой фиксацией имеется небольшое, но очень важное отличие от обычного HTML-кода. Возможно, вы заметили квадратные скобки (`[]`) в конце имени кнопки с независимой фиксацией. Они имеют очень важное значение.

```
echo '<input type="checkbox" value="' . $row['id'] . '" name="todelete[]">';
```

Квадратные скобки в конце имени кнопки с независимой фиксацией влекут за собой создание внутри суперглобального массива `$_POST` еще одного массива, в котором сохраняются значения атрибутов `value` всех нажатых кнопок с независимой фиксацией. Так как значение атрибута `value` каждой кнопки с независимой фиксацией — это значение первичного индекса соответствующей записи, **каждое значение массива `todelete` является идентификатором записи, которую необходимо удалить из таблицы**. Это дает нам возможность получить в цикле значения всех элементов массива `todelete` и создать SQL-запрос на удаление всех покупателей, помеченных в форме на удаление.

Квадратные скобки в конце имени кнопки с независимой фиксацией влекут за собой автоматическое создание элемента массива todelete со значением идентификатора соответствующей записи.

Каждая кнопка с независимой фиксацией содержит значение идентификатора, к которому можно получить доступ через суперглобальный массив $_POST.



Я все поняла.
Нам осталось только использовать цикл while, для того чтобы получить значения каждого из элементов массива todelete, и удалить записи тех покупателей, чей идентификатор содержится в соответствующем элементе массива.

Мы могли бы использовать цикл while, но существует более тонкое решение: использовать цикл другого типа.

Цикл `foreach` — это вид цикла, разработанный специально для извлечения значений элементов массива. Все, что вам необходимо в случае использования этого цикла, — это указать массив и переменную, которой будут присваиваться значения элементов этого массива в процессе прохождения цикла. PHP будет последовательно проходить все элементы массива — один за другим. Нет никакой необходимости в проверке условия.

Напишите, как, по вашему мнению, цикл foreach будет проходить по элементам массива, содержащего идентификаторы покупателей Элмера:

.....

Прохождение по элементам массива в цикле foreach

Цикл `foreach` последовательно проходит все элементы массива — один за другим. При этом нет никакой необходимости в проверке каких-либо условий или подсчете числа проходов. По мере прохождения каждого из элементов массива значение этого элемента сохраняется во временной переменной. Если, предположим, массив сохранен в переменной `$customers`, код вывода каждого из его элементов будет выглядеть примерно так:

Слева в круглых скобках указывается имя массива.

По мере прохождения через каждый отдельный элемент массива значение этого элемента будет временно сохраняться в переменной с таким именем.

```
foreach ($customers as $customer) {
  echo $customer;
};
```

Находясь внутри цикла, вы можете получить доступ к каждому элементу массива, используя переменную, имя которой вы только что объявили.

Итак, если вы хотите пройти в цикле через все идентификаторы покупателей, сохраненные в массиве `todelete` суперглобального массива `$_POST`, можете создать следующий код с циклом `foreach`:

Это массив, сохраненный в суперглобальном массиве $_POST с идентификатором todelete.

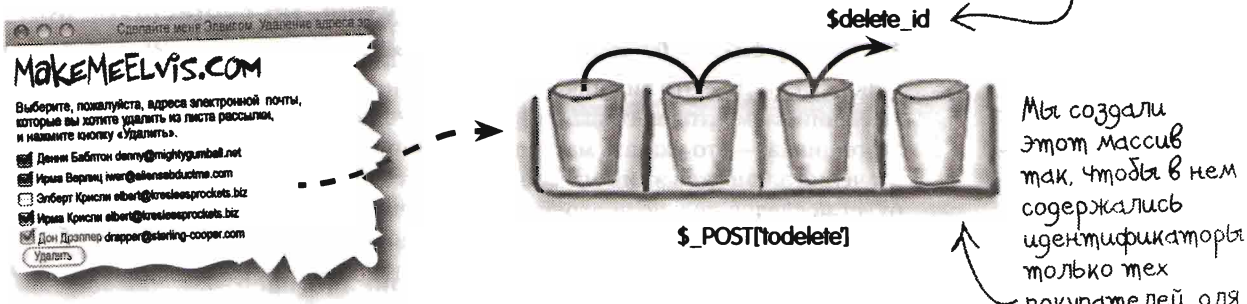
По мере прохождения через каждый отдельный элемент массива значение этого элемента будет доступно через переменную $delete_id.

```
foreach ($_POST['todelete'] as $delete_id) {
  // Удаление записи из таблицы
};
```

Находясь внутри цикла, вы можете удалить из таблицы запись любого покупателя.

Находясь внутри цикла, мы можем получить доступ к каждому идентификатору покупателя и удалить из таблицы его запись.

По мере прохождения цикла переменной `$delete_id` присваивается значение каждого элемента массива по одному за раз.



Мы создали этот массив так, чтобы в нем содержались идентификаторы только тех покупателей, для которых кнопка с независимой фиксацией была нажата (выбрана).

Теперь все, что нам необходимо, — это добавить внутри цикла `foreach` код, создающий запрос `DELETE` для каждого выбранного значения кнопки с зависимой фиксацией в форме «Удаление адреса электронной почты». Отправка запроса на сервер баз данных MySQL приведет к фактическому удалению соответствующей записи из таблицы `email_list`.

Закончите код нового, улучшенного сценария Элмера `removeemail.php` так, чтобы в результате его интерпретации отмеченные пользователи были удалены.

```
...
$dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
  or die('Ошибка соединения с MySQL-сервером.');

// Удаление записей покупателей (только в том случае,
// если форма была отправлена на сервер для выполнения).
if (..... ) {
  foreach ($_POST['todelete'] as $delete_id) {

    .....
  }

  echo 'Покупатель(ли) удален(ы).<br />';
}

// Ввод записей покупателей вместе с кнопками с независимой фиксацией
// для отметки удаляемых покупателей.
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($result)) {
  echo '<input type="checkbox" value="' . $row['id'] . '" name="todelete[]" />';
  echo $row['first_name'];
  echo ' ' . $row['last_name'];
  echo ' ' . $row['email'];
  echo '<br />';
}
mysqli_close($dbc);
?>
<input type="submit" name="_submit_" value="Удалить" />
</form>
```

removeemail.php

Решения к упражнению

Закончите код нового, улучшенного сценария Элмера `removeemail.php` так, чтобы в результате его интерпретации отмеченные пользователи были удалены.

```
...
$dbc = mysqli_connect('data.makemeelvis.com', 'elmer', 'theking', 'elvis_store')
  or die('Ошибка соединения с MySQL-сервером.');

// Удаление записей покупателей (только в том случае,
// если форма была отправлена на сервер для выполнения).
if (isset($_POST['submit'])) {
  foreach ($_POST['todelete'] as $delete_id) {
    $query = "DELETE FROM email_list WHERE id = $delete_id";
    mysqli_query($dbc, $query)
      or die('Ошибка запроса к базе данных.');
  }

  echo 'Покупатель(ли) удален(ы).<br />';
}

// Ввод записей покупателей вместе с кнопками с независимой фиксацией
// для отметки удаляемых покупателей.
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($result)) {
  echo '<input type="checkbox" value="' . $row['id'] . '" name="todelete[]" />';
  echo $row['first_name'];
  echo ' ' . $row['last_name'];
  echo ' ' . $row['email'];
  echo '<br />';
}
mysqli_close($dbc);
?>
  <input type="submit" name="_submit_" value="Удалить" />
</form>
```

Удаляйте покупателей только в том случае, если форма передана на сервер для исполнения!

Используйте значение переменной $delete_id для того, чтобы выбрать пользователя для удаления.

Код, генерирующий кнопки с независимой фиксацией, используемые для отметки кандидатов на удаление, тот же самый.

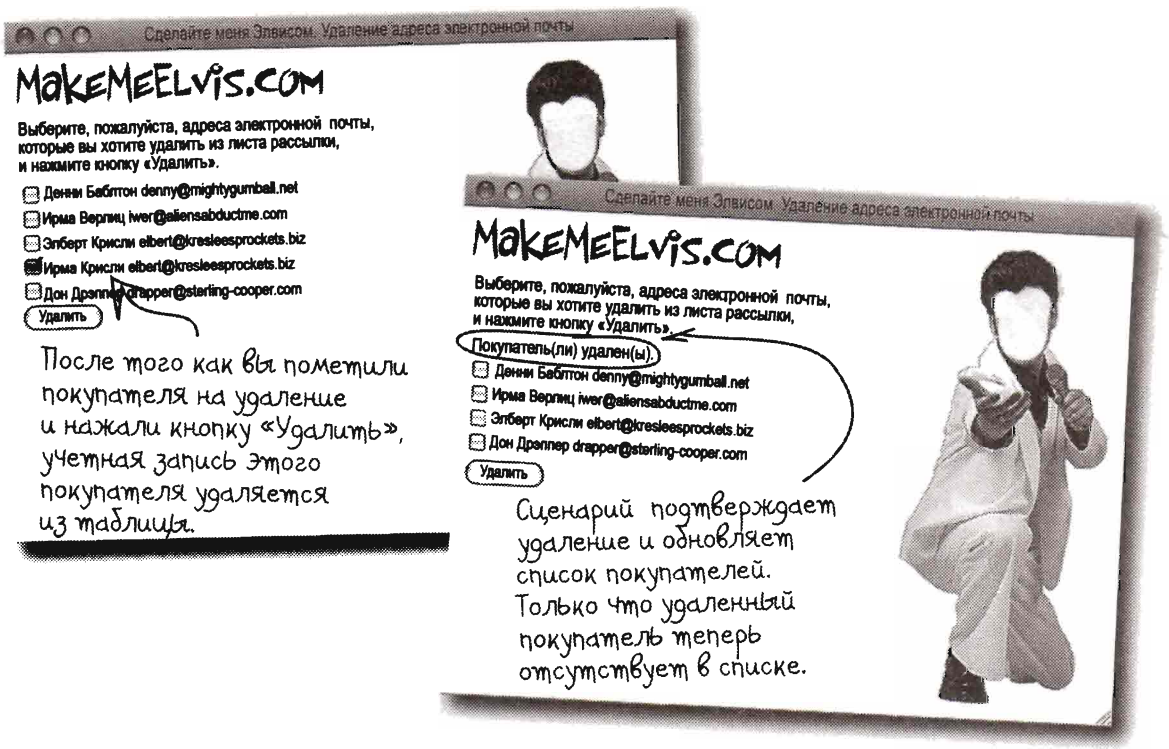
removeemail.php

Тест-драйв

Проверьте, как работает только что обновленный сценарий Элмера «Удаление адреса электронной почты».

Измените код сценария `removeemail.php` так, чтобы он генерировал кнопки с независимой фиксацией для выбора покупателя на удаление вместо старого текстового поля для ввода адреса электронной почты. Затем добавьте код для удаления учетных записей отмеченных покупателей после передачи формы на сервер для исполнения. Измените также значение атрибута `action` тега `<form>` так, чтобы форма ссылалась на себя.

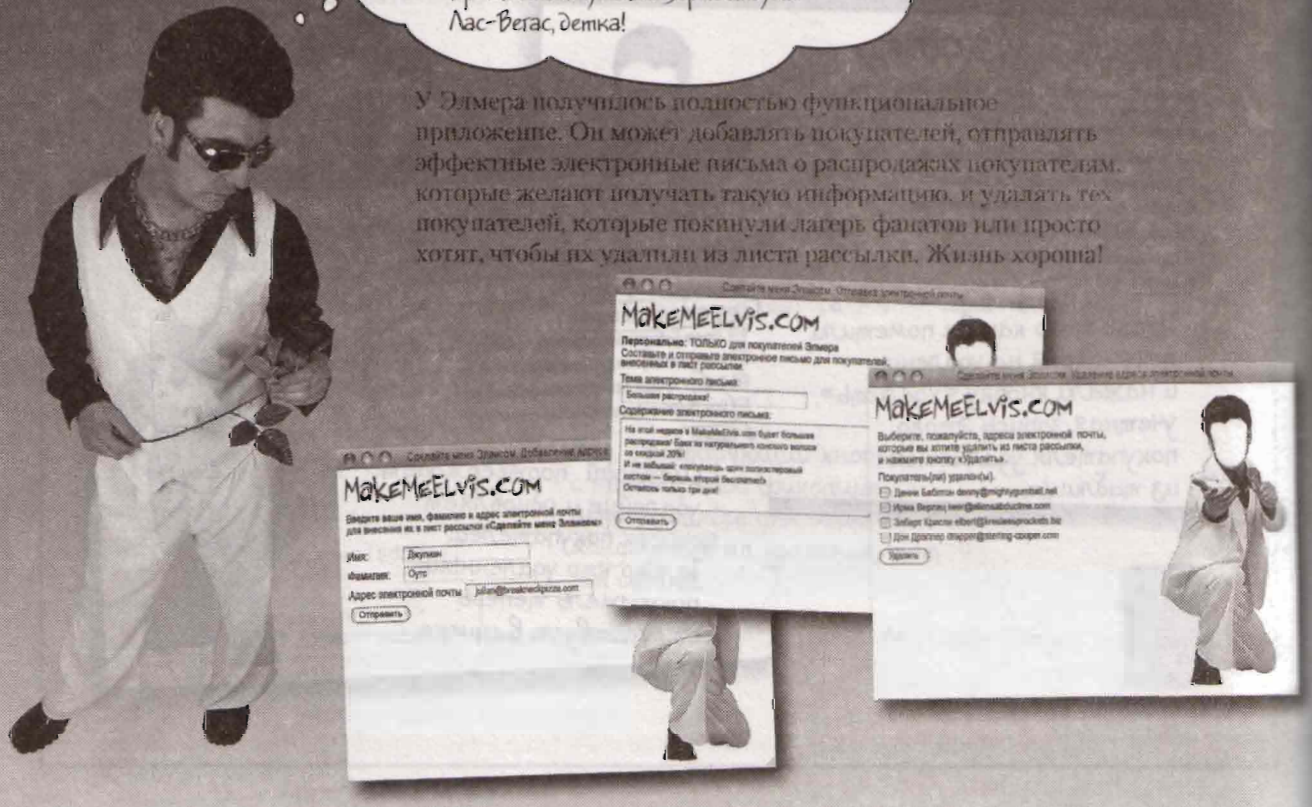
Теперь, после того как сценарий `removeemail.php` использует форму, ссылающуюся на себя, необходимость в присутствии на сервере файла `removeemail.html` отпадает и вы можете его удалить. Загрузите новую версию сценария `removeemail.php` на ваш веб-сервер и откройте его в браузере. Пометьте нескольких кандидатов на удаление и нажмите кнопку «Удалить». Форма должна немедленно измениться, отражая удаление покупателей.



ДОБРО ПОЖАЛОВАТЬ в Волшебный ЛАС-ВЕГАС НЕВАДА

Полностью перелопатил свою форму «Удаление адреса электронной почты». Время отдохнуть. Да здравствует Лас-Вегас, детка!

У Элмера получилось полностью функциональное приложение. Он может добавлять покупателей, отправлять эффектные электронные письма о распродажах покупателям, которые желают получать такую информацию, и удалять тех покупателей, которые покинули лагерь фанатов или просто хотят, чтобы их удалили из листа рассылки. Жизнь хороша!



Ваш инструментарий PHP и MySQL

Вы получили еще немного нового опыта в PHP и MySQL в процессе перевода веб-приложения Элмера на полностью новый уровень...

!

Оператор отрицания, или NOT, реверсирует значение переменных, содержащих значения true или false. При применении этого оператора к таким переменным значение true заменяется на false, и наоборот.

ALTER TABLE

В результате выполнения этого SQL-запроса изменяется структура таблицы. Например, может быть добавлена новая колонка. Это позволяет менять структуру таблицы без необходимости предварительно удалять ее и затем создавать заново.

if, else

Управляющая конструкция PHP if позволяет принимать решение, базирующееся на проверке истинности какого-либо выражения. Оно состоит из условного выражения, которое может принимать одно из двух значений: true (истина) или false (ложь), и блока кода, который выполняется в том случае, если условное выражение принимает значение true. Управляющая конструкция PHP if позволяет вам принимать различные виды решений. К этой конструкции может быть добавлено ключевое слово else, после которого следует альтернативный блок кода, который выполняется в том случае, если условное выражение принимает значение false.

foreach

Циклическая конструкция PHP, которая позволяет вам последовательно проходить по элементам массива без использования какого-либо условного выражения. Внутри этого цикла вы можете получить доступ к каждому элементу массива.

isset(), empty()

Встроенная PHP-функция isset() проверяет, существует ли переменная, переданная ей в качестве аргумента, или имеет ли она какое-либо значение. Функция empty() идет немного дальше и определяет, содержит ли переменная пустое значение (0 (ноль), пустую строку, false или NULL).

&&, OR

Это логические операторы. Они позволяют вам строить достаточно сложные выражения, включающие переменные, которые содержат значения true/false. Комбинация из двух переменных, объединенных логическим оператором && (AND), будет иметь значение true тогда и только тогда, когда обе переменные имеют значение true. Комбинация из двух переменных, объединенных логическим оператором || (OR), будет иметь значение true, когда хотя бы одна переменная имеет значение true.

==, <>, !=, <, >, ...

Операторы сравнения. Могут использоваться для конструирования условных выражений, в которых сравниваются различные значения. Они часто используются в условных выражениях управляющих конструкций if и циклах.

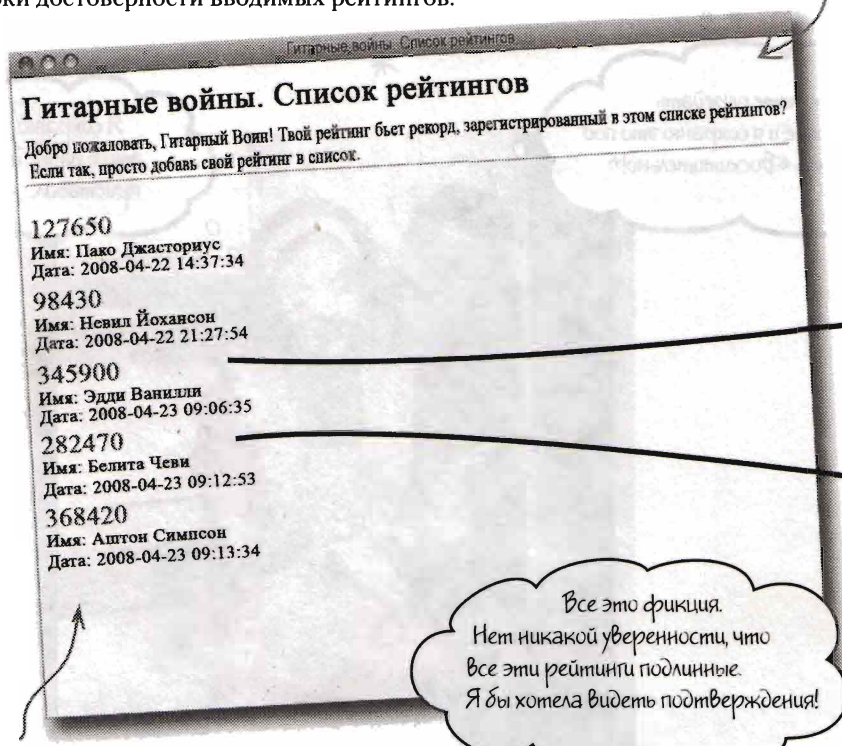
Когда просто базы данных недостаточно

Не верьте хвалебным речам... по поводу баз данных. Конечно, они исключительно удобны для сохранения различных видов данных, включая текст, но ***как обстоят дела с бинарными данными***? Скажем, такими как **изображения в формате JPEG или документы в формате PDF**. Есть ли смысл в том, чтобы хранить, например, картинки, изображающие коллекции редких гитарных медиаторов, в таблице базы данных? Чаще всего нет. Данные подобного типа обычно сохраняются в виде файлов, и будет лучше, если мы оставим их в этих файлах. В этой главе рассказывается, как **совместно использовать данные, сохраненные в файлах и базах данных, для того чтобы создавать PHP-приложения**, забитые под завязку бинарными данными.

Виртуальным гитаристам нравится соревноваться

Очевидно, что искусство для искусства не всегда устраивает, поэтому игроки новой популярной игры «Гитарные войны» без ума от соревнований виртуальных гитаристов. Они любят их до такой степени, что регулярно отправляют свои рейтинги на сайт «Гитарные войны», обязанности по техническому сопровождению которого возложены на вас. Имеется проблема, связанная с тем, что к настоящему времени не существует хорошего способа для проверки достоверности вводимых рейтингов.

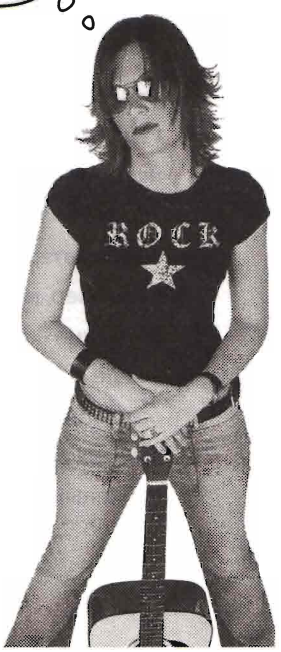
Приложение «Гитарные войны» позволяет пользователям самостоятельно добавлять свои рейтинги в список рейтингов.



Все это фикция. Нет никакой уверенности, что все эти рейтинги подлинные. Я бы хотела видеть подтверждения!

Не имея возможности проверить достоверность вводимых данных, мы не можем знать, чей рейтинг правдив, а чей — нет.

Белита, скептично настроенный рокер «Гитарных войн».

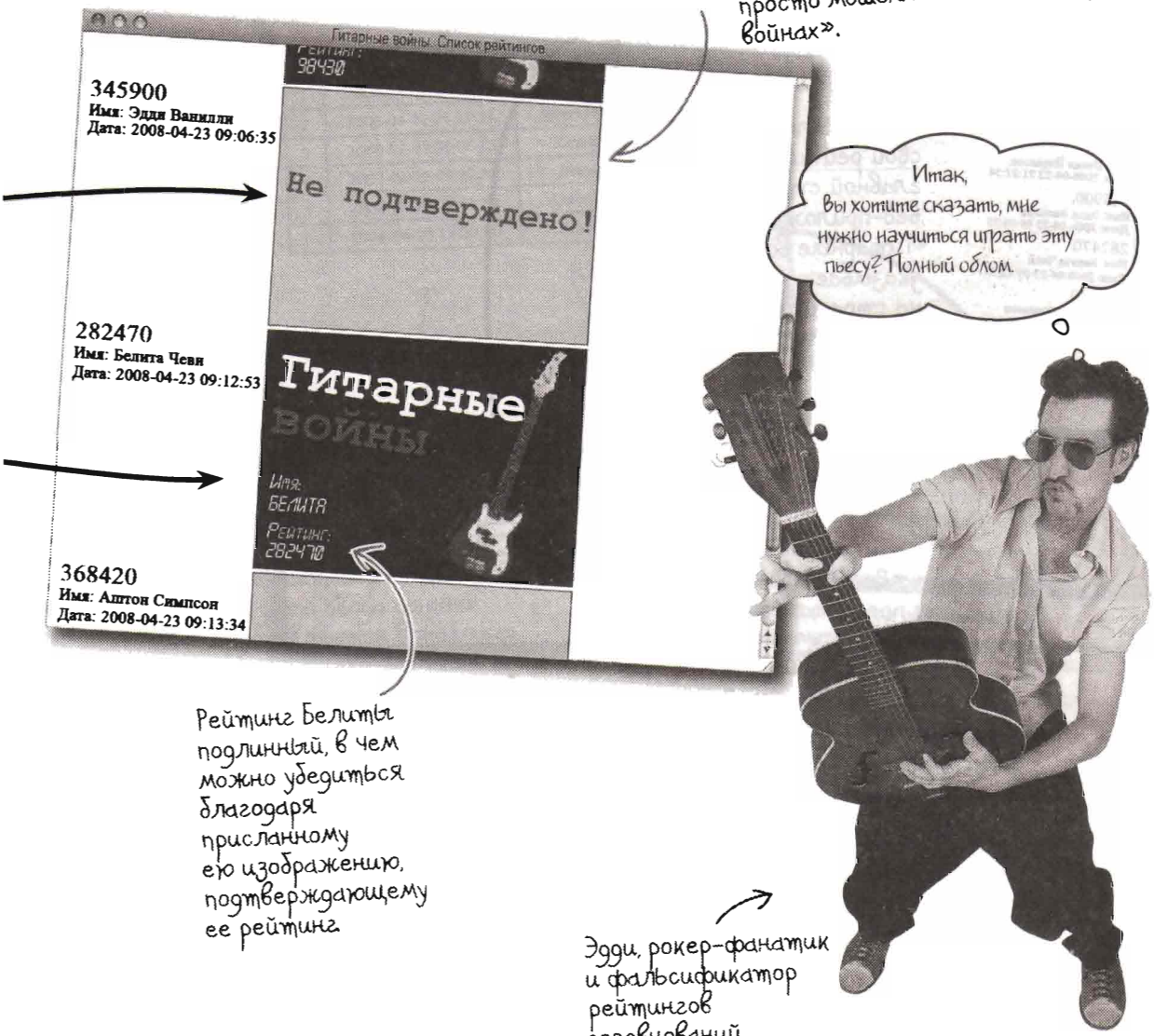


Текст можно подделать

Пока что исполнители просто пересылают свои рейтинги в виде простого текста, и разгорелось много споров о том, чьи рейтинги подлинные, а чьи — нет. Существует только один способ положить конец всем этим спорам и обеспечить легитимность чемпионату «Гитарные войны»...

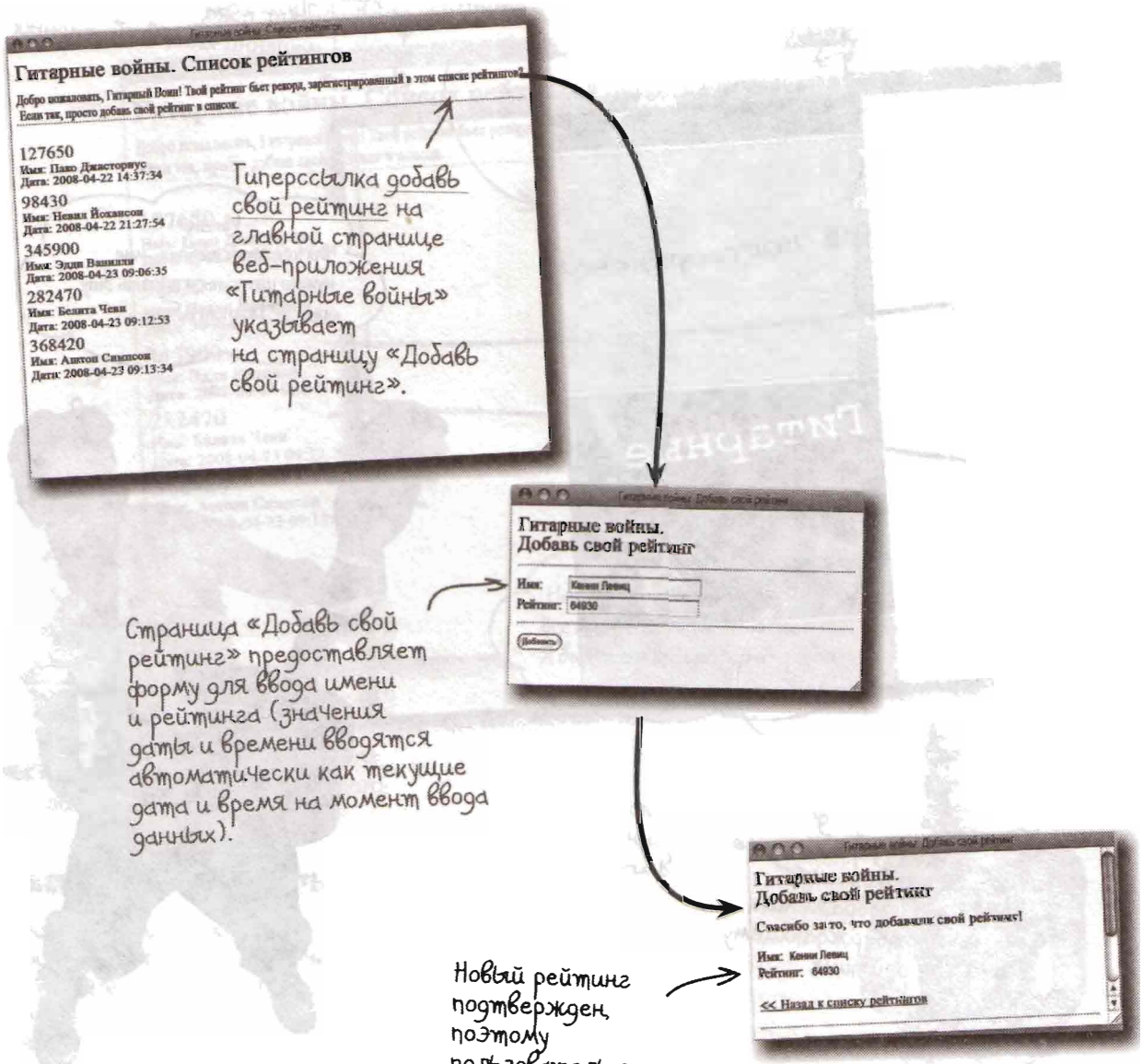
Доказательство подлинности — в изображении

Визуальное подтверждение высокого рейтинга — вот что нам необходимо, чтобы проверить, чей рейтинг подлинный, а чей — нет. Поэтому приложение «Гитарные войны» должно предоставить пользователям возможность загружать изображение, подтверждающее их рейтинг, когда они присылают свои рейтинги на сервер. Это значит, что список рейтингов будет содержать перечень не только рейтингов, имен и дат, но и изображений, подтверждающих подлинность этих данных.



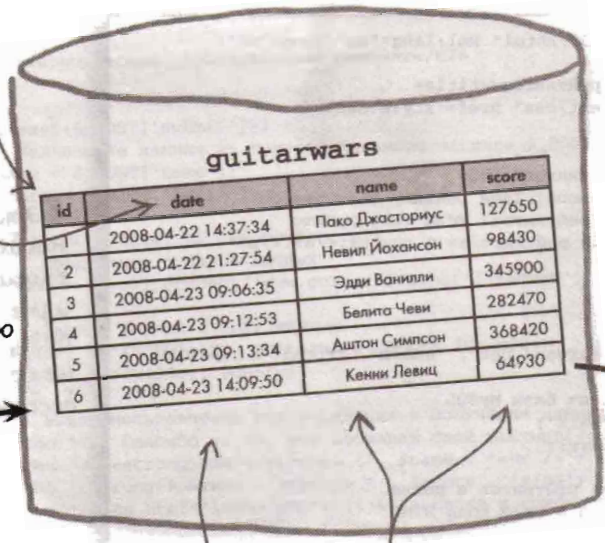
Приложению необходимо сохранять изображения

В настоящее время рейтинговое приложение «Гитарные войны» следит за тремя видами данных: датой и временем ввода нового рейтинга, именем человека, загрузившего этот рейтинг, и самим рейтингом. Информация вводится через форму, которая является частью пользовательского интерфейса этого приложения. После передачи формы на сервер все данные сохраняются в таблице с именем `guitarwars`.



Этот идентификатор является первичным ключом в таблице и автоматически генерируется для каждой записи.

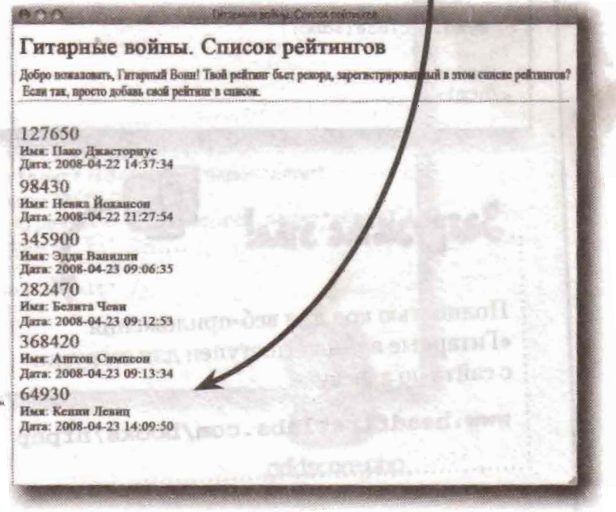
Это точная дата (и время), когда рейтинг был передан приложению «Гитарные войны».



id	date	name	score
1	2008-04-22 14:37:34	Пако Джасториус	127650
2	2008-04-22 21:27:54	Невил Йохансон	98430
3	2008-04-23 09:06:35	Эдди Ванилли	345900
4	2008-04-23 09:12:53	Белита Чеви	282470
5	2008-04-23 09:13:34	Аштон Симпсон	368420
6	2008-04-23 14:09:50	Кенни Левиц	64930

После ввода имени и рейтинга и нажатия кнопки «Добавить» новый рейтинг проходит проверку на достоверность, и дата ввода данных добавляется в таблицу guitarwars.

В таблице guitarwars также сохраняются имя и рейтинг для каждой записи.



Вновь занесенные рейтинги немедленно появляются на главной странице веб-приложения «Гитарные войны».

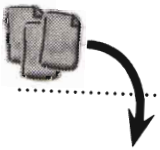
В приложение по сопровождению рейтингов «Гитарные войны» необходимо внести изменения, чтобы оно могло использовать загружаемые файлы изображений, подтверждающие подлинность рейтингов. Просмотрите и прокомментируйте ту часть приложения, которая должна отвечать за использование загружаемых пользователями файлов изображений.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
  <title>Гитарные войны. Список рейтингов</title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2> Гитарные войны. Список рейтингов </h2>
  <p> Добро пожаловать, гитарный воин! Твой рейтинг бьет рекорд,
зарегистрированный в этом списке рейтингов? Если так, просто
<a href="addscore.php">добавь свой рейтинг</a> в список.</a>.</p>
  <hr />

<?php
  // Соединение с базой данных
  $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');

  // Извлечение данных рейтингов из базы MySQL
  $query = "SELECT * FROM guitarwars";
  $data = mysqli_query($dbc, $query);

  // Извлечение данных из массива рейтингов в цикле.
  // Форматирование данных записей в виде кода HTML
  echo '<table>';
  while ($row = mysqli_fetch_array($data)) {
    // Вывод данных рейтинга
    echo '<tr><td class="scoreinfo">';
    echo '<span class="score">' . $row['score'] . '</span><br />';
    echo '<strong>Имя:</strong> ' . $row['name'] . '<br />';
    echo '<strong>Дата:</strong> ' . $row['date'] . '</td></tr>';
  }
  echo '</table>';

  mysqli_close($dbc);
?>
</body>
</html>
```

В этот файл нет необходимости вносить изменения, поэтому вы не должны о нем беспокоиться.

index.php

Загрузите это!



Полностью код для веб-приложения «Гитарные войны» доступен для загрузки с сайта по адресу

www.headfirstlabs.com/books/hfphp

guitarwars

id	date	name	score
1	2008-04-22 14:37:34	Пако Джисториус	127650
2	2008-04-22 21:27:54	Кевил Исхинсон	98430
3	2008-04-23 09:06:35	Эдди Ванилли	345900
4	2008-04-23 09:12:53	Бьелита Чеви	282470
5	2008-04-23 09:13:34	Аштон Симпсон	368420
6	2008-04-23 14:09:50	Кенни Левиц	64930

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
  <title>Гитарные войны. Добавьте свой рейтинг</title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2>Гитарные войны. Добавьте свой рейтинг</h2>

<?php
  if (isset($_POST['submit'])) {
    // Извлечение данных из суперглобального массива $_POST
    $name = $_POST['name'];
    $score = $_POST['score'];

    if (!empty($name) && !empty($score)) {
      // Соединение с базой данных
      $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');

      // Запись данных в базу данных
      $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score')";
      mysqli_query($dbc, $query);

      // Вывод пользователю подтверждения в получении данных
      echo '<p> Спасибо за то, что добавили свой рейтинг!</p>';
      echo '<p><strong>Имя:</strong> ' . $name . '<br />';
      echo '<strong>Рейтинг:</strong> ' . $score . '</p>';
      echo '<p><a href="index.php">&lt;&lt; Назад к списку рейтингов</a></p>';

      // Очистка полей ввода данных формы
      $name = "";
      $score = "";

      mysqli_close($dbc);
    }
    else {
      echo '<p class="error">'Введите, пожалуйста, всю информацию, необходимую ' .
        'для добавления вашего рейтинга.</p>';
    }
  }
?>

  <hr />
  <form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <label for="name">Имя:</label><input type="text" id="name" name="name"
      value="<?php if (!empty($name)) echo $name; ?>" /><br />
    <label for="score">Рейтинг:</label><input type="text" id="score" name="score"
      value="<?php if (!empty($score)) echo $score; ?>" />
    <hr />
    <input type="submit" value="Добавить" name="submit" />
  </form>
</body>
</html>
```

addscore.php

В приложение по сопровождению рейтингов «Гитарные войны» необходимо внести изменения, чтобы оно могло использовать загружаемые файлы изображений, подтверждающие подлинность рейтингов. Просмотрите и прокомментируйте ту часть приложения, которая должна отвечать за использование загружаемых пользователями файлов изображений.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
  <title>Гитарные войны. Список рейтингов</title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2> Гитарные войны. Список рейтингов </h2>
  <p>Добро пожаловать, гитарный воин! Твой рейтинг бьет рекорд, зарегистрированный
в этом списке рейтингов? Если так, просто <a href="addscore.php">добавь свой
рейтинг</a> в список.</p>
  <hr />

<?php
  // Соединение с базой данных
  $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit',

  // Извлечение данных рейтингов из базы MySQL
  $query = "SELECT * FROM guitarwars";
  $data = mysqli_query($dbc, $query);

  // Извлечение данных из массива рейтингов в цикле.
  // Форматирование данных записей в виде кода HTML

  echo '<table>';
  while ($row = mysqli_fetch_array($data)) {
    // Вывод данных рейтинга
    echo '<tr><td class="scoreinfo">';
    echo '<span class="score">' . $row['score'] . '</span><br />
    echo '<strong>Имя:</strong> ' . $row['name'] . '<br />';
    echo '<strong>Дата:</strong> ' . $row['date'] . '</td></tr>
  }
  echo '</table>';

  mysqli_close($dbc);
?>
</body>
</html>
```

Изображение должно быть выведено на главной странице

index.php

Изображение, подтверждающее рейтинг, должно быть получено через суперглобальный массив формы $_POST.

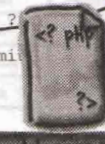
В результате выполнения SQL-запроса имя файла изображения должно быть внесено в таблицу guitarwars.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
  <title>Гитарные войны. Добавьте свой рейтинг</title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2>Гитарные войны. Добавьте свой рейтинг</h2>
<?php
  if (isset($_POST['submit'])) {
    // Извлечение данных из суперглобального массива $_POST
    $name = $_POST['name'];
    $score = $_POST['score'];
    if (!empty($name) && !empty($score)) {
      // Соединение с базой данных
      $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit',
'gwdb');
      // Запись данных в базу данных
      $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name',
'$score')";
      mysqli_query($dbc, $query);
      // Вывод пользователю подтверждения в получении данных
      // Вывод пользователю подтверждения в получении данных
      echo '<p> Спасибо за то, что добавили свой рейтинг!</p>';
      echo '<p><strong>Имя:</strong> ' . $name . '<br />';
      echo '<strong>Рейтинг:</strong> ' . $score . '</p>';
      echo '<p><a href="index.php">&lt;&lt; Назад к списку
рейтингов</a></p>';
      // Очистка полей ввода данных формы
      $name = "";
      $score = "";
      mysqli_close($dbc);
    }
    else {
      echo '<p class="error">Введите, пожалуйста, всю информацию
необходимую ' .
        'для добавления вашего рейтинга.</p>';
    }
  }
?>
  <hr />
  <form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <label for="name">Имя:</label><input type="text" id="name"
name="name"
      value="<?php if (!empty($name)) echo $name; ?>" /><br />
    <label for="score">Рейтинг:</label><input type="text" id="score"
name="score"
      value="<?php if (!empty($score)) echo $score; ?
    <hr />
    <input type="submit" value="Добавить" name="submi
  </form>
</body>
</html>
```

Вы должны убедиться, что имя файла изображения не пустое.

После успешного завершения убедитесь, что данные, запоминаемые формой, очищены.

Этот запрос выглядит довольно кратким, так как в нем не перечисляются имена колонок таблицы.



addscore.php

Изображение должно быть показано пользователю для подтверждения успешного завершения процесса.

guitarwars

id	date	name	score
1	2008-04-22 14:37:34	Пако Джасторимус	127650
2	2008-04-22 21:27:54	Невил Йохансон	98430
3	2008-04-23 09:06:35	Эдди Ванилли	345900
4	2008-04-23 09:12:53	Белита Чеви	282470
5	2008-04-23 09:13:34	Аштон Симпсон	368420
6	2008-04-23 14:09:50	Кенни Левиц	64930

Таблице необходима еще одна колонка для сохранения имени файла изображения.

Форме необходим еще тег <input> для поля ввода имени файла изображения.

Планирование загрузки изображений на сервер «Гитарные войны»

Хотя поддержка загрузки файлов может показаться не такой уж и сложной задачей, приложение должно быть изменено в ряде аспектов. Исходя из этого, составление плана операции перед углублением в ее детали выглядит достаточно разумной идеей. Давайте наполним эту идею содержанием, обозначив конкретные этапы, которые необходимо пройти, чтобы обеспечить загрузку файлов изображений для приложения «Гитарные войны».

1 Используйте запрос ALTER для того, чтобы добавить в таблицу колонку screenshot.

Во-первых, таблице необходима новая колонка для сохранения имен изображений, подтверждающих подлинность рейтинга. Так как мы предполагаем хранить все файлы на диске в одном каталоге, все, что нам необходимо сохранять в таблице, — это имя файла (а не его путь).

2 Измените форму «Добавь свой рейтинг», добавив еще одно поле для ввода загружаемого файла изображения.

В веб-странице «Добавь свой рейтинг» уже имеется форма для добавления рейтингов, поэтому вам необходимо модифицировать ее и добавить поле ввода имени файла. Оно должно согласовываться с браузером для предоставления пользователю возможности выбора файла для загрузки.

3 Напишите запрос INSERT, чтобы добавить имя файла изображения в колонку screenshot.

Сценарий «Добавь свой рейтинг», который обрабатывает данные формы по добавлению рейтинга, должен также принимать во внимание новое поле ввода и обеспечивать добавление имени файла изображения, подтверждающего подлинность рейтинга, в колонку `screenshot` таблицы `guitarwars`.

4 Измените главную страницу приложения «Гитарные войны» так, чтобы на ней появились изображения, подтверждающие подлинность рейтингов пользователей.

Последний пункт перечня изменений, которые необходимо внести в приложение, связан с файлом `index.php` страницы «Гитарные войны». Измените его так, чтобы на странице были выведены изображения, подтверждающие подлинность каждого рейтинга.

База данных рейтингов должна быть изменена

В дополнение ко всем изменениям, внесенным вами в PHP-код, для поддержки изображений приложению «Гитарные войны» в таблице `guitarwars` необходима новая колонка, в которой будут сохраняться имена файлов изображений, подтверждающих подлинность рейтингов. Составьте SQL-запрос, начинающийся с ключевого слова `ALTER` и позволяющий изменить таблицу во множестве интересных аспектов, включая добавление новой колонки. Вы уже использовали запрос ALTER TABLE в предыдущей главе для изменения таблицы `email_list`. Давайте кратко повторим, как работает этот запрос.

Запрос ALTER используется для изменения структуры базы данных.

За ключевым словом ALTER часто следует ключевое слово TABLE, которое указывает, что изменения будут касаться структуры таблицы. Имеется также возможность изменять структуру всей базы данных, используя в запросе ключевые слова ALTER DATABASE, но это уже другая история.

```
ALTER TABLE guitarwars DROP COLUMN score
```

Ключевые слова DROP COLUMN (удалить колонку) означают, что в результате выполнения запроса колонка будет полностью удалена из таблицы.

Возможно, это опасный пример, так как он показывает, как удалить колонку полностью, включая данные, сохраненные в ней. Тем не менее вполне возможно, что вы столкнетесь с ситуацией, когда у вас возникнет такая необходимость. Но чаще вы будете сталкиваться с необходимостью добавлять колонку, как в случае с приложением «Гитарные войны». Это делается путем добавления ключевых слов `ADD COLUMN`, что является одним из способов изменения структуры таблицы с помощью запроса ALTER.

ADD COLUMN (добавить колонку)

Добавляет новую колонку в таблицу. Просто укажите имя колонки и тип данных, которые будут в ней сохраняться, после ключевых слов `ADD COLUMN`.

```
ALTER TABLE guitarwars
ADD COLUMN age TINYINT
```

DROP COLUMN (удалить колонку)

Удаляет колонку (и все данные, сохраненные в ней) из таблицы. Просто укажите имя колонки после ключевых слов `DROP COLUMN`.

```
ALTER TABLE guitarwars
DROP COLUMN age
```

CHANGE COLUMN (изменить колонку)

Изменяет имя и тип данных колонки в таблице. Просто укажите старое имя колонки, ее новое имя и новый тип данных, которые будут в ней сохраняться, после ключевых слов `CHANGE COLUMN`.

```
ALTER TABLE guitarwars
CHANGE COLUMN score high_score INT
```

MODIFY COLUMN (модифицировать колонку)

Изменяет тип данных колонки и ее позицию в таблице. Просто укажите имя колонки и новый тип данных, которые будут в ней сохраняться, после ключевых слов `MODIFY COLUMN`. Для того чтобы изменить позицию колонки, укажите имя колонки и ее позицию: абсолютную (`FIRST` (первая) — единственный выбор при указании абсолютной позиции) или относительную (`AFTER` (после) другой существующей колонки, указанной по имени).

```
ALTER TABLE guitarwars
MODIFY COLUMN date DATETIME AFTER age
```

Время поработать

Напишите SQL-запрос, в результате выполнения которого в таблицу `guitarwars` будет добавлена новая колонка с именем `screenshot`. Убедитесь в том, что вы выбрали для новой колонки тип данных MySQL, соответствующий тому типу данных, которые будут в ней сохраняться. Затем напишите другой SQL-запрос, который позволит вам проверить структуру таблицы и убедиться, что колонка была добавлена успешно.

guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	

Напишите здесь SQL-запрос, в результате выполнения которого будет добавлена новая колонка.

.....

.....

Напишите второй SQL-запрос.

.....

Решение задачи

Напишите SQL-запрос, в результате выполнения которого в таблицу `guitarwars` будет добавлена новая колонка с именем `screenshot`. Убедитесь в том, что вы выбрали для новой колонки тип данных MySQL, соответствующий тому типу данных, которые будут в ней сохраняться. Затем напишите другой SQL-запрос, который позволит вам проверить структуру таблицы и убедиться, что колонка была добавлена успешно.

В результате выполнения запроса ALTER в таблицу guitarwars добавлена новая колонка screenshot.

guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	

Так как колонка новая, она имеет пустые значения для всех записей, которые были в таблице на момент ее добавления.

Выполнение запроса ALTER не оказывает никакого влияния на остальные данные таблицы.

Имя таблицы, структура которой должна быть изменена, следует непосредственно за ключевыми словами ALTER TABLE.

```
ALTER TABLE guitarwars
ADD COLUMN screenshot varchar(64)
```

Ключевые слова ADD COLUMN говорят о том, что мы хотим изменить структуру таблицы путем добавления новой колонки.

Имя колонки и тип данных, которые будут в ней сохраняться, указаны в самом конце SQL-запроса. 64 символов вполне достаточно, чтобы сохранить большинство имен файлов, хотя вы можете сделать длину колонки больше, если хотите быть более уверенными.

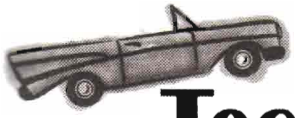
```
DESCRIBE guitarwars
```

В результате выполнения этого запроса будет показана структура таблицы, включая имена колонок и типы данных, которые в них сохраняются.

Первый этап закончен!

СДЕЛАНО

1. **Используйте запрос ALTER для того, чтобы добавить в таблицу колонку** `screenshot`.



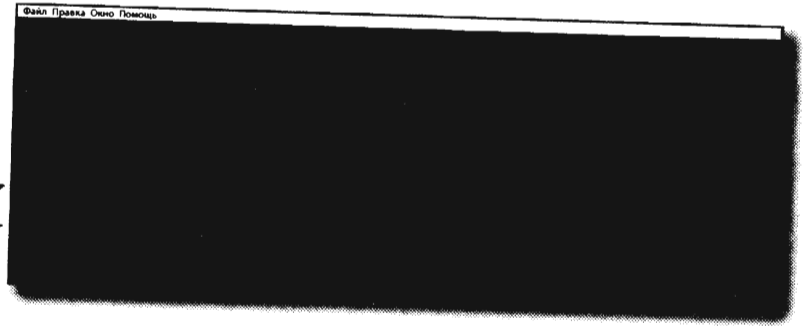
Тест-драйв

Добавьте колонку `screenshot` в таблицу `guitarwars`.

Используя инструментальную программу MySQL, выполните запрос ALTER, чтобы добавить колонку `screenshot` в таблицу `guitarwars`. Затем выполните запрос DESCRIBE, чтобы просмотреть структуру таблицы и убедиться, что колонка добавлена.

Вы можете сначала создать таблицу guitarwars, загрузив примеры кода для приложения «Гитарные войны», и затем выполнить SQL-запрос из файла guitarwars.sql.

В результате выполнения запроса DESCRIBE показана новая колонка.



Не бывает глупых вопросов

В: Новые колонки, добавляемые с помощью запроса ALTER, должны помещаться в конец таблицы?

О: Нет, они могут помещаться куда угодно. Но имейте в виду, что порядок расположения колонок не имеет большого значения. Иначе говоря, вы можете создавать свои запросы так, что данные будут выводиться в любом порядке, который вам необходим. Но, возможно, вы любите порядок во всем, в том числе и в расположении колонок в таблице. В этом случае можете добавлять колонки в определенное место в структуре таблицы. Вы добьетесь этого, используя в запросе ALTER ключевое слово `FIRST` или `AFTER`, чтобы разместить новую колонку относительно уже существующей:

```
ALTER TABLE guitarwars
ADD COLUMN age TINYINT AFTER name.
```

Если вы не укажете, где должна быть размещена новая колонка, по умолчанию она будет добавлена последней в таблице.

В: Что произойдет с существующими записями данных рейтингов после добавления новой колонки?

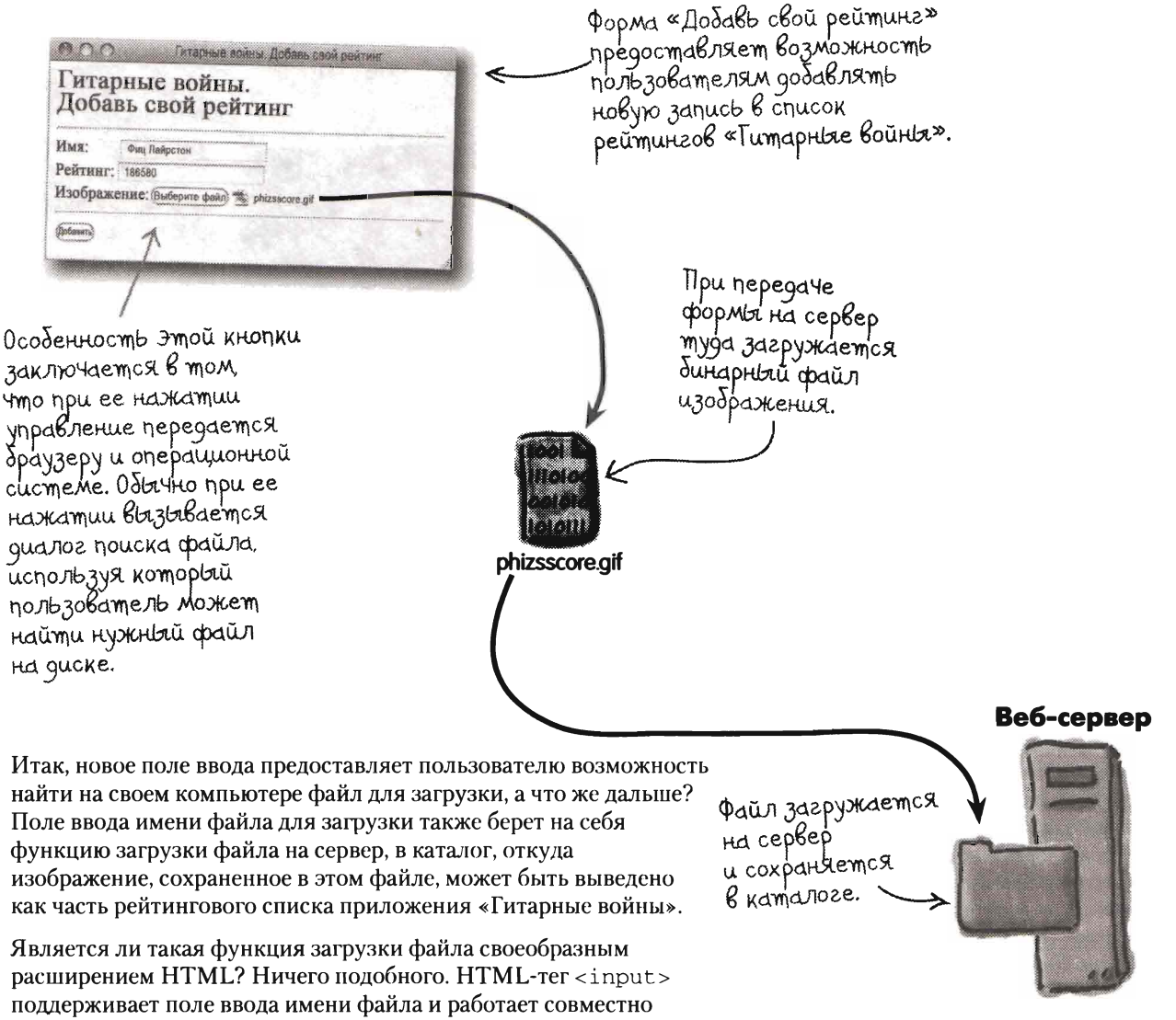
О: Так как запрос ALTER оказывает влияние только на структуру таблицы, новая колонка `screenshot` будет иметь пустое значение для всех записей, которые были в таблице на момент ее добавления. Хотя имеется возможность добавлять данные во все новые записи, значения колонки `screenshot` для всех предыдущих записей будут пустыми.

В: Существует ли способ добавить имена файлов изображений в колонку screenshot предыдущих записей?

О: Да. Это, безусловно, может быть сделано, для чего вам придется воспользоваться SQL-запросом UPDATE. Нет никаких препятствий для того, чтобы вручную загрузить файлы изображений на веб-сервер и затем, воспользовавшись SQL-запросом UPDATE, внести имена файлов изображений в уже существующие записи. Но не забывайте, что главная идея здесь заключается в том, чтобы предоставить пользователям возможность самостоятельно загружать свои собственные файлы изображений. И они смогут делать это, используя усовершенствованный сценарий «Добавь свой рейтинг», который вы сейчас создаете…

Как получить файл изображения от пользователя?

После добавления к таблице новой колонки мы готовы сосредоточиться на задаче загрузки пользователем файла изображения. Но как это сделать? С помощью FTP? Телепатии? Действительность возвращает нас к форме «Добавь свой рейтинг», в которой мы можем использовать поле ввода, чтобы дать пользователю возможность выбрать имя файла для загрузки.



Итак, новое поле ввода предоставляет пользователю возможность найти на своем компьютере файл для загрузки, а что же дальше? Поле ввода имени файла для загрузки также берет на себя функцию загрузки файла на сервер, в каталог, откуда изображение, сохраненное в этом файле, может быть выведено как часть рейтингового списка приложения «Гитарные войны».

Является ли такая функция загрузки файла своеобразным расширением HTML? Ничего подобного. HTML-тег `<input>` поддерживает поле ввода имени файла и работает совместно с PHP, позволяя загружать указанный в поле ввода файл. Но прежде чем мы перейдем к стороне дела, связанной с PHP, давайте посмотрим повнимательней на само поле ввода имени файла...

Форма «Добавь свой рейтинг» в деталях

Этот атрибут формы говорит ей использовать при загрузке изображений специальное кодирование данных. От этого зависит, как данные, передаваемые методом POST, будут упакованы для передачи на сервер.

Устанавливает максимальный размер файла для загрузки. В данном случае 32КВ (32 768 байт).

Это форма, ссылающаяся на себя.

```
<form enctype="multipart/form-data" method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
  <input type="hidden" name="MAX_FILE_SIZE" value="32768" />
  <label for="name">Имя:</label>
  <input type="text" id="name" name="name" value="<?php if (!empty($name)) echo $name; ?>" />
  <br />
  <label for="score">Рейтинг:</label>
  <input type="text" id="score" name="score" value="<?php if (!empty($score)) echo $score; ?>" />
  <br />
  <label for="screenshot">Файл изображения:</label>
  <input type="file" id="screenshot" name="screenshot" />
  <hr />
  <input type="submit" value="Добавить" name="submit" />
</form>
```

Это поле ввода имени файла, полностью полагающееся на диалог операционной системы, который позволяет пользователю находить и выбирать нужный файл на диске.

СДЕЛАНО

2 **Измените форму «Добавь свой рейтинг», добавив еще одно поле для ввода загружаемого файла изображения.**

имени файла

Добавление изображения в базу данных

Имя файла изображения используется как часть запроса INSERT при сохранении в базе данных.

Простой загрузки файла изображения на веб-сервер с использованием формы недостаточно. Нам также необходимо сохранить имя файла в новой колонке `screenshot`, чтобы к изображению можно было получить доступ и вывести его на экран. Как мы уже говорили, сценарий «Добавь свой рейтинг» добавляет записи с данными рейтингов в таблицу `guitarwars`, используя SQL-запрос `INSERT`, но в нем не принимает никакого участия новая колонка `screenshot`:

```
INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score')
```

Значение колонки id устанавливается автоматически через механизм AUTO_INCREMENT, значение 0 игнорируется. Оно указывается здесь потому, что форма запроса требует наличия какого-либо значения в этом месте.

MySQL-функция NOW() используется для добавления текущих даты и времени.

Так как в этом SQL-запросе значения данных перечислены без указания *имен колонок, в нем должны использоваться данные для всех колонок.* Но мы только что добавили новую колонку, что означает, что запрос больше работать не будет: в нем нет значения для колонки `screenshot`. Поэтому добавление к таблице имени файла изображения как части новой записи рейтинга требует добавления еще одного значения в запрос `INSERT`:

Добавление имени файла в запрос INSERT помещает его значение в базу данных.

```
INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')
```

Записи, которые были в таблице на момент добавления колонки screenshot, имеют пустые значения.

Порядок, в котором перечисляются эти значения, очень важен, поскольку при выполнении запроса считается, что данные следуют в том же порядке, что и колонки в структуре таблицы.

guitarwars

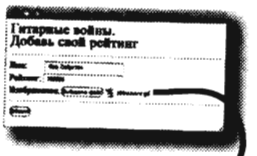
id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif

В результате выполнения запроса INSERT в колонку screenshot новой записи добавляется имя файла изображения.

СДЕЛАНО

3 **Напишите запрос INSERT, чтобы добавить имя файла изображения в колонку** `screenshot`.

Определение имени загружаемого файла



Запрос выглядит неплохо, но мы до сих пор не знаем, какое же имя имеет файл изображения. Вполне можно предположить, что поле ввода формы как-то предоставляет доступ к имени файла, но как? Ответ связан со встроенной суперглобальной переменной `$_FILES`, во многом похожей на суперглобальную переменную `$_POST`, которую мы использовали для получения доступа к данным формы. Как и переменная `$_POST`, переменная `$_FILES` является массивом, и его элементы содержат не только имя загружаемого файла, но и ряд других сведений о нем, которые могут оказаться весьма полезными.

```
<input type="file" name="screenshot" />
```

При поступлении на сервер форма содержит некоторую полезную для PHP-сценария информацию о загружаемом файле в суперглобальном массиве $_FILES.

phizsscore.gif

Это файл изображения, который загружается благодаря полю ввода имени файла на форме.

```
$_FILES['screenshot']['name']
```

phizsscore.gif ← Имя загружаемого файла.

```
$_FILES['screenshot']['type']
```

image/gif ← MIME тип загружаемого файла, в данном случае GIF.

```
$_FILES['screenshot']['size']
```

12244 ← Размер загружаемого файла (в байтах).

```
$_FILES['screenshot']['tmp_name']
```

/tmp/phpE7qJky ← Полное имя временного файла, под которым сохранен на сервере загружаемый файл.

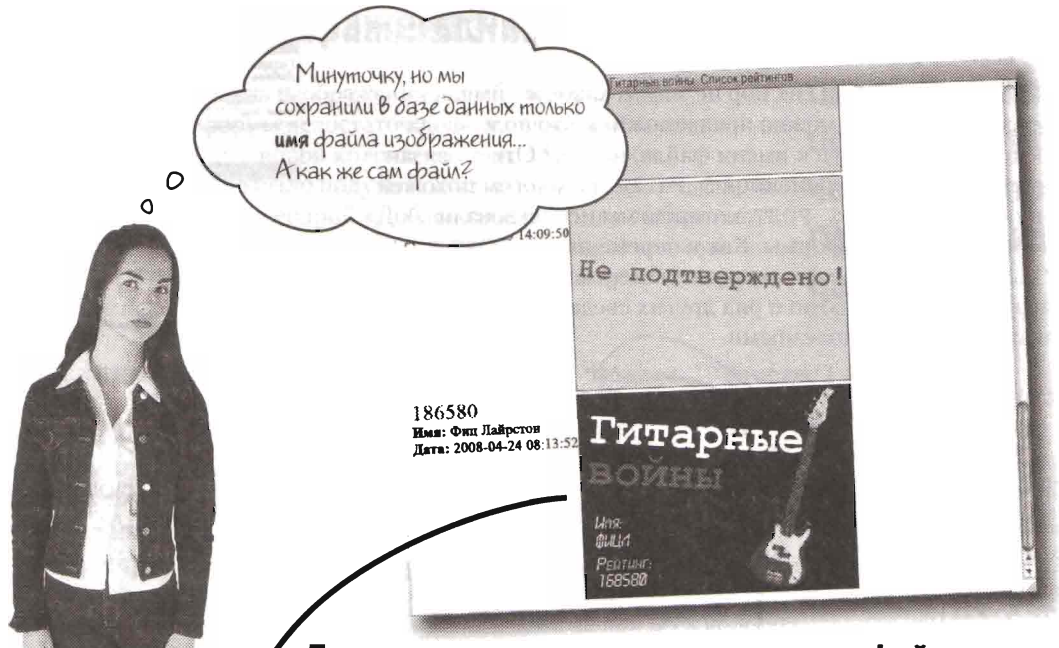
```
$_FILES['screenshot']['error']
```

0 ← В случае успешной загрузки этот элемент массива принимает значение 0 (ноль). Если же загрузка файла не удалась — код ошибки.

Суперглобальный массив $_FILES позволяет получить доступ к информации о загружаемом файле.

Вся информация, содержащаяся в суперглобальном массиве `$_FILES`, безусловно, очень важна, но в данный момент нам необходимо только имя файла изображения, которое может быть сохранено в локальной переменной (`$screenshot`) и использовано в SQL-запросе `INSERT`.

```
$screenshot = $_FILES['screenshot']['name'];
```



Данные, содержащиеся во внешних файлах, обычно так и остаются в них, даже в приложениях, использующих базы данных.

В этом случае данные — это набор пикселов, из которых состоит изображение. Он сохранен во внешнем файле в формате `GIF`, `JPEG` или `PNG` (существует множество других форматов). Хотя во многих базах данных (в том числе в MySQL) предусмотрена возможность сохранять бинарные данные (например, изображения) в полях типа BLOB (Binary Large Object, то есть «большие бинарные объекты»), операции с данными типа INT, CHAR, VARCHAR и другими значительно эффективнее. Поэтому в базах данных часто гораздо удобнее и эффективнее сохранять не сами изображения, а ссылки на содержащие их файлы. Такими ссылками являются имена их файлов.

Еще одной причиной, почему в веб-приложениях изображения не сохраняются в базе данных, является то, что в этом случае было бы труднее кодировать в HTML вывод их на экран. Вспомните, что HTML ссылается на изображения через имя внешнего файла, в котором это изображение сохранено. Поэтому в HTML-теге, выводящем на экран изображение, используется имя файла изображения, а не сами изображения.

Для вывода изображения на веб-страницу достаточно только ссылки на файл изображения, сохраненный на сервере.

```
<img src="phizsscore.gif" alt="Изображение,
подтверждающее подлинность рейтинга" />
```

Имя файла изображения.

HTML-тег <img> использует имя файла в качестве ссылки на файл изображения, сохраненный на сервере.

Время поработать

На главной странице приложения «Гитарные войны» (index.php) все еще не выводятся изображения, подтверждающие подлинность рейтингов. Закончите код, чтобы они выводились.

```
<?php
 // Соединение с базой данных
 $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');

 // Извлечение данных рейтингов из базы MySQL
 $query = ..... ;
 $data = mysqli_query($dbc, $query);

 // Прохождение в цикле по массиву записей рейтингов.
 // Форматирование данных записей в виде HTML-кода.
 echo '<table>';
 while ($row = mysqli_fetch_array($data)) {
  // Вывод данных рейтингов
  echo '<tr><td class="scoreinfo">';
  echo '<span class="score">' . $row['score'] . '</span><br />';
  echo '<strong>Имя:</strong> ' . $row['name'] . '<br />';
  echo '<strong>Дата:</strong> ' . $row['date'] . '</td>';
  if (is_file(.....) && filesize(.....) > 0) {
   echo '<td><img src="' . ..... . '" alt="Подтверждено" /></td></tr>';
  }
  else {
   echo '<td><img src="unverified.gif" alt="Не подтверждено!" /></td></tr>';
  }
 }
 echo '</table>';

 mysqli_close($dbc);
?>
```

Решение задачи

На главной странице приложения «Гитарные войны» (index.php) все еще не выводятся изображения, подтверждающие подлинность рейтингов. Закончите код, чтобы они выводились.

Для упрощения кода мы не используем здесь выражение or die(), которое выводит сообщение об ошибке в случае, если выполнение функции не удастся. Возможно, вы захотите добавить его при включении этого кода в свое приложение, но с данного момента мы будем пропускать его для краткости.

```
<?php
 // Соединение с базой данных
 $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');

 // Извлечение данных рейтингов из базы MySQL
 $query = "SELECT * FROM guitarwars";
 $data = mysqli_query($dbc, $query);

 // Прохождение в цикле по массиву записей рейтингов.
 // Форматирование данных записей в виде HTML-кода.
 echo '<table>';
 while ($row = mysqli_fetch_array($data)) {
  // Вывод данных рейтингов
  echo '<tr><td class="scoreinfo">';
  echo '<span class="score">' . $row['score'] . '</span><br />';
  echo '<strong>Имя:</strong> ' . $row['name'] . '<br />';
  echo '<strong>Дата:</strong> ' . $row['date'] . '</td>';
  if (is_file( $row['screenshot'] ) && filesize( $row['screenshot'] ) > 0) {
   echo '<td><img src="' . $row['screenshot'] . '" alt="Подтверждено" /></td></tr>';
  }
  else {
   echo '<td><img src="unverified.gif" alt="Не подтверждено!" /></td></tr>';
  }
 }
 echo '</table>';

 mysqli_close($dbc);
?>
```

SQL-запрос, в результате выполнения которого извлекаются данные рейтингов, совершенно не изменился!

Эта функция проверяет, что в файле изображения содержатся данные.

В таблице в колонке screenshot сохранено имя файла изображения, подтверждающего подлинность данного рейтинга.

Эта функция проверяет, существует ли файл изображения, подтверждающий подлинность рейтинга.

СДЕЛАНО

4 **Измените главную страницу приложения «Гитарные войны» так, чтобы на ней появились изображения, подтверждающие подлинность рейтингов пользователей.**

Тест-драйв

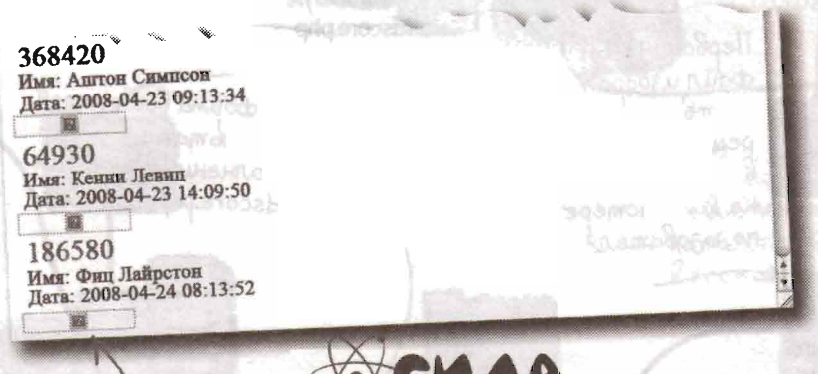
Добавьте новый рейтинг в приложение «Гитарные войны», которое уже выводит изображения, подтверждающие подлинность рейтингов.

Если вы еще не сделали этого, загрузите примеры кода для веб-приложения «Гитарные войны» с сайта по адресу `www.headfirstlabs.com/books/hfphp`. Этот код находится в каталоге `chapter05` и включает главную веб-страницу (`index.php`), сценарий «Добавь свой рейтинг» (`addscore.php`) и каскадные таблицы стилей (`style.css`).

Первое, что вы должны сделать, — это внести изменения в сценарий `addscore.php`, которые обеспечат форме «Добавь свой рейтинг» поддержку загрузки на сервер файлов изображений, подтверждающих рейтинги. Это включает добавление в форму новых полей ввода данных, корректировку тега `<form>` и проверку того, что переменная `$screenshot` имеет какое-либо непустое значение. Затем необходимо добавить в сценарий SQL-запрос `INSERT`, добавляющий новые записи рейтингов.

Теперь нужно перейти к сценарию `index.php` и добавить новый код, описанный на предыдущей странице, чтобы при открытии сценария на экран выводились изображения, подтверждающие подлинность рейтингов для всех записей.

Загрузите все эти файлы на ваш веб-сервер и откройте сценарий `addscore.php` в браузере. Введите новый рейтинг и нажмите кнопку «Добавить». Затем перейдите к странице `index.php` и проверьте новый рейтинг



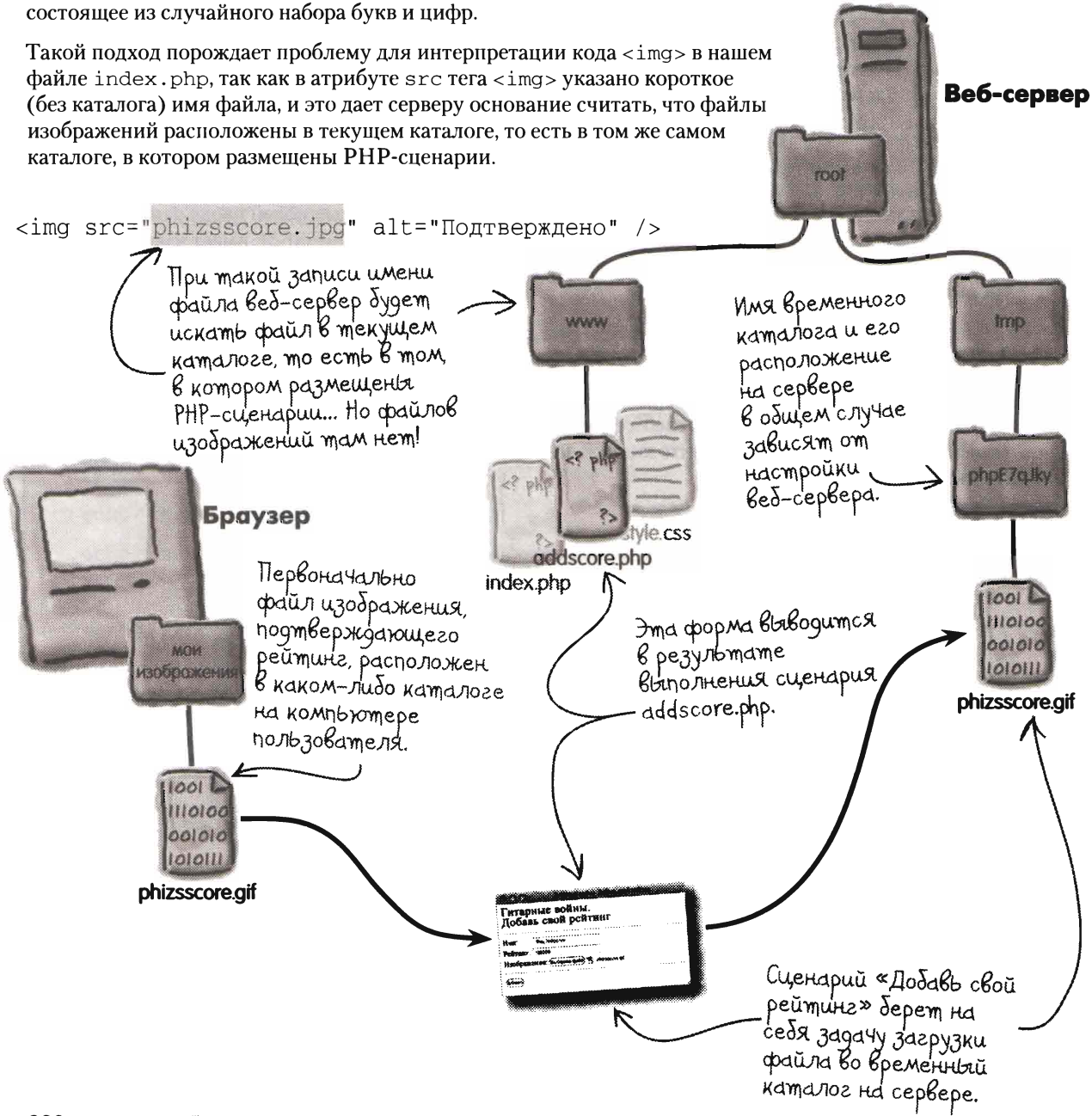
Что-то не то! Изображения, подтверждающие подлинность рейтингов, не выводятся для новой записи, как должны.

Почему, по вашему мнению, изображения, подтверждающие подлинность рейтингов, не выводятся для новой записи? И что вы можете сказать по поводу рейтингов, которые уже были в базе данных на момент корректировки приложения?

Куда деваются загружаемые на сервер файлы?

Проблема, связанная с файлами, загруженными на сервер, заключается в том, что мы предполагаем, что файлы должны сохраняться в тот же каталог на веб-сервере, в котором размещены наши PHP-сценарии. Оказывается, это совершенно неверное предположение. Форма «Добавь свой рейтинг» дает возможность пользователю выбрать файл на своем собственном компьютере, но загруженный файл сохраняется во временном каталоге. Он создается на сервере автоматически и обычно имеет бессмысленное имя, состоящее из случайного набора букв и цифр.

Такой подход порождает проблему для интерпретации кода `<img>` в нашем файле `index.php`, так как в атрибуте `src` тега `<img>` указано короткое (без каталога) имя файла, и это дает серверу основание считать, что файлы изображений расположены в текущем каталоге, то есть в том же самом каталоге, в котором размещены PHP-сценарии.

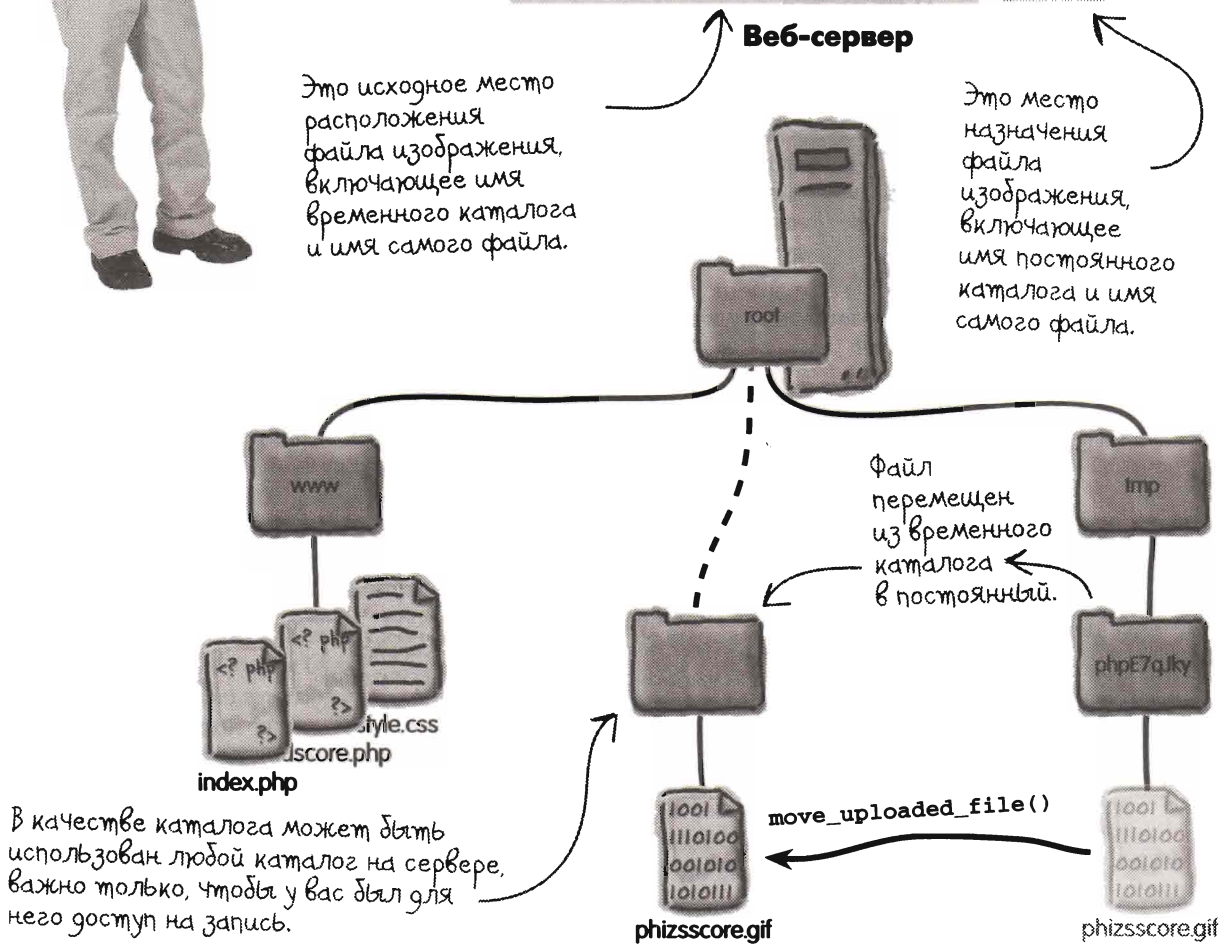


Сохранение файлов изображений во временных каталогах с непонятными именами создает ненужную путаницу. Можем ли мы контролировать процесс сохранения файлов?

Да! PHP предоставляет вам возможность контролировать процесс сохранения файлов.

Тем не менее вы не можете контролировать расположение каталога для первоначального сохранения загружаемых файлов. Но вы можете переместить файл в другое место, после того как процесс загрузки завершился. PHP-функция `move_uploaded_file()`, используя исходное место расположения файла и место его назначения, выполняет задачу по его перемещению:

```
move_uploaded_file($_FILES['screenshot']['tmp_name'], $target);
```



не бывает глупых вопросов

В: Разве я не могу изменить первоначальное место расположения загружаемых файлов, изменяя файл php.ini?

О: Да. Инициализационный файл PHP (`php.ini`) может быть использован для изменения первоначального места расположения загружаемых файлов через его опцию `upload_tmp_dir`. Но если вы арендуете виртуальный сервер у хостинговой компании, у вас может не быть доступа с правом записи к этому файлу, что означает, что вы все же должны будете переместить файл в свой собственный каталог с помощью сценария PHP.

В: Почему первоначальный каталог для загружаемых файлов называется «временный»? Он удаляется после того, как файл перемещен из него?

О: Нет. Каталог называется «временным» в том смысле, что он не рассматривается как окончательное место расположения загружаемых файлов. Вы можете рассматривать его как область промежуточного хранения, где загружаемые файлы хранятся до тех пор, пока не будут перемещены на их постоянное место.

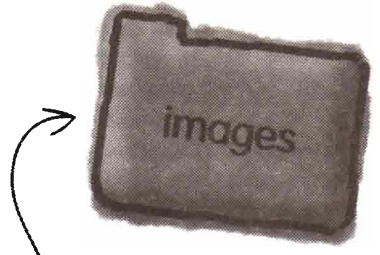
В: Почему я не могу просто оставить файл во временном каталоге?

О: Вы можете это сделать. В этом случае необходимо добавить значение переменной `$_FILES['screenshot']['tmp_name']` к имени файла изображения, чтобы к нему был доступ во временном каталоге. Но имейте в виду, что в общем случае у вас не будет полного контроля над именем и расположением каталога. И, что более важно, в некоторых системах временные каталоги могут периодически очищаться. Другая потенциальная проблема заключается в том, что временные каталоги могут не быть доступными для всех, поэтому вы не сможете ссылаться на них из HTML-кода, на чем базируется веб-приложение «Гитарные войны» и большинство других PHP-приложений. Перемещая файлы из временного каталога, вы получаете полный контроль над тем, где конкретно они будут размещаться и с каким уровнем доступа.

Каждому приложению необходим каталог images.

Возможно, необходим — слишком громко сказано, но очень важно организовывать все составные части своего PHP-приложения настолько тщательно, насколько это возможно, так как загрузка файлов изображений осуществляется пользователем, а пользователи не относятся к тем лицам, чьи действия вы можете контролировать напрямую, по крайней мере, что касается имен файлов и их количества. Поэтому очень хорошо сохранять их отдельно от других файлов приложения.

После всего сказанного становится очевидным, что нам необходим каталог `images`, в котором будут храниться файлы изображений, загруженные для приложения «Гитарные войны». Этот каталог при необходимости может также быть местом расположения других изображений, используемых приложением.



Фактически каталог images не больше любого другого каталога, но он помогает в организации приложения, собирая все файлы изображений в одном месте.

Создание места для загруженных файлов изображений

Каталог `images` почти ничем не отличается от любого другого каталога на веб-сервере, за исключением того, что он должен быть размещен в главном каталоге веб-приложения или в одном из его подкаталогов на веб-сервере. Обычно его создают как подкаталог главного веб-каталога приложения, но при желании вы можете создать более сложную иерархию.

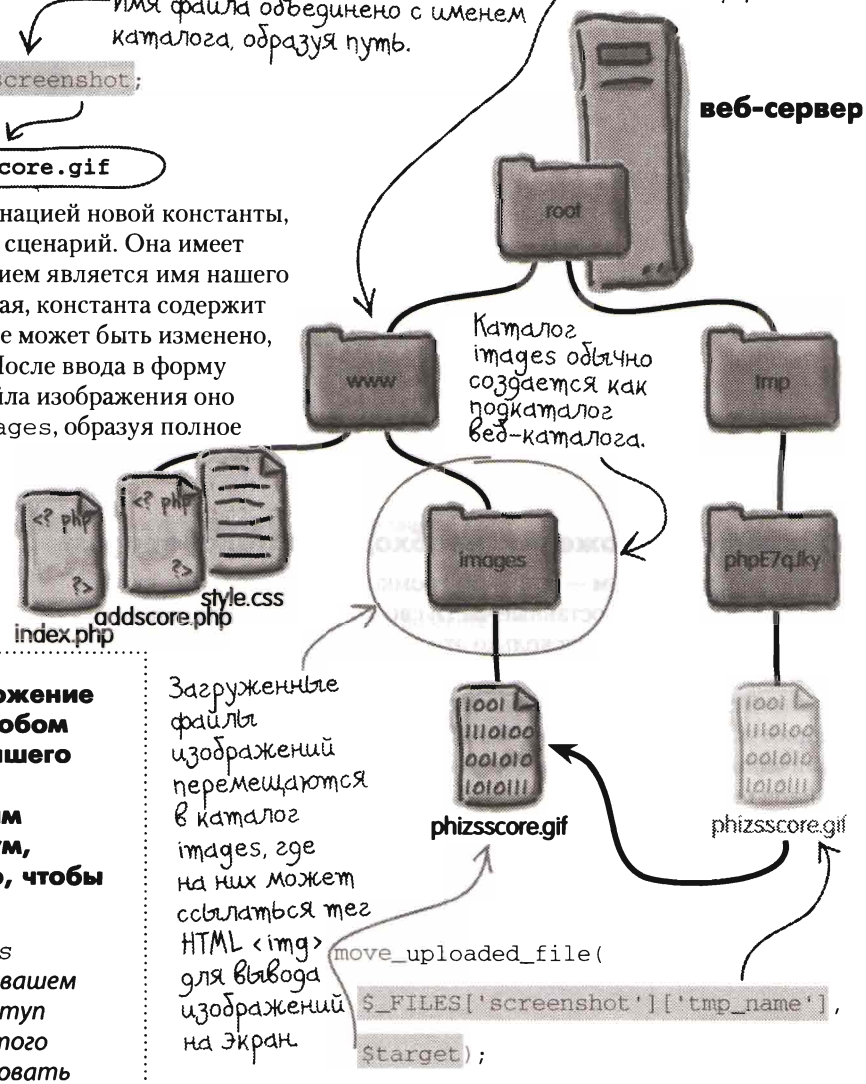
Если ваш каталог `images` размещен в главном каталоге веб-приложения на веб-сервере, можно ссылаться на файлы изображений из PHP-сценария следующим образом:

```
$target = GW_UPLOADPATH . $screenshot;
```

Имя файла объединено с именем каталога, образуя путь.

`images/phizsscore.gif`

Путь `$target` образован конкатенацией новой константы, которую мы намерены добавить в сценарий. Она имеет имя `GW_UPLOADPATH`, и ее значением является имя нашего каталога `images`. Как и переменная, константа содержит данные. Но значение константы не может быть изменено, после того как оно установлено. После ввода в форму «Добавь свой рейтинг» имени файла изображения оно добавляется к имени каталога `images`, образуя полное имя файла, или путь.

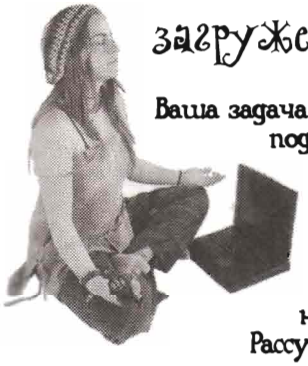


Запомни!

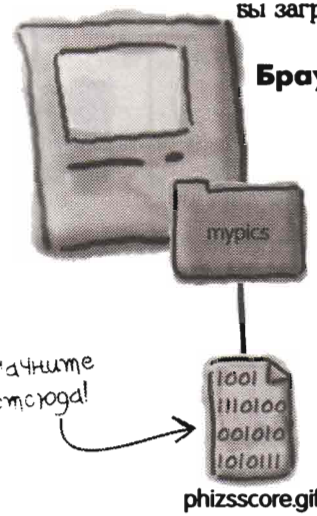
Если PHP-приложение размещено в любом месте, кроме вашего локального компьютера, вам понадобится, как минимум, использовать FTP для того, чтобы создать каталог image.

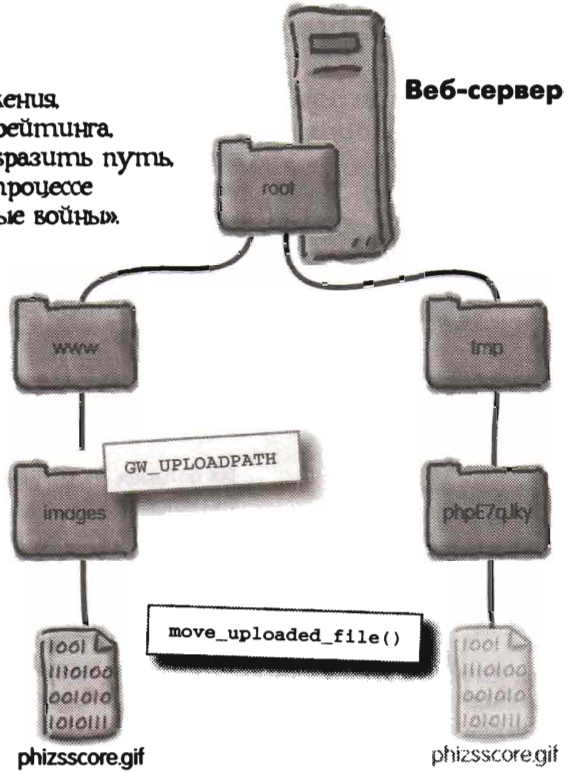
Для создания каталога `images` в веб-каталоге приложения на вашем веб-сервере вам необходим доступ к его файловой системе. Для этого вы можете, например, использовать любую FTP-утилиту.

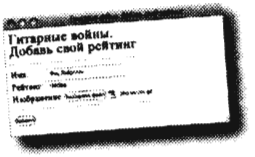
Станьте файлом изображения, загруженным на сервер



Ваша задача — играть роль файла изображения, подтверждающего подлинность рейтинга, загруженного на сервер, и изобразить путь, который файл проходит в процессе работы приложения «Гитарные войны». Начертите свой путь через все этапы приложения, не забыв базу данных. Рассуждайте так, как рассуждал бы загруженный файл!







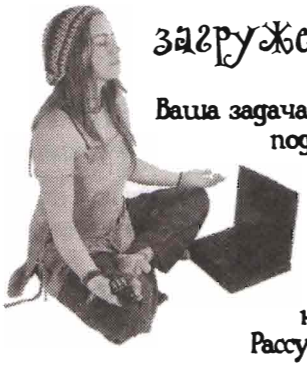

```
$screenshot = $_FILES['screenshot']['name'];
```

```
INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')
```

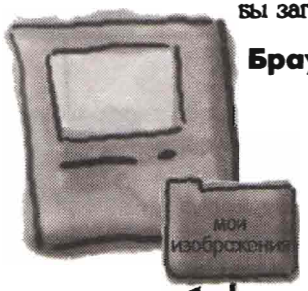
guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чейн	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левинц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif

Станьте файлом изображения, загруженным на сервер



Ваша задача — играть роль файла изображения, подтверждающего подлинность рейтинга, загруженного на сервер, и изобразить путь, который файл проходит в процессе работы приложения «Гитарные войны». Начертите свой путь через все этапы приложения, не забыв базу данных. Рассуждайте так, как рассуждал бы загруженный файл!

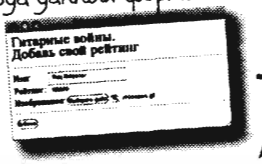


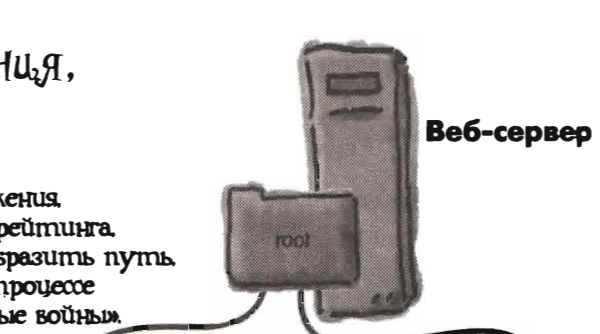
Этот каталог размещен на компьютере пользователя, поэтому у вас нет к нему никакого доступа. Вы не знаете ни его имени, ни того, где он расположен, — вообще ничего.

phizsscore.gif

Вы должны быть очень внимательными с именем и размещением этого каталога, потому что он используется везде в приложении «Гитарные войны» для сохранения загружаемых файлов изображений и ссылок на них.

Во-первых, файл был загружен с использованием поля ввода данных формы.





GW — это начальные буквы слов Guitar Wars («Гитарные войны»). Они означают здесь, что эта константа принадлежит приложению.

Во-вторых, файл перемещен из временного каталога для загружаемых файлов в постоянный каталог для файлов изображений.

После того как файл был загружен на сервер и перемещен на свое постоянное место, его имя было добавлено в базу данных.

```
$screenshot = $_FILES['screenshot']['name'];
INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')
```

Вот так так!

Ну да, это новый этап, который мы ранее не планировали. Схема приложения должна быть гибкой!

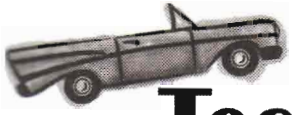
addscore.php

СДЕЛАНО

5 **Переместите загруженный файл изображения из временного каталога для загружаемых файлов в постоянный каталог для файлов изображений.**

guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Иоханссон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif



Тест-драйв

Поселите загруженные файлы изображений, подтверждающих рейтинг, на их постоянное место жительства, в их собственный каталог для размещения файлов изображений.

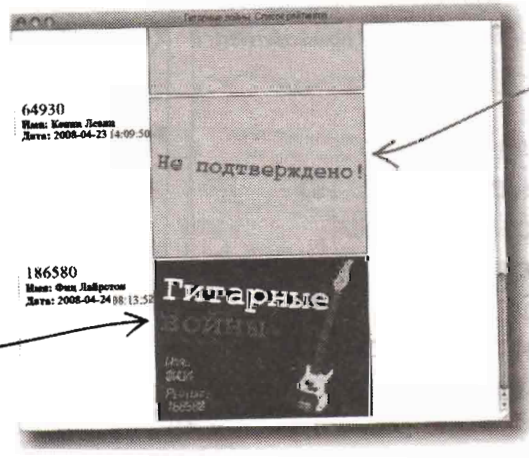
Внесите в сценарий `addscore.php` изменения, касающиеся использования константы `GW_UPLOADPATH` и сохранения загруженных файлов изображений в их собственный каталог. Вот фрагмент кода, который должен быть изменен:

```
<?php
   // Инициализация константы, содержащей имя каталога
   // для загружаемых файлов изображений
   define('GW_UPLOADPATH', 'images/');

   if (isset($_POST['submit'])) {
      // Извлечение данных из суперглобального массива $_POST
      $name = $_POST['name'];
      $score = $_POST['score'];
      $screenshot = $_FILES['screenshot']['name'];

      if (!empty($name) && !empty($score) && !empty($screenshot)) {
         // Перемещение файла в постоянный каталог для файлов изображений
         $target = GW_UPLOADPATH . $screenshot;
         if (move_uploaded_file($_FILES['screenshot']['tmp_name'], $target)) {
            // Соединение с базой данных
            $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');

            // Запись данных в базу данных
            $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score')";
            mysqli_query($dbc, $query);

            // Вывод пользователю подтверждения в получении данных
            echo '<p> Благодарим вас за то, что добавили свой новый рейтинг!</p>';
            echo '<p><strong>Имя:</strong> ' . $name . '<br />';
            echo '<strong>Рейтинг:</strong> ' . $score . '</p>';
            echo '<img src="' . GW_UPLOADPATH . $screenshot . '"
            alt="Изображение, подтверждающее подлинность рейтинга" /><br />';
            echo '<p><a href="index.php">&lt;&lt; Назад к списку рейтингов</a></p>';
```

addscore.php

В сценарии `index.php` также используется константа `GW_UPLOADPATH`. Не забудьте внести касающиеся ее изменения и в этот сценарий. После внесения всех этих изменений загрузите сценарии на ваш веб-сервер и попробуйте опять ввести рейтинг.

Загруженный на сервер файл изображения, подтверждающего подлинность рейтинга, теперь виден на главной странице приложения.



Для старых записей рейтингов, для которых файлы изображений, подтверждающих подлинность, отсутствуют, выводится изображение с фразой «Не подтверждено!»

не бывает глупых вопросов

В: Если файл php.ini может быть использован для контроля над местом размещения загружаемых файлов, зачем тогда перемещать эти файлы?

О: Потому что не всегда имеется возможность вносить изменения в файл `php.ini`. И даже если у вас есть такая возможность, вы рискуете получить проблемы, если возникнет необходимость переместить приложение на другой сервер. Иначе говоря, приложение будет зависимо от пути, контролируемого внешним файлом `php.ini`, в отличие от пути, контролируемого внутренним кодом PHP вашего приложения.

В: Почему дата вводится приложением, а не пользователем?

О: Дата является важной частью рейтинга и определяется моментом, когда рейтинг был официально отправлен на сайт. Как и в случае любого другого рекорда, человек, первым зарегистрировавший свой рекорд, получает всю славу и преимущества. Чем полагаться на пользователя в вопросе даты регистрации рейтинга, мы можем просто использовать дату и время поступления рейтинга на сайт как дату и время его официальной регистрации. Это уменьшает опасность того, что появятся недостоверные даты регистрации рейтингов, и повышает уровень доверия к перечню рейтингов со стороны пользователей. Пользователи таких приложений всегда будут искать какие-либо лазейки для получения преимуществ, поэтому необходимо сократить их количество настолько, насколько это возможно.

В: Разве у пользователя нет возможности переписать чей-нибудь файл изображения, подтверждающий рейтинг, путем загрузки файла с таким же именем?

О: Да. Проблема связана с тем, что файл изображения, подтверждающий рейтинг, сохраняется на веб-сервере под тем же самым именем, которое пользователь ввел в поле ввода имени загружаемого файла. Поэтому, если два пользователя загружают файлы с одинаковым именем, файл первого пользователя будет заменен файлом второго. Это плохо. Одно из решений заключается в том, чтобы повысить уровень уникальности имен файлов, загружаемых на сервер. Простой способ достижения этой цели — добавление текущего на момент загрузки времени в секундах в начало имени файла, как показано ниже:

```
$target = GW_UPLOADPATH .time() .
 $screenshot;
```

В результате интерпретации этого кода имя файла будет `1221634560phizsscore.gif` вместо `phizsscore.gif`, где число 1221634560 является текущим (на момент генерации имени файла, то есть на момент загрузки) временем, выраженным в секундах.

Функция time() не выполняет никакой роли, кроме той, что она является источником уникальных данных... Как известно, в реальной жизни время движется только вперед, поэтому значение, возвращаемое этой функцией, постоянно возрастает!

В: Могли бы мы сохранять непосредственно в базе данных «Гитарных войн» само изображение, подтверждающее рейтинг пользователя?

О: Да. База данных, являясь исключительно гибкой системой, позволяет сохранять бинарные данные. Но основная проблема здесь заключается в том, что приложение «Гитарные войны» использует HTML-код для вывода загруженных изображений на экран в главной странице `index.php`. HTML-тег `<img>` разработан таким образом, что он использует ссылку на имя файла изображения, сохраненного на сервере, а не на бинарные данные, содержащиеся в базе данных. Поэтому, если вы измените таблицу `guitarwars` так, чтобы она смогла хранить бинарные данные изображений рейтингов, то столкнетесь с серьезной проблемой, связанной с извлечением их из таблицы в формате, необходимом для вывода на экран с использованием HTML-кода.

КЛЮЧЕВЫЕ МОМЕНТЫ

- Запрос ALTER используется для того, чтобы изменить структуру таблицы базы данных MySQL, например, чтобы добавить новую колонку данных.
- С небольшой помощью PHP и MySQL HTML-тег `<input>` может быть использован для загрузки файлов изображений на сервер.
- Суперглобальная переменная `$_FILES` служит для того, чтобы сохранять информацию о загруженных на сервер файлах.
- Стандартная PHP-функция `move_uploaded_file()` дает вам возможность перемещать файлы на сервере и играет ключевую роль в обработке загружаемых на сервер файлов.
- В большинстве веб-приложений считается очень важным сохранять все файлы изображений, используемые приложением, в отдельном каталоге, обычно с именем `images`. Особенно важно это для файлов, загружаемых пользователями этих приложений.

В базах данных очень удобно хранить текстовые данные, но, когда дело касается бинарных данных, обычно удобнее использовать базы данных для сохранения в них ссылок на имена внешних файлов, содержащих эти бинарные данные.

Мне нравится то, что имя каталога для хранения загруженных на сервер файлов содержится в константе, но почему она создается в двух местах: index.php и addscore.php? Что произойдет при изменении этого имени?

В константе GW_UPLOADPATH содержится имя каталога для хранения загруженных на сервер файлов изображений.

```
define('GW_UPLOADPATH', 'images/');
```

Функция define() используется для того, чтобы создать и инициализировать константу.

Имя константы.

Значение константы, которое не может быть изменено... На то она и константа!

Если имя каталога изменится, вы должны будете изменить код в двух местах... Дублирование кода — очень плохая практика!

Итак, внутри каждого из отдельных файлов `index.php` и `addscore.php` константа `GW_UPLOADPATH` работает хорошо. Но инициализация этой константы осуществляется в каждом сценарии, и это означает, что если имя каталога должно быть изменено, то это должно быть сделано в каждом сценарии. Такой вид дублирования кода считается очень плохой практикой, и его следует избегать, насколько это возможно.

```
// Инициализация константы, содержащей
// имя каталога для загружаемых файлов изображений
define('GW_UPLOADPATH', 'images/');
```

index.php

Константа инициализирована дважды, и это означает, что одна и та же константа должна поддерживаться в двух разных местах.

```
// Инициализация константы, содержащей
// имя каталога для загружаемых файлов изображений,
// и константы, содержащей значение максимального размера файла
define('GW_UPLOADPATH', 'images/');
define("GW_MAXFILESIZE", 32768);
```

addscore.php

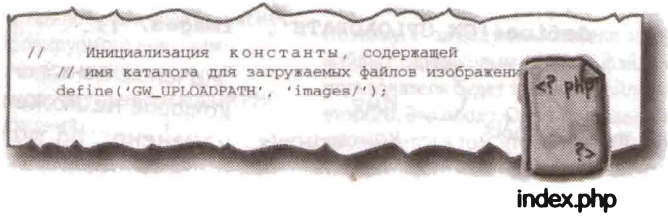
СИЛА МЫСЛИ

Для решения проблемы дублирования кода нам необходимо инициализировать константу в одном месте. Где бы вы инициализировали эту константу: в сценарии `index.php` или в сценарии `addscore.php`? Почему?

Общие данные должны использоваться совместно

Когда речь идет о данных, используемых совместно несколькими сценариями приложения, вам необходим способ инициализировать эти данные в одном месте и затем по необходимости использовать их в разных сценариях. Но это высказывание еще не отвечает на вопрос, где же именно нужно инициализировать эти данные...

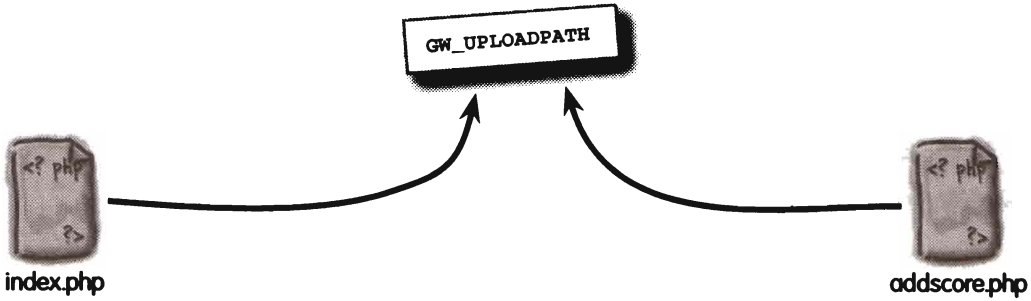
Вы могли бы инициализировать данные в сценарии index.php...



Файлы, используемые совместно несколькими сценариями приложения, должны быть доступны любому из них без дублирования кода.

...но тогда другие сценарии не будут иметь доступа к этим данным.

Инициализация совместно используемых данных в каком-то одном сценарии не дает ожидаемого результата, поскольку другие сценарии не имеют к ним доступа, в связи с чем эти данные фактически не являются совместными. Решение этой проблемы заключается в поиске способа, который обеспечил бы нескольким сценариям доступ к данным, не инициализируя их ни в одном из этих сценариев.



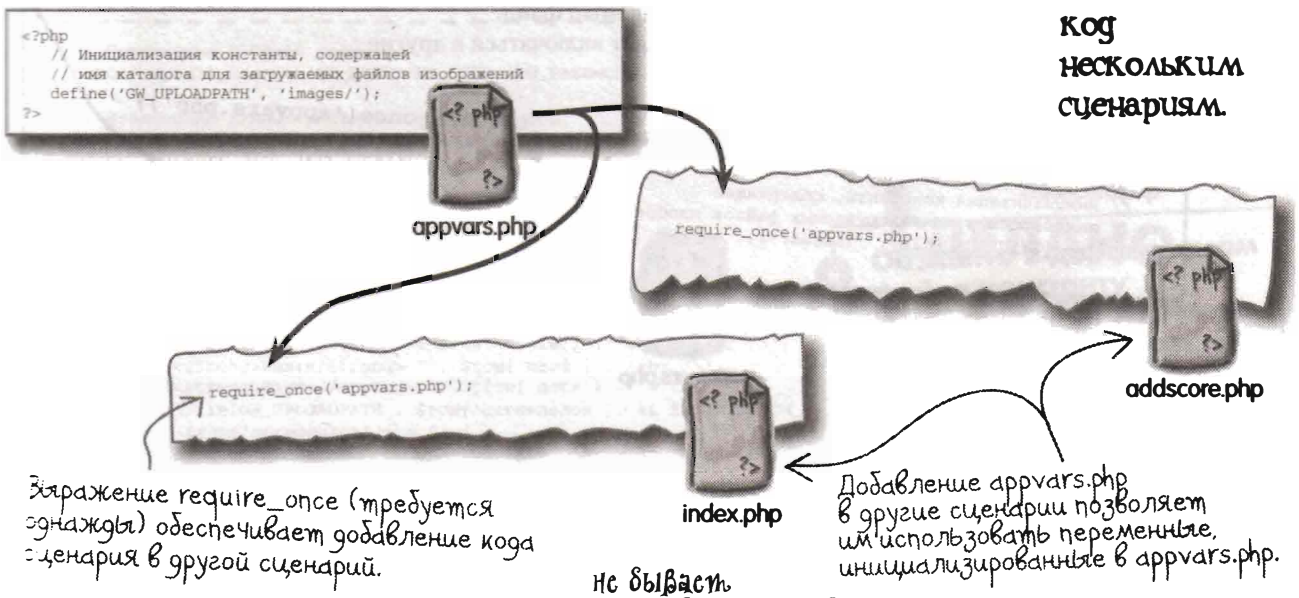
Как же мы сможем сделать данные доступными двум сценариям, не инициализируя эти данные ни в одном из них?

Обеспечение совместного доступа к данным заключается в использовании **включаемых** файлов, содержащих PHP-код, который при необходимости может быть включен в любой PHP-сценарий.

Совместные данные включаются в сценарий в тот момент, когда они требуются

Включаемые файлы позволяют совместно использовать код нескольким сценариям.

Использование добавляемых файлов — это очень мощный механизм, потому что вы создаете их однажды и затем включаете их код в любые сценарии по необходимости, что позволяет очень эффективно использовать совместный код. Константа `GW_UPLOADPATH` может быть инициализирована в добавляемом файле, формируя таким образом «набор переменных уровня приложения».



Не бывает глупых вопросов

В: Послушайте, а действительно ли эти «переменные уровня приложения» являются константами?

О: Иногда да. Но здесь все в порядке. Нашей целью не является дать строгое определение разницы между переменной и константой. Мы всего лишь пытаемся найти то место, где мы могли бы поместить код для совместного использования всеми сценариями приложения. И таким местом явился для нас сценарий `appvars.php`.

В: Код в добавляемом сценарии должен быть ограничен только данными?

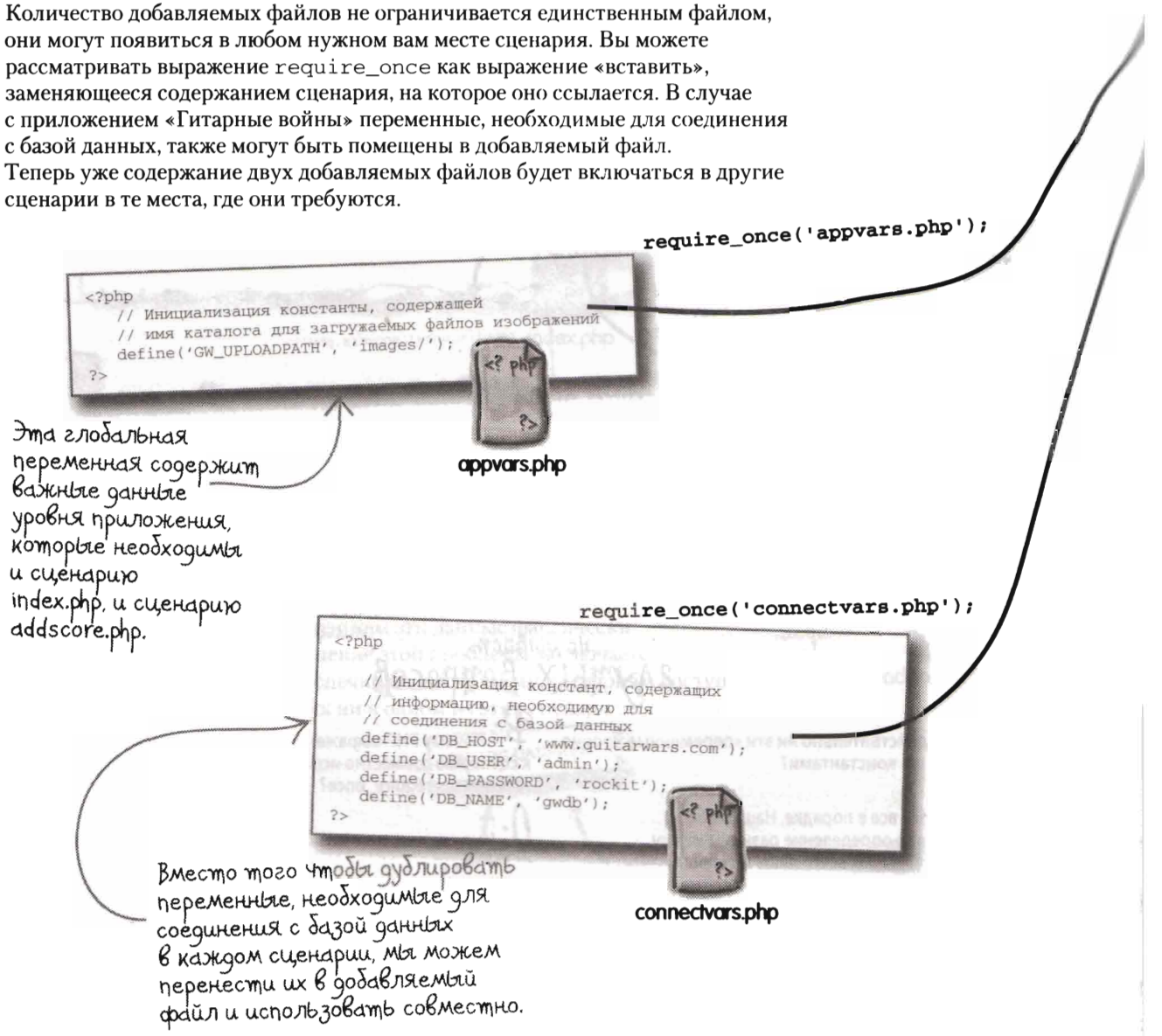
О: Совсем нет. Любой PHP-код может быть помещен в свой собственный сценарий и добавляться в другие сценарии приложения, если использовать выражение `require_once`. Фактически использование разными сценариями приложения совместного кода является очень распространенной практикой. Это часто очень помогает в вопросах организации кода всего приложения.

В: Почему PHP-выражение, обеспечивающее добавление к сценарию совместно используемого кода из другого сценария, называется require_once?

О: Имя «включить файл» (`include file`) происходит от PHP-выражения `include` (включить), которое выполняет функцию, подобную функции выражения `require_once`. Разница заключается в том, что если файл, указанный в выражении `require_once`, не будет найден, то будет выведено сообщение об ошибке. В случае же использования выражения `include` ошибка выведена не будет. Слово `once` (однажды) в этом выражении означает, что код будет добавлен в сценарий только однажды, даже если это выражение встретится в нем (по ошибке) более одного раза. Иногда вы увидите использование выражения `include` вместо выражения `require_once` для включения не слишком важного кода, например чистого HTML-кода, не используемого в критических ситуациях. В PHP также используются выражения `include_once` и `require`, которые являются вариантами выражений `require_once` и `include`.

Рассматривайте выражение require_once как «вставить»

Количество добавляемых файлов не ограничивается единственным файлом, они могут появиться в любом нужном вам месте сценария. Вы можете рассматривать выражение `require_once` как выражение «вставить», заменяющееся содержанием сценария, на которое оно ссылается. В случае с приложением «Гитарные войны» переменные, необходимые для соединения с базой данных, также могут быть помещены в добавляемый файл. Теперь уже содержание двух добавляемых файлов будет включаться в другие сценарии в те места, где они требуются.



Выражение REQUIRE_ONCE добавляет совместно используемый код в другие сценарии.

```
<?php
   // Инициализация константы, содержащей
   // имя каталога для загружаемых файлов изображений
   define('GW_UPLOADPATH', 'images/');

   // Инициализация констант, содержащих
   // информацию, необходимую для
   // соединения с базой данных
   define('DB_HOST', 'www.quitarwars.com');
   define('DB_USER', 'admin');
   define('DB_PASSWORD', 'rockit');
   define('DB_NAME', 'gwdb');

   // Соединение с базой данных
   $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

   // Извлечение данных рейтингов из базы MySQL
   $query = "SELECT * FROM guitarwars ORDER BY score DESC, date ASC";
   $data = mysqli_query($dbc, $query);

   // Прохождение в цикле по массиву записей рейтингов.
   // Форматирование данных записей в виде HTML-кода.
   echo '<table>';
   while ($row = mysqli_fetch_array($data)) {
      // Вывод данных рейтингов
      echo '<tr><td class="scoreinfo">';
      echo '<span class="score">' . $row['score'] . '</span><br />';
      echo '<strong>Имя:</strong> ' . $row['name'] . '<br />';
      echo '<strong>Дата:</strong> ' . $row['date'] . '</td>';
      if (is_file(GW_UPLOADPATH . $row['screenshot']) && filesize(GW_
UPLOADPATH . $row['screenshot']) > 0) {
         echo '<td><img src="' . GW_UPLOADPATH . $row['screenshot'] . '"
alt="Подтвержденс " /></td></tr>';
      }
      else {
         echo '<td><img src="' . GW_UPLOADPATH . 'unverified.gif' . '"
alt="Не подтверждено!" /></td></tr>';
      }
   }
   echo '</table>';

   mysqli_close($dbc);
?>
```

index.php

addscore.php

Отлично! Теперь и я имею доступ к совместно используемым данным.

СДЕЛАНО

6 **Объявите в добавляемом файле константу и присвойте ей значение имени каталога для загружаемых файлов. Добавьте в сценарий код, обеспечивающий ее совместное использование.**

Ой, еще один новый этап! Некоторые задачи не так легко планировать, поэтому будьте всегда готовы корректировать схему своего приложения на лету.

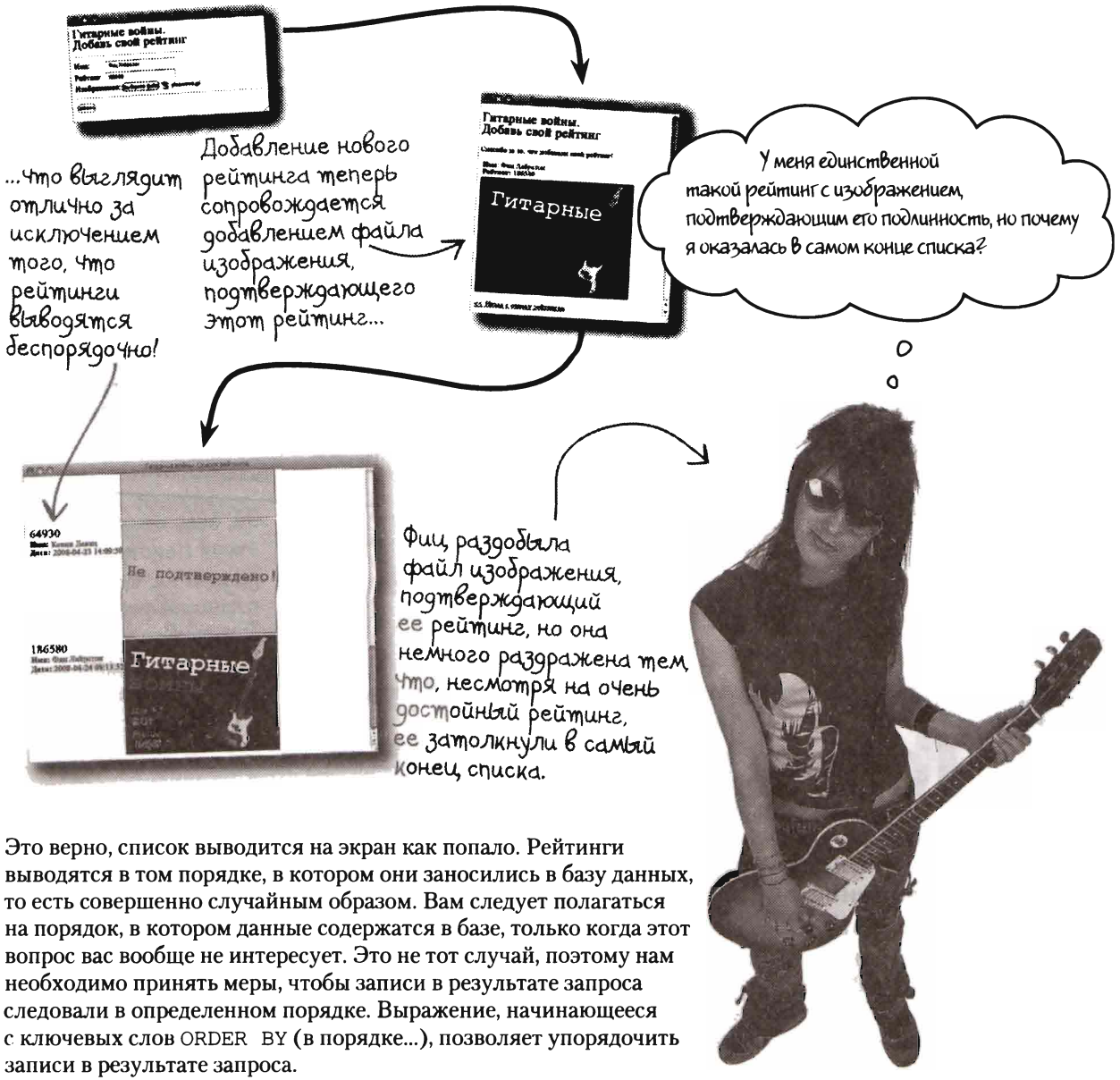
Тест-драйв

Создайте два добавляемых файла для приложения «Гитарные войны» и откорректируйте другие сценарии, обеспечив совместное использование общих переменных.

Создайте два добавляемых файла `appvars.php` и `connectvars.php` и введите в них код, показанный на предыдущей странице. Затем добавьте выражение `require_once` в сценарии `index.php` и `addscore.php` так, чтобы содержание обоих добавляемых файлов было включено в эти сценарии. Загрузите новые версии всех сценариев на ваш веб-сервер, проверьте работу формы «Добавь свой рейтинг» и главной страницы. Убедитесь в том, что все в порядке с новой организационной структурой, основанной на совместном использовании данных с помощью добавляемых файлов.

Распределение по ~~времени~~ порядку — это все, когда речь идет о рейтингах

Наконец-то приложение «Гитарные войны» позволяет пользователям загружать на сервер файлы изображений, подтверждающие их рейтинги. Хотя это и является существенным усовершенствованием приложения, оно не решает проблему, на которую пользователи периодически жалуются, а именно отсутствие определенного порядка, в котором их рейтинги появляются на главной странице.



Это верно, список выводится на экран как попало. Рейтинги выводятся в том порядке, в котором они заносились в базу данных, то есть совершенно случайным образом. Вам следует полагаться на порядок, в котором данные содержатся в базе, только когда этот вопрос вас вообще не интересует. Это не тот случай, поэтому нам необходимо принять меры, чтобы записи в результате запроса следовали в определенном порядке. Выражение, начинающееся с ключевых слов `ORDER BY` (в порядке...), позволяет упорядочить записи в результате запроса.

Магниты PHP и MySQL

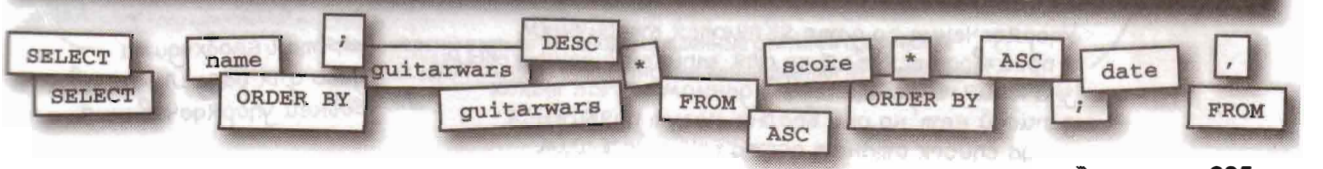
Проверьте, как работает выражение `ORDER BY`, создавая с помощью магнитов упорядоченные запросы `SELECT`, в результате выполнения которых будут получены показанные ниже результаты. Подумайте также, какой из запросов в большей степени отражает главную идею приложения «Гитарные войны». Примечание: `ASC` — сокращение от ASCending (возрастающий), `DESC` — от DESCending (убывающий).

```
Файл Правка Окно Помощь XYZ
mysql>

+----+---------------------+-----------------+---------+---------------+
| id | date                | name            | score   | screenshot    |
+----+---------------------+-----------------+---------+---------------+
| 5  | 2008-04-23 09:13:34 | Аштон Симпсон   | 368420  |               |
| 4  | 2008-04-23 09:12:53 | Белита Чеви     | 282470  |               |
| 3  | 2008-04-23 09:06:35 | Эдди Ванилли    | 345900  |               |
| 6  | 2008-04-23 14:09:50 | Кенни Левиц     |  64930  |               |
| 2  | 2008-04-22 21:27:54 | Невил Йохансон  |  98430  |               |
| 1  | 2008-04-22 14:37:34 | Пако Джасториус | 127650  |               |
| 7  | 2008-04-24 08:13:52 | Фиц Лайрстон    | 186580  | phizsscore.gif |
+----+---------------------+-----------------+---------+---------------+

7 строк в списке (0.0005 сек)
```

В результате запроса записи расположены в порядке убывания цифровых значений рейтингов и затем в порядке возрастания дат.

В результате запроса записи расположены в прямом (возрастающем) алфавитном порядке имен пользователей.

```
Файл Правка Окно Помощь XYZ
mysql>

+----+---------------------+-----------------+---------+---------------+
| id | date                | name            | score   | screenshot    |
+----+---------------------+-----------------+---------+---------------+
| 5  | 2008-04-23 09:13:34 | Аштон Симпсон   | 368420  |               |
| 3  | 2008-04-23 09:06:35 | Эдди Ванилли    | 345900  |               |
| 4  | 2008-04-23 09:12:53 | Белита Чеви     | 282470  |               |
| 7  | 2008-04-24 08:13:52 | Фиц Лайрстон    | 186580  | phizsscore.gif |
| 1  | 2008-04-22 14:37:34 | Пако Джасториус | 127650  |               |
| 2  | 2008-04-22 21:27:54 | Невил Йохансон  |  98430  |               |
| 6  | 2008-04-23 14:09:50 | Кенни Левиц     |  64930  |               |
+----+---------------------+-----------------+---------+---------------+

7 строк в списке (0.0005 сек)
```



Магниты PHP и MySQL

Проверьте, как работает выражение `ORDER BY`, создавая с помощью магнитов упорядоченные запросы `SELECT`, в результате выполнения которых будут получены показанные ниже результаты. Подумайте также, какой из запросов в большей степени отражает главную идею приложения «Гитарные войны». Примечание: `ASC` — сокращение от ASCending (возрастающий), `DESC` — от DESCending (убывающий).

```
Файл Правка Окно Помощь XYZ
mysql> SELECT * FROM guitarwars ORDER BY name ASC ;
+----+---------------------+-----------------+--------+-----------------+
| id | date                | name            | score  | screenshot      |
+----+---------------------+-----------------+--------+-----------------+
| 5  | 2008-04-23 09:13:34 | Аштон Симпсон   | 368420 |                 |
| 4  | 2008-04-23 09:12:53 | Белита Чеви     | 282470 |                 |
| 3  | 2008-04-23 09:06:35 | Эдди Ванилли    | 345900 |                 |
| 6  | 2008-04-23 14:09:50 | Кенни Левиц     |  64930 |                 |
| 2  | 2008-04-22 21:27:54 | Невил Йохансон  |  98430 |                 |
| 1  | 2008-04-22 14:37:34 | Пако Джасториус | 127650 |                 |
| 7  | 2008-04-24 08:13:52 | Фиц Лайрстон    | 186580 | phizsscore.gif  |
+----+---------------------+-----------------+--------+-----------------+

7 строк в списке (0.0005 сек)
```

В результате запроса записи расположены в порядке убывания цифровых значений рейтингов и затем в порядке возрастания дат.

В результате запроса записи расположены в прямом (возрастающем) алфавитном порядке имен пользователей.

Этот запрос в большей степени отражает главную идею приложения «Гитарные войны»!

```
Файл Правка Окно Помощь XYZ
mysql> SELECT * FROM guitarwars ORDER BY score DESC , date ASC ;
+----+---------------------+-----------------+--------+-----------------+
| id | date                | name            | score  | screenshot      |
+----+---------------------+-----------------+--------+-----------------+
| 5  | 2008-04-23 09:13:34 | Аштон Симпсон   | 368420 |                 |
| 3  | 2008-04-23 09:06:35 | Эдди Ванилли    | 345900 |                 |
| 4  | 2008-04-23 09:12:53 | Белита Чеви     | 282470 |                 |
| 7  | 2008-04-24 08:13:52 | Фиц Лайрстон    | 186580 | phizsscore.gif  |
| 1  | 2008-04-22 14:37:34 | Пако Джасториус | 127650 |                 |
| 2  | 2008-04-22 21:27:54 | Невил Йохансон  |  98430 |                 |
| 6  | 2008-04-23 14:09:50 | Кенни Левиц     |  64930 |                 |
+----+---------------------+-----------------+--------+-----------------+

7 строк в списке (0.0005 сек)
```

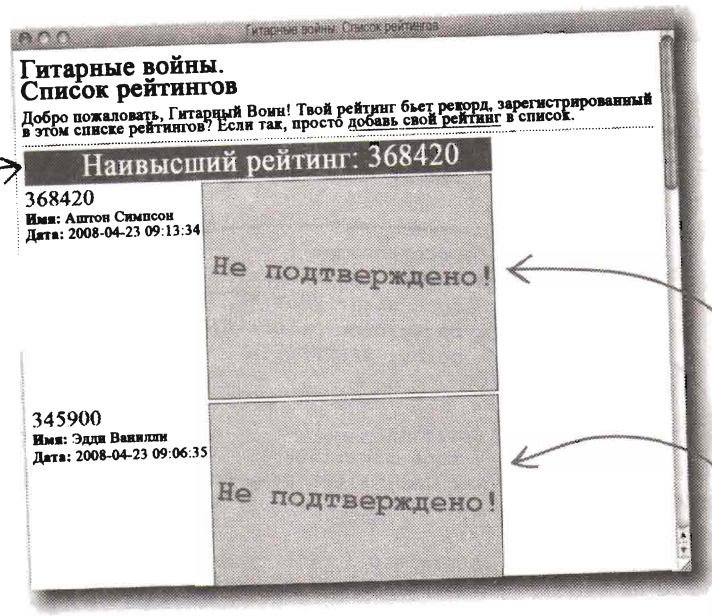
Упорядочение по дате является вторичным и производится только для записей с одинаковыми значениями рейтингов. В данном случае таких записей нет, но они вполне могут появиться, когда список станет достаточно большим.

Запятая необходима здесь для разделения двух уровней упорядочивания.

Награды самым лучшим гитарным воинам

После того как мы разобрались с упорядочиванием рейтингов, теперь можем сделать неожиданное улучшение списка рейтингов путем перемещения наивысших рейтингов в самое начало списка. Гитарный воин с наивысшим рейтингом заслуживает права находиться в заголовке списка, чтобы все видели, кто самый сильный гитарный воин и против какого рейтинга придется воевать, чтобы его победить.

Заголовок списка рейтингов наглядно демонстрирует наивысший рейтинг, показывая цель, к которой будут стремиться гитарные воины в своей борьбе.



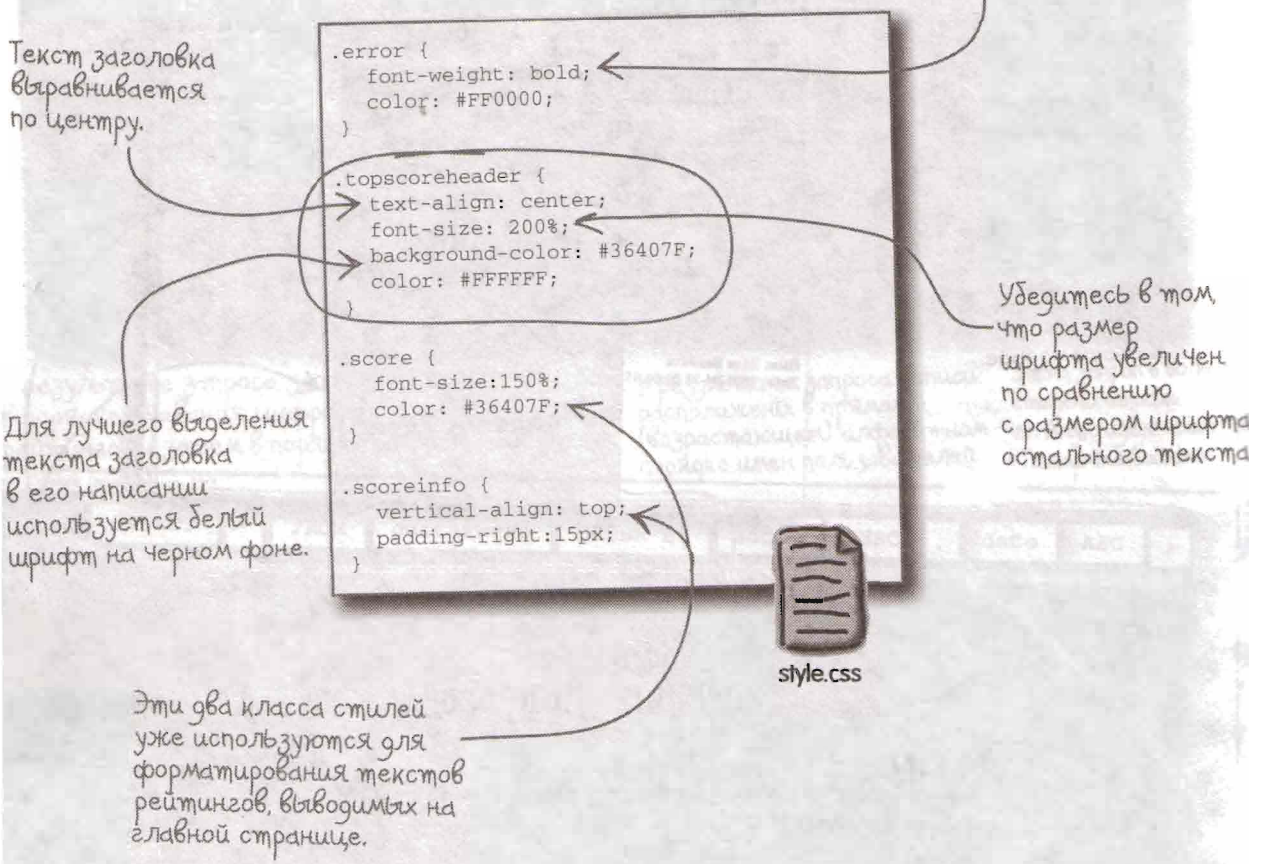
Не бывает глупых вопросов

В: Многие рейтинги все еще не подтверждены. Разве это не проблема?

О: Да, это проблема. Но это не должно останавливать нас на пути к акцентированию наивысшего рейтинга. Это всего лишь означает, что нам в итоге придется очистить рейтинговый лист и удалить из него записи с неподтвержденными рейтингами. Фактически мы намерены взяться за неподтвержденные рейтинги сразу после того, как закончим дело с акцентированием внимания на наивысшем рейтинге.

Форматирование списка рейтингов с помощью HTML и CSS

Наиболее важным моментом в добавленном заголовке списка рейтингов является то, что он должен быть хорошо виден в самом начале списка. Для того чтобы визуально выделить его на фоне остального списка, нам потребуется помощь и HTML, и CSS. Заголовок генерируется как строка HTML-таблицы, а для визуального выделения его из общего фона к нему применяется особый CSS-стиль. Этот стиль, `topscoreheader`, должен быть добавлен в файл каскадных таблиц стилей приложения «Гитарные войны» `style.css`.



Сценарий `index.php` уже генерирует HTML-таблицу, содержащую список рейтингов. Для создания заголовка в самой верхней части списка требуется определить значение самого высокого рейтинга, который гарантированно находится в самом начале списка, так как он теперь упорядочен по убыванию значений рейтинга. Цикл `while` проходит через все рейтинги по порядку, поэтому нам необходимо найти способ как-то подсчитать эти проходы цикла и сгенерировать заголовок только в первом из них...

Измените код в файле `index.php` приложения «Гитарные войны» так, чтобы добавить форматирование заголовка списка в соответствии с классом CSS-стиля `topscoreheader`.

Совет: не забывайте, что заголовок списка рейтингов — это часть состоящей из двух колонок HTML-таблицы, в которой располагается список рейтингов.

```
...
// Прохождение в цикле по массиву записей рейтингов.
 // Форматирование данных записей в виде HTML-кода.
echo '<table>';
$i = 0;
while ($row = mysqli_fetch_array($data)) {
  // Вывод данных рейтингов
  if (.....) {
                          .....  ..... .
    .....
  }
  echo '<tr><td class="scoreinfo">';
  echo '<span class="score">' . $row['score'] . '</span><br />';
  echo '<strong>Имя:</strong> ' . $row['name'] . '<br />';
  echo '<strong>Дата:</strong> ' . $row['date'] . '</td>';
  if (is_file(GW_UPLOADPATH . $row['screenshot']) &&
    filesize(GW_UPLOADPATH . $row['screenshot']) > 0) {
    echo '<td><img src="' . GW_UPLOADPATH . $row['screenshot'] .
      '" alt="Подтверждено" /></td></tr>';
  }
  else {
    echo '<td><img src="' . GW_UPLOADPATH . 'unverified.gif' .
      '" alt="Не подтверждено!" /></td></tr>';
  }
}
echo '</table>';
 ..
```

index.php

Решение к упражнению

Измените код в файле `index.php` приложения «Гитарные войны» так, чтобы добавить форматирование заголовка списка в соответствии с классом CSS-стиля `topscoreheader`.

Совет: не забывайте, что заголовок списка рейтингов — это часть состоящей из двух колонок HTML-таблицы, в которой располагается список рейтингов.

```
...
// Прохождение в цикле по массиву записей рейтингов.
 // Форматирование данных записей в виде HTML-кода.
echo '<table>';
$i = 0;
while ($row = mysqli_fetch_array($data)) {
  // Вывод данных рейтингов
  if ($i == 0) {
    echo '<tr><td colspan="2" class="topscoreheader">Наивысший рейтинг:
    $row['score'] . '</td></tr>';
  }
  echo '<tr><td class="scoreinfo">';
  echo '<span class="score">' . $row['score'] . '</span><br />';
  echo '<strong>Имя:</strong> ' . $row['name'] . '<br />';
  echo '<strong>Дата:</strong> ' . $row['date'] . '</td>';
  if (is_file(GW_UPLOADPATH . $row['screenshot']) &&
    filesize(GW_UPLOADPATH . $row['screenshot']) > 0) {
    echo '<td><img src="' . GW_UPLOADPATH . $row['screenshot'] .
      '" alt="Подтверждено" /></td></tr>';
  }
  else {
    echo '<td><img src="' . GW_UPLOADPATH . 'unverified.gif' .
      '" alt="Не подтверждено!" /></td></tr>';
  }
  $i++;
}
echo '</table>';
```

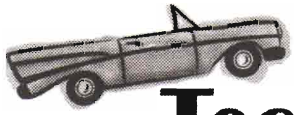
$i — это переменная, которая считает количество прохождений цикла. Мы можем использовать ее для того, чтобы определить первое из них, которому соответствует первый (наивысший) рейтинг.

Когда $i равно 0, мы точно знаем, что в этом проходе цикла мы рассматриваем первую запись рейтинга, которой соответствует наивысшее его значение, поэтому именно в этом месте необходимо записать HTML-код для вывода заголовка списка.

Класс topscoreheader определен в файле style.css.

Значение счетчика проходов цикла увеличивается на единицу в конце каждого прохода. Этот код эквивалентен коду $i = $i + 1;

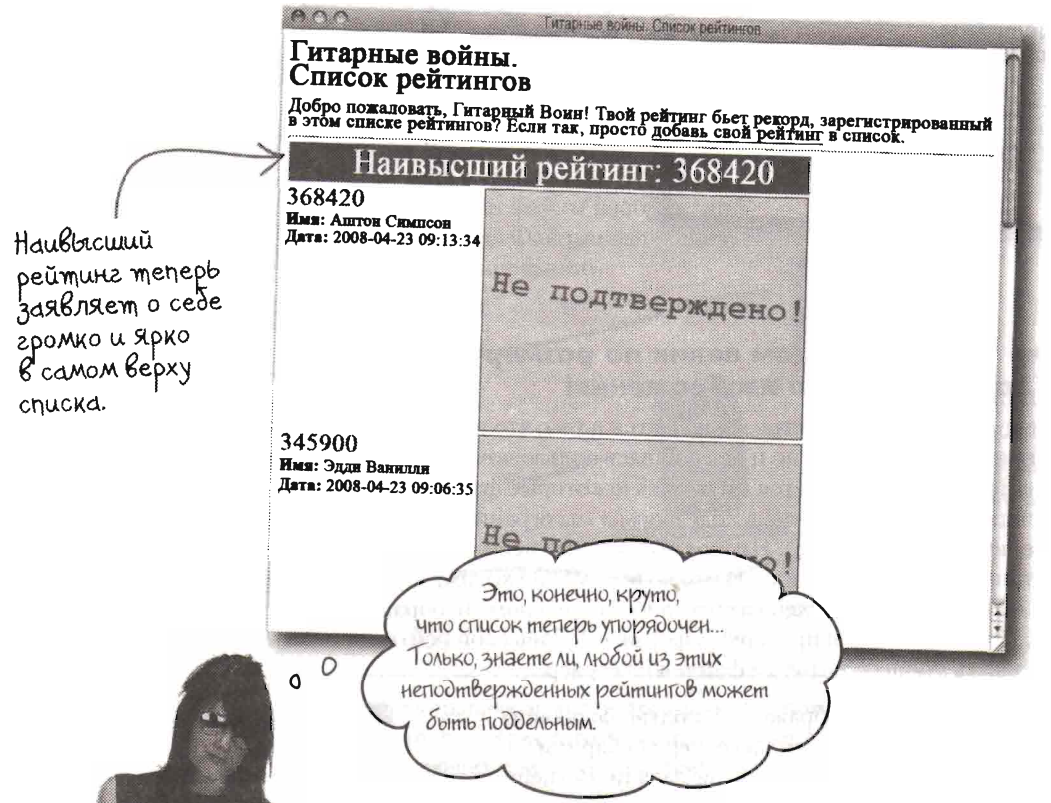
index.php

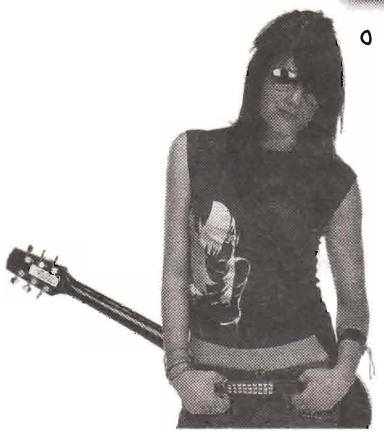


Тест-драйв

Упорядочьте список рейтингов и продемонстрируйте всем наивысший рейтинг.

Внесите в файл `index.php` изменения, касающиеся упорядочения результата запроса `SELECT`, затем добавьте код, который генерирует заголовок списка рейтингов. Загрузите откорректированный сценарий на ваш веб-сервер и откройте его в браузере, чтобы убедиться в том, что наивысший рейтинг выделяется на общем фоне страницы.





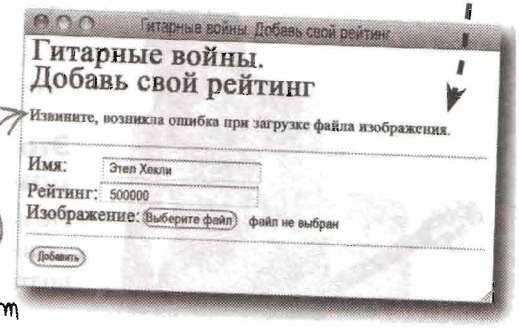
Это верно, с неподтвержденными рейтингами надо разбираться.

Но не все сразу. Похоже, возникла еще одна проблема, мешающая пользователям загружать свои файлы изображений, подтверждающие подлинность рейтингов...

Файл не только слишком велик по размеру, но это даже и не файл изображения!

У нас имеется проблема, которая заключается в том, что приложение не только отказывается загружать некоторые файлы, но и не сообщает пользователю, по какой причине. Хорошо то, что приложение отказывается загружать некоторые файлы, в данном случае потому, что он слишком велик (помните, в коде формы мы ограничили размер загружаемых файлов величиной в 32 Кбайт). Но нам необходимо ясно проинформировать пользователя, по какой причине это произошло. И это не все. Мы не хотим, чтобы пользователи загружали на сервер файлы, которые содержат что-либо иное, кроме изображений, подтверждающих рейтинги. Дополнительная проверка в форме «Добавь свой рейтинг» позволит нам лучше контролировать процесс загрузки файлов на сервер.

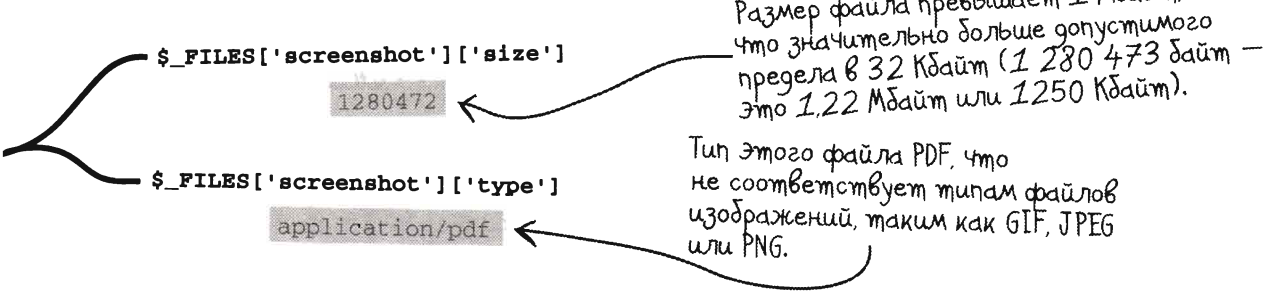
Итак, проверка файлов изображений, подтверждающих рейтинги, во время их загрузки на сервер сценарием `addscore.php` должна выполнять две задачи. Во-первых, она должна предотвращать загрузку больших файлов, сообщая пользователю, что размер файла превышает 32 Кбайт. И во-вторых, она не должна допускать, чтобы пользователи загружали любые другие файлы, кроме изображений. Сценарий «Добавь свой рейтинг» должен проверять как размер файла, так и его тип.



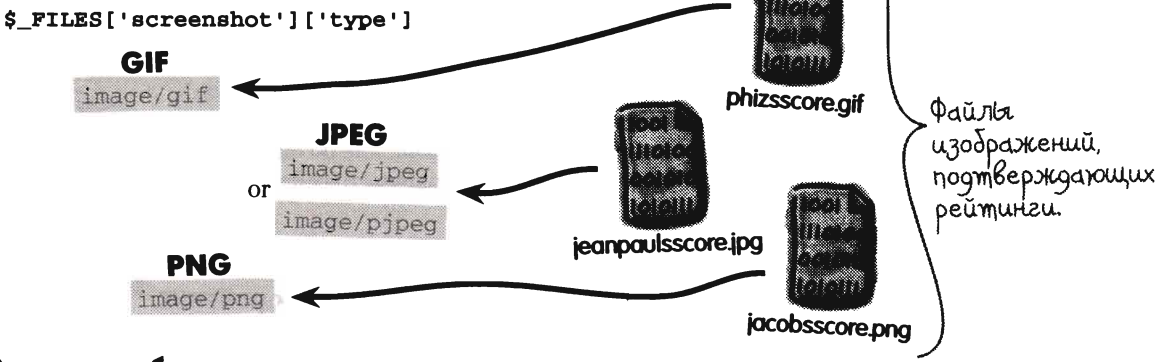
Это сообщение об ошибке мало что говорит пользователю по поводу того, что он делает неправильно при загрузке своего рейтинга.

Допускаются только небольшие файлы изображений

Итак, как именно мы будем проверять в сценарии «Добавь свой рейтинг», что загружаемые файлы отвечают требованиям размера и типа? Ответ заключается в использовании встроенной суперглобальной переменной `$_FILES`, с помощью которой, если вы помните, мы определяли ранее место временного хранения загружаемых на сервер файлов, чтобы затем перенести их в каталог `images`. Теперь мы намерены использовать эту переменную, чтобы узнать размер и тип MIME загружаемого файла.



Мы не просто хотим, чтобы размеры загружаемых файлов были меньше 32 Кбайт, но нам также необходимо, чтобы это были файлы изображений тех типов, которые могут быть выведены в браузере. Файлы следующих типов MIME обычно используются в веб-программировании:



Время поработать

Напишите управляющую конструкцию `if`, с помощью которой организуйте проверку того, что загружаемые файлы изображений, подтверждающих рейтинги, действительно являются файлами изображений и их размер больше 0 байт, но меньше, чем значение константы `GW_MAXFILESIZE`. Имеется в виду, что значение размера файла и его тип были предварительно сохранены в переменных с именами `$screenshot_size` и `$screenshot_type` соответственно.

```
if (
.....

.....

.....) {
```

Решение задачи

Напишите управляющую конструкцию `if`, с помощью которой организуйте проверку того, что загружаемые файлы изображений, подтверждающих рейтинги, действительно являются файлами изображений и их размер больше 0 байт, но меньше, чем значение константы `GW_MAXFILESIZE`. Имеется в виду, что значение размера файла и его тип были предварительно сохранены в переменных с именами `$screenshot_size` и `$screenshot_type` соответственно.

Некоторые браузеры используют этот тип MIME для распознавания изображений в формате JPEG.

```
if (($screenshot_type == 'image/gif') || ($screenshot_type == 'image/jpeg') ||
    ($screenshot_type == 'image/pjpeg') || ($screenshot_type == 'image/png')) &&
    ($screenshot_size > 0) && ($screenshot_size <= GW_MAXFILESIZE)) {
```

```
<?php
  // Инициализация константы, содержащей
  // имя каталога для загружаемых файлов изображений,
  // и константы, содержащей значение максимального
размера файла
  define('GW_UPLOADPATH', 'images/');
  define('GW_MAXFILESIZE', 32768);    //32 KB
?>
```

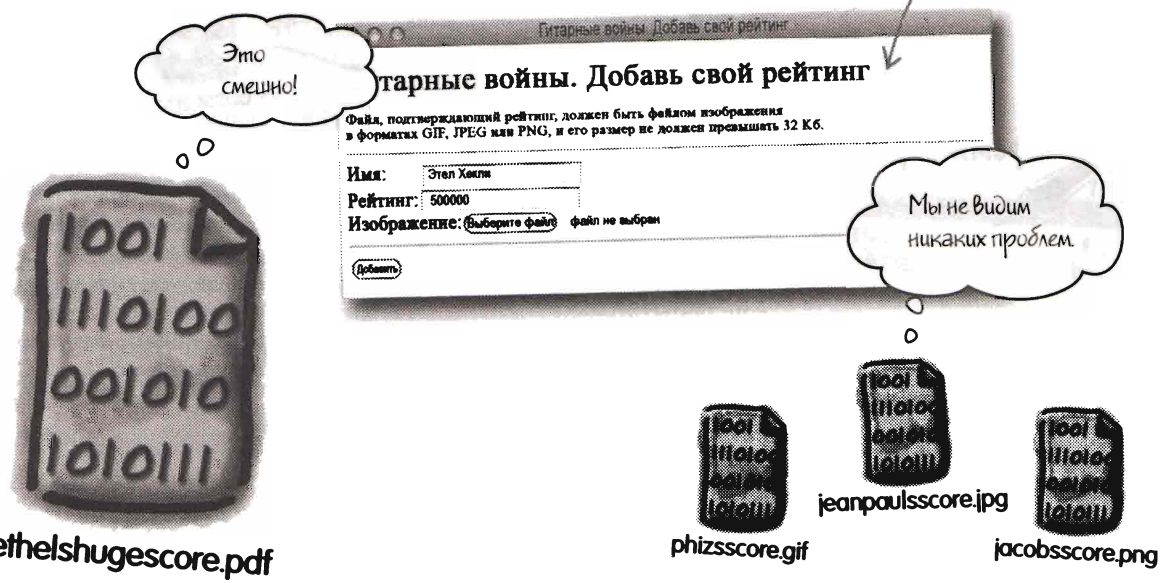
appvars.php

Так как допустимый максимальный размер файла появляется теперь более чем в одном месте в сценарии «Добавь свой рейтинг», есть смысл сохранить это значение в константе.

Проверка параметров файлов во время загрузки делает сценарий более надежным

Простая проверка прошла долгий путь, чтобы сделать PHP-приложение более интуитивным и простым в использовании, не говоря уже о том, чтобы повысить уровень его защиты от взлома. Теперь информативное сообщение об ошибке даст пользователю возможность понять, какие ограничения в параметрах файла были им нарушены при его загрузке.

Сообщение об ошибке помогает понять, какие виды файлов недопустимы для загрузки.



Раз мы пытаемся сделать сценарий более надежным, очень хорошо также добавить проверку через суперглобальную переменную $_FILES, не было ли ошибки при загрузке файла.

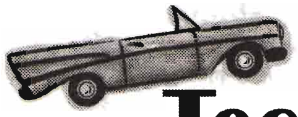
Выведите информативное сообщение об ошибке в том случае, если файл не того типа или имеет слишком большой размер.

```
if (!empty($name) && !empty($score) && !empty($screenshot)) {
  if (($screenshot_type == 'image/gif') || ($screenshot_type == 'image/jpeg') ||
      ($screenshot_type == 'image/pjpeg') || ($screenshot_type == 'image/png')) &&
      ($screenshot_size > 0) && ($screenshot_size <= GW_MAXFILESIZE)) {
    if ($_FILES['screenshot']['error'] == 0) {          // Перемещение файла в постоянный
      // каталог для файлов изображений
      $target = GW_UPLOADPATH . $screenshot;
      if (move_uploaded_file($_FILES['screenshot']['tmp_name'], $target)) {
        // Соединение с базой данных
        $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

        // Запись данных в базу данных
        $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')";
        mysqli_query($dbc, $query);

        // Вывод пользователю подтверждения в получении данных
        echo '<p> Спасибо за то, что добавили свой рейтинг! ' .
             'Он будет просмотрен и добавлен к списку рейтингов как можно скорее.</p>';
        echo '<p><strong>Имя:</strong> ' . $name . '<br />';
        echo '<strong>Рейтинг:</strong> ' . $score . '<br />';
        echo '<img src="' . GW_UPLOADPATH . $screenshot .
          '" alt="Изображение, подтверждающее подлинность рейтинга " /></p>';
        echo '<p><a href="index.php">&lt;&lt; Назад к списку рейтингов </a></p>';

        // Очистка полей ввода данных формы
        $name = "";
        $score = "";
        $screenshot = "";

        mysqli_close($dbc);
      }
      else {
        echo '<p class="error"> Извините, возникла ошибка при загрузке файла изображения.</p>';
      }
    }
  }       else {          echo '<p class="error">Файл, подтверждающий рейтинг, должен' .
                            'быть файлом изображения в форматах GIF, JPEG или PNG,' .
                            'и его размер не должен превышать ' .
                            (GW_MAXFILESIZE / 1024) . ' Кб.</p>';        }

  // Попытка удалить временный файл изображения, подтверждающий рейтинг пользователя
  @unlink($_FILES['screenshot']['tmp_name']);
}
else {
  echo '<p class="error">Введите, пожалуйста, всю информацию для добавления вашего
       рейтинга.</p>';
```

addscore.php

В результате вызова функции unlink() файл удаляется с веб-сервера. Мы подавляем ее сообщение об ошибке с помощью символа @, если загрузка файла не удастся.

Новый и улучшенный сценарий «Добавь свой рейтинг» теперь проводит проверку загружаемых файлов изображений, подтверждающих рейтинги.



Тест-драйв

Добавьте в сценарий «Добавь свой рейтинг» проверку файлов изображений, подтверждающих рейтинги, при загрузке их на сервер.

Внесите в файл `addscore.php` изменения, касающиеся проверки файлов изображений, подтверждающих рейтинги, при загрузке их на сервер. Загрузите откорректированный сценарий на ваш веб-сервер и проверьте работу формы «Добавь свой рейтинг» как с допустимыми, так и с недопустимыми файлами изображений (размер файла больше допустимого максимума и файл недопустимого формата).

не бывает глупых вопросов

В: Почему для изображений JPEG используются два типа MIME?

О: Этот вопрос лучше задать разработчикам браузеров, которые по каким-то причинам решили использовать разные типы `MIME` для `JPEG`-изображений. Для того чтобы быть уверенным, что проверка файла `JPEG` будет проводиться на большем, по возможности, количестве разных браузеров, необходимо проверять соответствие файла обоим типам.

В: Какая необходимость в проверке того, что размер файла изображения больше 0 байт? Разве не все файлы изображений больше 0 байт?

О: Теоретически — да. Но технически вполне возможно создать на сервере файл нулевой длины. Например, такое произойдет, если пользователь укажет для загрузки на сервер имя файла, который не существует на локальном компьютере. Для предотвращения подобных инцидентов сценарий `addscore.php` ведет себя осторожно и проверяет, не является ли файл пустым.

В: Почему переменная GW_MAXFILESEZE помещена в файл appvars.php, хотя она используется только в сценарии addscore.php?

О: Хотя и справедливо утверждение, что файл `appvars.php` необходим только для того, чтобы размещать в нем данные, которые совместно используются несколькими сценариями, но это также хорошее место для инициализации любых констант. В этом случае размещение константы `GW_MAXFILESEZE` в файле `appvars.php` облегчает ее поиск, если вы захотите увеличить максимально допустимый размер загружаемого файла изображения, подтверждающего рейтинг пользователя.

В: Что делает строка кода @unlink()?

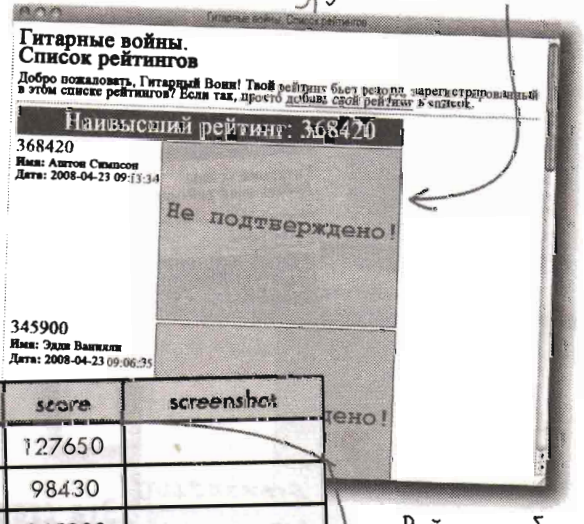
О: Встроенная PHP-функция `unlink()` удаляет файл на сервере: в нашем случае — подтверждающего рейтинг пользователя временный файл изображения, который был загружен. Так как не исключено, что загрузка файла не удалась и на сервере нет временного файла изображения (удалять нечего), мы подавляем любое сообщение об ошибке, выводимое функцией `unlink()`, путем введения перед ее именем символа `@`. Вы можете ставить этот символ перед любой PHP-функцией, чтобы подавить ее возможное сообщение об ошибке.

А что делать с этими неподтвержденными рейтингами? Они, знаете ли, должны быть удалены.

Список рейтингов должен быть расчищен.

Хотя загрузка файлов на сервер стала более надежной благодаря их проверке, мы больше не можем игнорировать проблему неподтвержденных рейтингов. Новый рейтинг с загруженным файлом изображения не должен играть партию второй скрипки наряду со старыми рейтингами без такого изображения, которые могут не быть подлинными. Приложению «Гитарные войны» необходим способ, позволяющий удалять старые рейтинги.

Текущий наивысший рейтинг не подтвержден, что не внушает большого доверия другим пользователям.



guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йоханссон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Пайрстон	186580	phizsscore.gif

Рейтинги без файлов изображений, подтверждающих их достоверность, необходимо удалить из базы данных немедленно!

Напишите, что бы вы сделали, чтобы удалить неподтвержденные рейтинги из списка:

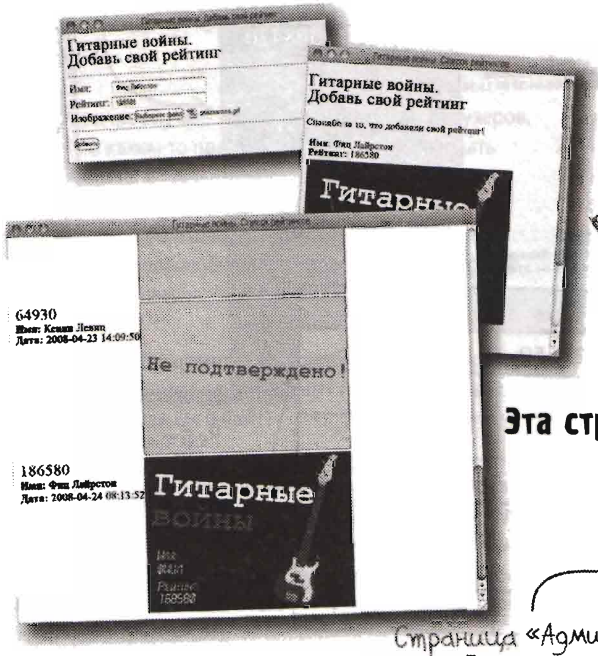
План создания страницы «Админ»

Так как нам всего лишь необходимо удалить из базы данных неподтвержденные рейтинги, вполне резонно было бы запустить инструментальную программу SQL и вручную удалить записи из таблицы, используя несколько запросов DELETE. Но вполне возможно, что это не последний раз, когда вам приходится удалять записи. Кроме того, не слишком весело обращаться к инструментальным программам SQL каждый раз, когда вам необходимо техническое обслуживание вашего веб-приложения. Главная мысль заключается в том, что создаваемое приложение должно предоставлять возможность проводить его техническое обслуживание с минимумом препятствий.

Что нам необходимо, так это страница, к которой имел бы доступ только веб-администратор и которая позволяла бы удалять рейтинги... Страница «Админ»! Но нам нужно быть очень внимательными и четко различать, какие части приложения относятся к пользователю, а какие — к администратору.

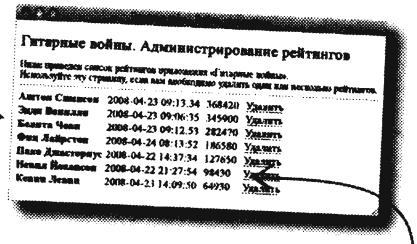
Веб-приложения часто включают и страницы с доступом для всех, и страницы с доступом для администратора, которые используются только для технической поддержки сайта.

Страницы, предназначенные для пользователей



Страница «Добавь свой рейтинг» и главная страница «Гитарных войн» разработаны для того, чтобы пользователь мог загрузить на сервер свой рейтинг и просматривать список всех рейтингов.

Эта страница только для администратора сайта:



Страница «Админ» разработана только для использования администратором сайта. Вряд ли вы захотите, чтобы пользователь имел возможность удалять рейтинги.

Нажатие кнопки с независимой фиксацией «Удалить» приведет к удалению соответствующей записи.

Напишите, что необходимо делать сценариям «Адмін» и «Удалить рейтинг» для того, чтобы обеспечить приложению «Гитарные войны» функцию удаления рейтингов. Затем нарисуйте схему, как удаление рейтингов повлияет на записи таблицы `guitarwars` и файлы изображений, подтверждающих рейтинги, связанные с ними.

admin.php

Веб-сервер

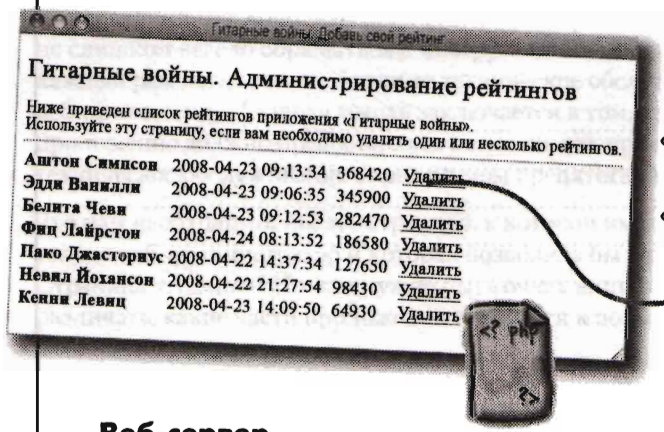
removescore.php

`guitarwars`

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йоханссон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif

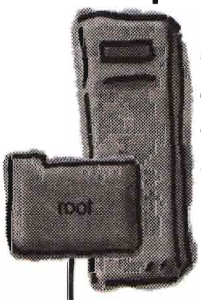
РЕШЕНИЕ К УПРАЖНЕНИЮ

Напишите, что необходимо делать сценариям «Админ» и «Удалить рейтинг» для того, чтобы обеспечить приложению «Гитарные войны» функцию удаления рейтингов. Затем нарисуйте схему, как удаление рейтингов повлияет на записи таблицы `guitarwars` и файлы изображений, подтверждающих рейтинги, связанные с ними.



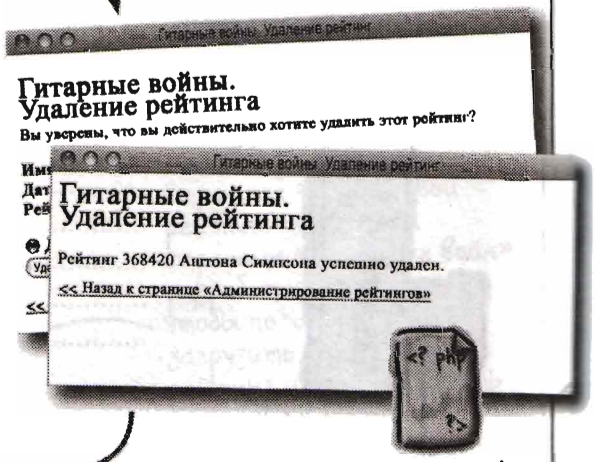
admin.php

Сценарий admin.php перечисляет рейтинги всех пользователей с гиперссылкой «Удалить» для каждой строки рейтинга. Гиперссылка указывает на сценарий «Удалить рейтинг».



Веб-сервер

Сценарий removescore.php обеспечивает фактическое удаление рейтинга из базы данных, удаление файла изображения, доказывающего его подлинность, и выводит сообщение, подтверждающее эти действия.



removescore.php

guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif

Хотя у этого конкретного рейтинга отсутствовал файл изображения, подтверждающего его подлинность, сценарию «Удалить рейтинг» необходимо удалять с сервера такой файл для рейтингов, у которых он присутствует.

Создание на странице «Админ» гиперссылки для удаления рейтинга

Хотя ответственность за непосредственное удаление рейтинга лежит на сценарии «Удалить рейтинг», нам необходим также и сценарий «Админ», который предоставит возможность выбрать рейтинг для удаления.

Сценарий `admin.php` перечисляет рейтинги всех пользователей с гиперссылкой «Удалить» для каждой строки рейтинга. По этой гиперссылке сценарию «Удалить рейтинг» передаются данные о рейтинге, который должен быть удален.

admin.php

```
<?php
  require_once('appvars.php');
  require_once('connectvars.php');

  // Соединение с базой данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  // Извлечение данных из базы данных MySQL
  $query = "SELECT * FROM guitarwars ORDER BY score DESC, date ASC";
  $data = mysqli_query($dbc, $query);

  // Извлечение данных из массива рейтингов в цикле.
  // Форматирование данных записей в виде кода HTML
  echo '<table>';
  while ($row = mysqli_fetch_array($data)) {
    // Вывод данных рейтинга
    echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
    echo '<td>' . $row['date'] . '</td>';
    echo '<td>' . $row['score'] . '</td>';
    echo '<td><a href="removescore.php?id=' . $row['id'] . '&amp;date=' . $row['date'] .
         '&amp;name=' . $row['name'] . '&amp;score=' . $row['score'] .
         '&amp;screenshot=' . $row['screenshot'] . '">Удалить</a></td></tr>';
  }
  echo '</table>';

  mysqli_close($dbc);
?>
```

URL сценария «Удалить рейтинг» — это не просто адрес сценария... Он также включает данные, передаваемые ему для обработки!

Этот код генерирует HTML-гиперссылку на сценарий removescore.php, передавая ему данные о рейтинге, который должен быть удален.

```
<a href="removescore.php?id=5&date=2008-
04-23%2009:13:34&name=Ashton%20
Simpson&score=368420&screenshot="
```

Сценарии могут обмениваться друг с другом информацией

Для того чтобы удалить рейтинг из списка, сценарий «Удалить рейтинг» должен знать, какой рейтинг необходимо убрать. Но это решение принимается в сценарии «Адмін». Это влечет за собой вопрос: как сценарий «Адмін» сообщит сценарию «Удалить рейтинг», какой именно рейтинг он должен удалить? Такой обмен информацией может быть осуществлен путем добавления данных о рейтинге, который должен быть удален, в строку URL атрибута `href` в тегах `<a ...>Удалить</a>` для каждого из рейтингов, перечисленных на странице «Адмін». Если вы внимательно проанализируете URL для всех рейтингов, вы заметите, что в каждом из них записаны данные соответствующего рейтинга.

URL сценария может быть использован для обмена данными в виде запроса GET.

```
<a href="removescore.php?
  id=5&
  date=2008-04-23%2009:13:34&
  name=%d0%90%d1%88%d1%82%d0%be%d0%bd%20
%d0%a1%d0%b8%d0%bc%d0%bf%d1%81%d0%be%d0%bd%0a&
  score=368420&
  screenshot=
```

Гитарные войны. Добавь свой рейтинг

Гитарные войны. Администрирование рейтингов

Ниже приведен список рейтингов приложения «Гитарные войны». Используйте эту страницу, если вам необходимо удалить один или несколько рейтингов.

Аштон Симпсон	2008-04-23 09:13:34	368420	Удалить
Эдди Ванилли	2008-04-23 09:06:35	345900	Удалить
Белита Чеви	2008-04-23 09:12:53	282470	Удалить
Фиц Лайрстон	2008-04-24 08:13:52	186580	Удалить
Пако Джасториус	2008-04-22 14:37:34	127650	Удалить
Невил Йохансон	2008-04-22 21:27:54	98430	Удалить
Кения Левиц	2008-04-23 14:09:50	64930	Удалить

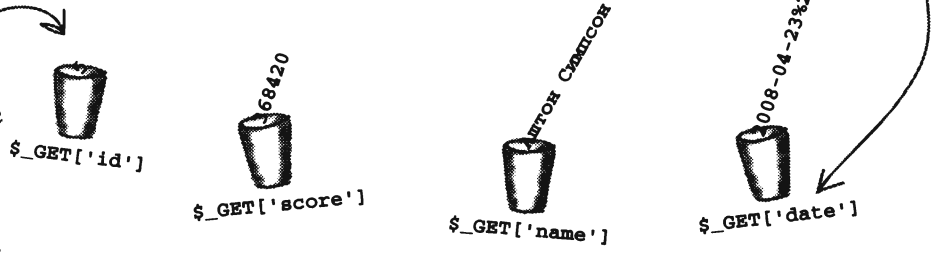
Данные передаются в виде пары: имя параметра и его значение. Каждая пара отделена от другой знаком амперсанда (&).

Атрибут href тега <a> (URL) ссылается на сценарий removescore.php, но содержит также данные о рейтинге, который должен быть удален.

Щелчок кнопкой мыши на этой гиперссылке не только открывает сценарий «Удалить рейтинг», но также отправляет данные, необходимые вызываемому сценарию, в виде запроса GET.

Итак, данные переданы с помощью URL, но как сценарий «Удалить рейтинг» сможет воспользоваться ими? Доступ к данным, переданным сценарию с использованием URL, может быть получен через суперглобальный массив `$_GET`, который очень похож на суперглобальный массив `$_POST`. Добавление данных в URL гиперссылки является тем же самым, что и использование запроса GET в веб-форме. В традиционном HTML-запросе GET данные формы **автоматически** отправляются сценарию, обрабатывающему данные этой формы, как часть URL. Мы делаем то же самое, создавая **вручную** наш собственный запрос GET в виде индивидуального URL.

Имя параметра используется для получения доступа к элементу массива `$_GET`, содержащего значение этого параметра.



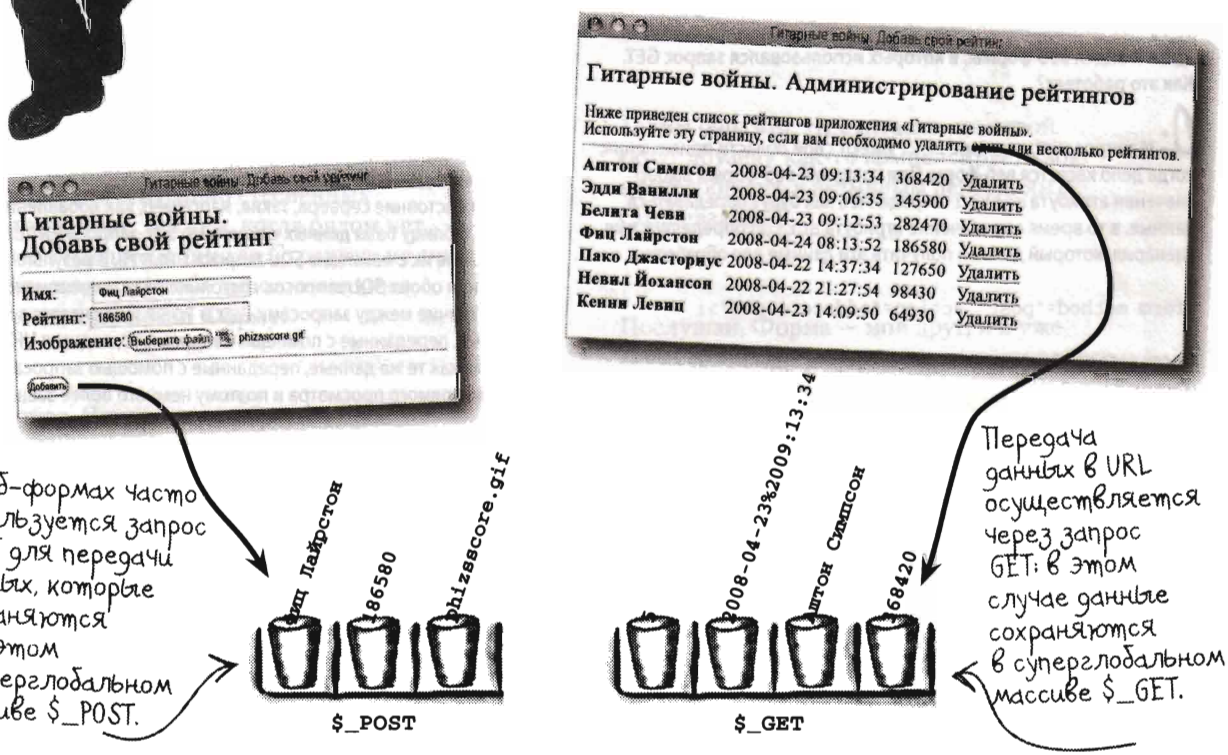
URL сценария используется как удобный способ передать важные данные: такие, например, как идентификатор записи таблицы.

Я не пойму, чем вызван весь этот ажиотаж вокруг GET. Почему мы просто не можем передать данные сценарию, используя POST, как мы делали это до сих пор?

Запрос POST может быть инициирован только через форму, в то время как GET может быть создан в виде URL.

До сих пор мы отправляли сценарию данные через веб-форму, в которой сценарий указывался как атрибут `action` для кнопки «Отправить». После того как пользователь заполнил поля данных формы и нажал кнопку «Отправить», данные формы отправлялись на сервер в виде запроса `POST`.

Проблема в том, что страница «Админ» не использует форму для инициирования сценария «Удалить рейтинг». Она просто ссылается на этот сценарий через URL. Поэтому, для того чтобы отправить данные сценарию на обработку, нам не остается ничего другого, кроме как добавить эти данные в качестве параметров в URL. Именно здесь очень удобно применение запроса `GET`, так как он обеспечивает доступ к данным, добавленным к URL в виде параметров. Так же как и при использовании запроса `POST`, данные, переданные сценарию через запрос `GET`, доступны в виде элементов суперглобального массива, только его имя `$_GET` вместо `$_POST`.



О GET и POST

Разница между запросами GET и POST не сводится просто к разнице в механизмах передачи данных с формой или URL. Действительная разница заключается в разнице в **намерениях**, с которыми делается тот или иной запрос. Запрос GET обычно используется для получения данных с сервера, при этом **состояние сервера остается без изменения**. С другой стороны, запрос POST используется обычно для передачи данных на сервер, что влечет за собой **изменение в состоянии сервера** как результат обработки им полученных таким образом данных.

Два типа веб-запросов, GET и POST, управляют способом обмена данными между сценариями.

POST

Используется для передачи данных на сервер, что влечет за собой какие-либо изменения в состоянии сервера, например добавление данных в базу. Данные также могут быть возвращены в ответе. В отличие от запроса GET запрос POST может быть сделан только с использованием атрибута веб-формы action. Кроме того, в отличие от запроса GET данные в запросе POST передаются в виде, недоступном для просмотра.

GET

Обычно используется для получения данных без оказания какого-либо влияния на состояние сервера. Для передачи небольших количеств данных также очень удобно использовать запрос GET, встраивая эти данные в URL. В отличие от запроса POST запрос GET находит большее применение при передаче небольших количеств данных.

Не бывает глупых вопросов

В: Я видел веб-формы, в которых использовался запрос GET. Как это работает?

О: И запрос GET, и запрос POST имеют свои области применения, когда дело касается веб-форм. При создании веб-формы выбор значения атрибута method определяет, как будут передаваться данные, в то время как значение атрибута action определяет имя сценария, который должен получить эти данные для обработки:

```
<form method="post" action="addscore.php">
```

После того как для передачи данных формы будет нажата кнопка «Отправить», будет запущен сценарий addscore.php, а данные формы будут переданы ему через суперглобальный массив $_POST. Но вы могли бы с таким же успехом написать тег <form>, как показано ниже, причем в этом случае сценарий сможет получить доступ к данным формы уже через суперглобальный массив $_GET:

```
<form method="get" action="addscore.php">
```

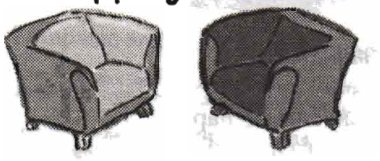
В: Вот так! Значит, не имеет значения, какой из методов запроса использовать, GET или POST?

О: Неверно. Выбор метода имеет значение, и значительное. Запрос GET обычно используется для получения данных без оказания какого-либо влияния на состояние сервера. Поэтому GET является идеальным методом для составления информационного запроса к серверу, в результате выполнения которого состояние сервера остается неизменным. К таким запросам можно отнести, например, извлечение записей данных из таблицы базы данных. С другой стороны, метод POST больше соответствует запросам, которые изменяют состояние сервера, такие, например, как добавление данных в таблицу базы данных с помощью SQL-запроса INSERT или удаление их с помощью SQL-запроса DELETE. В результате выполнения обоих SQL-запросов состояние базы данных изменяется. Другое отличие между запросами GET и POST заключается в том, что данные, переданные с помощью запроса GET, видны в URL, в то время как те же данные, переданные с помощью запроса POST, скрыты от прямого просмотра и поэтому немного более защищены.

В: Как эта разница между запросами GET и POST учитывается при передаче данных через URL?

О: Прежде всего вы можете передать данные сценарию через URL, используя только запрос GET, следовательно, запрос POST отпадает сразу. Более того, раз метод GET предназначен исключительно для запросов, которые не изменяют состояние сервера, значит нет необходимости создавать какие-либо формы, связанные с SQL-запросами INSERT, или DELETE, или чем-нибудь еще, что приводит к изменению состояния сервера в сценариях, которые получают данные через свои URL.

Беседы у камина



Сегодня вечером: **GET и POST**

GET:

Так ты говоришь, болтают, что все, на что я способен, — это только задавать вопросы, но не давать на них ясных ответов. Это правда?

Ладно, это правда, что я действительно не намерен быть причиной каких-либо изменений на сервере, будь то удаление файла или добавление записи в таблицу базы данных. Но это вовсе не означает, что я бесполезен.

Пусть так, но ты все время касаешься только своего друга Формы, в то время как для меня Форма — не более чем случайный знакомый. У меня есть и другие друзья, например URL.

Ладно, тогда у меня к тебе вопрос. Как ты сможешь предпринять какие-либо активные действия в относительно стесненных обстоятельствах, скажем, когда рядом нет Формы? Знаешь ли, иногда Страница не считает нужным создавать себе дополнительные проблемы, привлекая Форму.

Успокойся. Я лишь хочу обратить твое внимание на то, что хотя я и могу только запрашивать данные у сервера, но я очень гибок в методах, которые использую, чтобы выполнить это.

Рад это слышать. Было приятно побеседовать с тобой...

POST:

Конечно. Давай посмотрим на факты: у тебя нет реальных возможностей, кроме того, чтобы только спрашивать сервер о чем-либо.

Все, что я знаю, — это то, что огромное количество дел на сервере не могло бы быть сделано без таких парней, как я, которые как раз и занимаются этим. Если бы сервер оставался все время в неизменном состоянии, вот была бы скукотища!

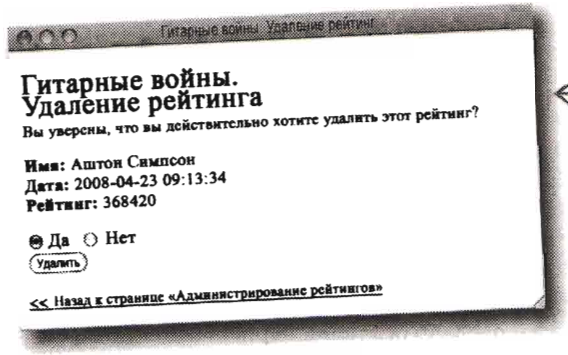
Таким образом, ты считаешь, что твой «круг друзей» как-то компенсирует твою неспособность к активным действиям? Сомневаюсь.

Послушай, Форма — мой друг, и я уже давно принял решение не делать ни одного запроса без его участия. Ты, конечно, можешь осуждать мою преданность, если тебе хочется, но я никогда не предам своего друга.

Ладно, будем считать, что я одобряю твои действия.

GET, POST и удаление рейтинга

Мы установили, что удаление рейтинга в приложении «Гитарные войны» начинается с гиперссылки «Удалить» на странице «Админ», которая указывает на сценарий «Удалить рейтинг». Мы также знаем, что данные рейтинга могут быть отправлены сценарию «Удалить рейтинг» через URL. Но здесь возникает проблема, связанная с тем, что в результате выполнения запроса GET на сервере не должно произойти никаких изменений, в том числе удаления рейтинга. Возможное решение заключается в том, чтобы не производить никаких изменений на сервере... пока. Что если сценарий «Удалить рейтинг» выведет страницу запроса подтверждения до того, как произведет само удаление рейтинга из базы данных?



Вместо того чтобы немедленно удалять рейтинг из базы данных, страница запроса подтверждения дает пользователю возможность подтвердить такое удаление.

Страница запроса подтверждения выводит на экран данные по рейтингу, который должен быть удален, с помощью формы, использующей пару кнопок с зависимой фиксацией «Да» и «Нет». Выбор кнопки «Да» и нажатие кнопки «Удалить» приведет к фактическому удалению рейтинга. В случае выбора кнопки «Нет» удаление не состоится.

Если рассматривать это в понятиях запросов GET и POST, сценарий «Удалить рейтинг» может вывести страницу запроса подтверждения как ответ на запрос GET, сделанный из сценария «Админ». Но так как страница запроса подтверждения сама содержит форму, она может сделать свой собственный запрос POST, когда будет нажата кнопка «Удалить». Если форма ссылается на себя, то этот же самый сценарий (`removescore.php`) сможет обработать запрос POST и выполнить фактическое удаление рейтинга из базы данных. Ниже приведены отдельные этапы этого процесса.

1. **В результате щелчка кнопкой мыши по гиперссылке «Удалить» на странице «Админ» запускается сценарий «Удалить рейтинг», который получает через запрос GET данные об удаляемом рейтинге.**

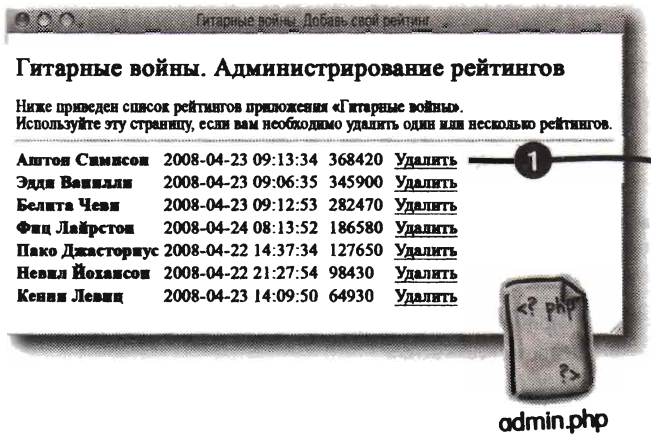
2. **Сценарий «Удалить рейтинг» использует данные об удаляемом рейтинге, переданные ему через суперглобальный массив $_GET, чтобы создать страницу запроса подтверждения с формой.**

3. **Сценарий «Удалить рейтинг» запускается вновь, на этот раз через запрос POST, пользователем, нажавшим кнопку «Удалить» на форме страницы запроса подтверждения.**

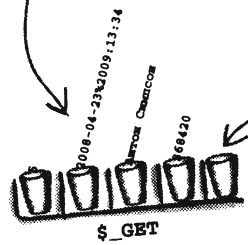
4. **Сценарий «Удалить рейтинг» удаляет рейтинг из базы данных. Кроме того, с веб-сервера удаляется также файл изображения, подтверждающего рейтинг.**

Имеется возможность, и даже рекомендуется в некоторых случаях одному и тому же сценарию отвечать на оба запроса: GET и POST.

Давайте посмотрим поэтапно, как разворачиваются события в процессе удаления рейтинга из базы данных...



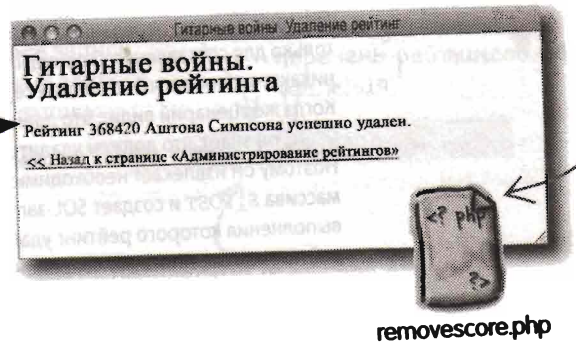
Запускается сценарий «Удалить рейтинг», который получает через запрос GET данные об удаляемом рейтинге, добавленные к URL.

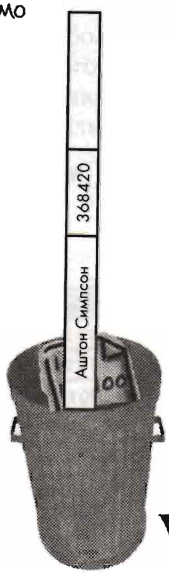


Данные колонки screenshot для этого рейтинга отсутствуют.

Запрос POST используется для запуска сценария «Удалить рейтинг» (опять!) и передачи данных о рейтинге, который необходимо удалить.

Один и тот же сценарий реагирует по-разному в зависимости от того, получил он запрос GET или POST.





Сценарий «Удалить рейтинг» удаляет рейтинг из базы данных и файл изображения, подтверждающего рейтинг, с веб-сервера.

Не бывает глупых вопросов

В: Как может один и тот же сценарий обрабатывать и запрос GET, и запрос POST?

О: Это связано с тем, как он вызывается. Сценарий «Удалить рейтинг» вызывается двумя различными способами. В первом случае вызов происходит в результате щелчка кнопкой мыши по гиперссылке «Удалить» на странице «Админ», при этом все необходимые сценарию данные добавляются к URL. Так как данные добавлены к URL, запрос рассматривается как `GET`. В результате обработки этого запроса `GET` сценарий генерирует веб-форму, атрибут `action` тега `<form>` которой ссылается на этот же самый сценарий. Поэтому, когда пользователь нажмет кнопку формы «Удалить», сценарий вызывается во второй раз. Но в отличие от первого вызова URL уже не выглядит столь необычно (с данными, добавленными к адресу сценария), поэтому запрос не рассматривается как `GET`. Теперь это уже запрос `POST`, и данные рейтинга посылаются на сервер через суперглобальный массив `$_POST`.

В: Значит способ, с помощью которого вызывается сценарий, определяет, что конкретно он будет выполнять?

О: Да! Когда сценарий видит, что данные переданы ему в составе URL в виде запроса `GET`, он знает, что должен вывести страницу подтверждения, а не удалять какие-либо данные из базы. Поэтому данные, переданные в составе массива `$_GET`, используются только для составления страницы подтверждения и не оказывают никакого влияния на состояние сервера.

Когда же сценарий видит, что данные переданы ему в составе запроса `POST`, он знает, что должен удалить запись из таблицы базы данных. Поэтому он извлекает необходимые данные из суперглобального массива `$_POST` и создает SQL-запрос `DELETE FROM`, в результате выполнения которого рейтинг удаляется. А так как большинству рейтингов соответствуют сохраненные на сервере файлы изображений, подтверждающих эти рейтинги, то вместе с удалением из базы данных рейтинга удаляется также и соответствующий ему файл изображения с веб-сервера.

Определение рейтинга для удаления

После того как составлена схема процесса удаления рейтинга, мы можем сосредоточить свое внимание на той стороне приложения, которая связана с базой данных. Сценарий «Удалить рейтинг» отвечает за удаление рейтинга, что означает удаление записи из таблицы рейтингов базы данных. Если вы помните, SQL-запрос `DELETE FROM` позволяет нам убрать запись из таблицы. Но прежде чем удалять запись, мы должны определить ее. Это достигается путем включения условного выражения `WHERE` в запрос `DELETE FROM`. Например, в результате этого запроса исчезнет запись, в которой колонка `name` имеет значение «Аштон Симпсон»:

В результате выполнения этого запроса будет удалена запись, в которой колонка name имеет значение «Аштон Симпсон».

```
DELETE FROM guitarwars WHERE name = 'Аштон Симпсон'
```

После ключевых слов DELETE FROM должно следовать имя таблицы, чтобы было известно, из какой таблицы нужно удалить данные.

guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif

Имя пользователя применяется для указания, какие рейтинги должны быть удалены.

Однако с этим запросом связаны определенные проблемы. В мире, в котором живут миллионы гитарных воинов, есть вероятность существования более чем одного Аштона Симсона. В результате выполнения этого запроса будет удалена не одна запись, а **все** записи, в которых колонка `name` имеет значение «Аштон Симпсон». В запросе должно быть больше информации, чтобы удалить нужную запись:

Использование значения рейтинга в дополнение к имени уточняет перечень рейтингов для удаления.

```
DELETE FROM guitarwars WHERE name = 'Аштон Симпсон' AND score = '368420'
```

Логический оператор AND определяет, что запись будет отвечать требованиям условного выражения запроса тогда, когда и имя, и рейтинг соответствуют заданным значениям.

guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif

Теперь, когда и имя, и рейтинг должны соответствовать заданным значениям, вероятность случайного удаления более чем одной записи существенно уменьшилась.

Установка предельного количества удаляемых записей с помощью выражения LIMIT

Для повышения безопасности установите предельное количество записей, которые могут быть удалены.

Использование значений и колонки `name`, и колонки `score` как основы для определения записи на удаление — это хорошо... но не слишком. При разработке приложений необходимо любой ценой стремиться к уменьшению риска, а здесь все еще остается небольшой риск удаления нескольких записей с одинаковыми именами и значениями рейтинга. Решение этой проблемы заключается в том, чтобы ограничить их количество **одной записью**. Это может быть сделано с помощью выражения `LIMIT` (предел):

```
DELETE FROM guitarwars WHERE name = 'Аштон Симпсон' AND score = '368420' LIMIT 1
```

Число, следующее после ключевого слова `LIMIT` устанавливает количество удаляемых записей (в данном случае одну). Таким образом гарантируется, что в результате выполнения этого запроса будет удалено не более одной записи. Что, если существует две записи с именами «Аштон Симпсон» и одинаковыми рейтингами. Конечно, это маловероятно, но при тщательной разработке приложения имеет смысл рассматривать и такие ситуации.

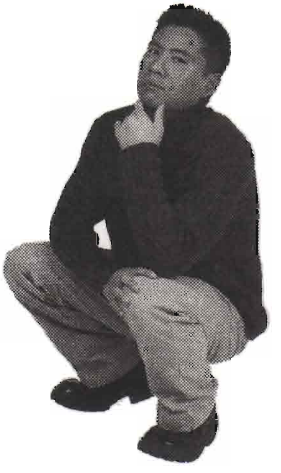
guitarwars

id	date	name	score	screenshot
1	2008-04-22 14:37:34	Пако Джасториус	127650	
2	2008-04-22 21:27:54	Невил Йохансон	98430	
3	2008-04-23 09:06:35	Эдди Ванилли	345900	
4	2008-04-23 09:12:53	Белита Чеви	282470	
5	2008-04-23 09:13:34	Аштон Симпсон	368420	
6	2008-04-23 14:09:50	Кенни Левиц	64930	
7	2008-04-24 08:13:52	Фиц Лайрстон	186580	phizsscore.gif
...				
523	2008-11-04 10:03:21	Аштон Симпсон	368420	ashtonsscore.jpg

Два рейтинга с одинаковыми именами и значениями рейтинга являются проблемой для нашего запроса DELETE.

Напишите, что произойдет с этой таблицей, когда будет выполнен приведенный выше запрос DELETE. Как сделать, чтобы удалялся именно тот из двух рейтингов Аштона Симпсона, который должен быть удален?

А не лучше ли использовать идентификатор рейтинга в условном выражении WHERE запроса DELETE FROM? Это могло бы помочь в обретении уверенности в том, что мы удаляем именно тот рейтинг, который должен быть удален, не так ли?



Да, это обеспечит такую уверенность! Идентификатор — это самый лучший способ определить рейтинг для удаления.

Обеспечение уникальности — одна из главных целей в создании первичных ключей для ваших таблиц. Колонка `id` таблицы `guitarwars` является первичным ключом, поэтому ее значение является уникальным для всех записей таблицы. При использовании этой колонки в условном выражении `WHERE` SQL-запроса `DELETE FROM` мы устраняем какие-либо сомнения, касающиеся вопроса о том, какая запись должна быть удалена. Ниже приведен новый SQL-запрос, в условном выражении которого используется колонка `id` для обеспечения уникальности:

```
DELETE FROM guitarwars WHERE id = 5
```

Удаление данных, базирующееся на первичном ключе, позволяет повысить точность определения записи, предназначенной для удаления.

Уверенность в том, что колонка `id` действительно является первичным ключом, влечет за собой уверенность в том, что в результате выполнения этого запроса будет удалена только одна запись. А что если вы не создавали эту базу данных и уникальность не была реализована правильно? Тогда будет иметь смысл использование выражения `LIMIT`. Логическим обоснованием этому может служить следующее соображение: если вам необходимо создать SQL-запрос, который должен воздействовать только на одну запись, выразите это в запросе в явном виде.

```
DELETE FROM guitarwars WHERE id = 5 LIMIT 1
```

Выражение LIMIT устанавливает в явном виде то, что в результате выполнения запроса не может быть удалено более одной записи.

Никогда не вредно выражать в явном виде то, что вы ожидаете получить в результате выполнения SQL-запроса, и в этом случае выражение `LIMIT` обеспечивает дополнительную степень безопасности запроса `DELETE`.

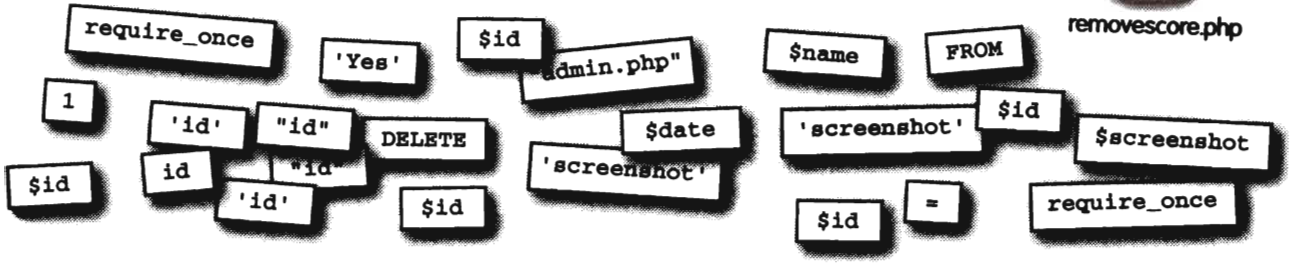
Магниты PHP и MySQL

Сценарий `removescore.php` почти закончен, но в нем не хватает некоторых важных фрагментов кода. Используя магниты, добавьте эти фрагменты и предоставьте приложению «Гитарные войны» возможность истреблять ненужные рейтинги.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ru" lang="ru">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Гитарные войны. Удаление рейтинга</title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2> Гитарные войны. Удаление рейтинга </h2>

<?php
.....('appvars.php');

.....('connectvars.php');

  if (isset($_GET['id']) && isset($_GET['date']) && isset($_GET['name']) &&
      isset($_GET['score']) && isset($_GET[.....])) {
    // Извлечение данных рейтинга из суперглобального массива $_GET
    $id = $_GET['id'];
    $date = $_GET['date'];
    $name = $_GET['name'];
    $score = $_GET['score'];
    .....= $_GET[.....];
  }
  else if (isset($_POST['id']) && isset($_POST['name']) && isset($_POST['score'])) {
    // Извлечение данных рейтинга из суперглобального массива $_POST
    .....= $_POST[.....];
    $name = $_POST['name'];
    $score = $_POST['score'];
  }
  else {
    echo '<p class="error">Извините, ни одного рейтинга не выбрано для удаления.</p>';
  }

  if (isset($_POST['submit'])) {
    if ($_POST['confirm'] ==.....) {
      // Удаление с сервера файла изображения,
      // подтверждающего рейтинг
      @unlink(GW_UPLOADPATH . $screenshot);

      // Соединение с базой данных
      $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
```

```
    // Удаление рейтинга из базы данных
    $query = "..... .....guitarwars WHERE.....LIMIT.....";
    mysqli_query($dbc, $query);
    mysqli_close($dbc);

    // Вывод пользователю страницы подтверждения
    echo '<p>Рейтинг со значением ' . $score . ' для пользователя ' .
        $name . ' был успешно удален из базы данных.</p>';
  }
  else {
    echo '<p class="error"> Рейтинг не удален.</p>';
  }
}
else if (isset(.....) && isset(.....) && isset(.....) &&
        isset($score) && isset($sreenshot)) {
  echo '<p>Вы уверены, что хотите удалить этот рейтинг?</p>';
  echo '<p><strong>Имя: </strong>' . $name . '<br /><strong>Дата: </strong>' . $date .
    '<br /><strong>Рейтинг: </strong>' . $score . '</p>';
  echo '<form method="post" action="removescore.php">';
  echo '<input type="radio" name="confirm" value="Да" /> Да ';
  echo '<input type="radio" name="confirm" value="Нет" checked="checked" /> Нет <br />';
  echo '<input type="submit" value="Удалить" name="submit " />';
  echo '<input type="hidden" name="id" value="' . $id . '" />';
  echo '<input type="hidden" name=.....value="' ..... '" />';
  echo '<input type="hidden" name="score" value="' . $score . '" />';
  echo '</form>';
}

echo '<p><a href=.....>&lt;&lt; Назад к списку рейтингов </a></p>';
?>

</body>
</html>
```

removescore.php



Магниты PHP и MySQL

Сценарий `removescore.php` почти закончен, но в нем не хватает некоторых важных фрагментов кода. Используя магниты, добавьте эти фрагменты и предоставьте приложению «Гитарные войны» возможность истреблять ненужные рейтинги.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ru" lang="ru">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Гитарные войны. Удаление рейтинга</title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2> Гитарные войны. Удаление рейтинга </h2>

<?php
  require_once 'appvars.php');
  require_once 'connectvars.php');

  if (isset($_GET['id']) && isset($_GET['date']) && isset($_GET['name']) &&
      isset($_GET['score']) && isset($_GET['screenshot'])) {
    // Извлечение данных рейтинга из суперглобального массива $_GET
    $id = $_GET['id'];
    $date = $_GET['date'];
    $name = $_GET['name'];
    $score = $_GET['score'];
    $screenshot = $_GET['screenshot'];
  }
  else if (isset($_POST['id']) && isset($_POST['name']) && isset($_POST['score'])) {
    // Извлечение данных рейтинга из суперглобального массива $_POST
    $id = $_POST['id'];
    $name = $_POST['name'];
    $score = $_POST['score'];
  }
  else {
    echo '<p class="error">Извините, ни одного рейтинга не выбрано для удаления.</p>';
  }

  if (isset($_POST['submit'])) {
    if ($_POST['confirm'] == 'Yes') {
      // Удаление с сервера файла изображения,
      // подтверждающего рейтинг
      @unlink(GW_UPLOADPATH . $screenshot);

      // Соединение с базой данных
      $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
```

Включение файлов сценариев с общими данными. При этом используется выражение require_once.

В сценарии выполняются разные фрагменты кода в зависимости от того, в результате какого запроса (GET или POST) он был вызван.

PHP-директива @, подавляющая вывод сообщения об ошибке. Это необходимо при вызове функции unlink(), потому что мы можем вызвать ее для удаления несуществующего файла. В этом случае мы не хотим, чтобы пользователь видел это сообщение.

Сценарий может быть использован для удаления любого рейтинга, поэтому файл изображения, его подтверждающего, должен быть также удален, и это составная часть общего процесса удаления рейтинга.

Удаляется запись только с заданным значением колонки id. Кроме того, количество удаляемых записей ограничивается одной записью.

```
    // Удаление рейтинга из базы данных
    $query = "DELETE FROM bitarwars WHERE id = $id LIMIT 1";
    mysqli_query($dbc, $query);
    mysqli_close($dbc);

    // Вывод пользователю страницы подтверждения
    echo '<p>Рейтинг со значением ' . $score . ' для пользователя ' .
        $name . ' был успешно удален из базы данных.</p>';
  }
  else {
    echo '<p class="error"> Рейтинг не удален.</p>';
  }
}
else if (isset($id) && isset($name) && isset($date) &&
    isset($score) && isset($screenshot)) {
  echo '<p>Вы уверены, что хотите удалить этот рейтинг?</p>';
  echo '<p><strong>Имя: </strong>' . $name . '<br /><strong>Дата: </strong>' . $date .
    '<br /><strong>Рейтинг: </strong>' . $score . '</p>';
  echo '<form method="post" action="removescore.php">';
  echo '<input type="radio" name="confirm" value="Да" /> Да ';
  echo '<input type="radio" name="confirm" value="Нет" checked="checked" /> Нет <br />';
  echo '<input type="submit" value="Удалить" name="submit " />';
  echo '<input type="hidden" name="id" value="' . $id . '" />';
  echo '<input type="hidden" name="id" value="' . $id . '" />';
  echo '<input type="hidden" name="score" value="' . $score . '" />';
  echo '</form>';
}

echo '<p><a href="admin.php">&lt;&lt; Назад к списку рейтингов </a></p>';
?>

</body>
</html>
```

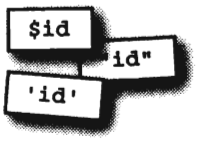
removescore.php

Страница запроса подтверждения выводится только в том случае, если все эти переменные имеют непустые значения.

Мы не используем здесь переменную $_SERVER['PHP_SELF'], потому что она содержит все данные, включенные в URL при вызове сценария в результате выполнения запроса GET. Мы должны быть уверены, что никакие данные не передаются в составе URL, когда сценарий вызывается с помощью запроса POST.

Несколько скрытых полей используются для передачи данных как часть запроса POST.

Гиперссылка на страницу «Админ» облегчает возвращение к ней.

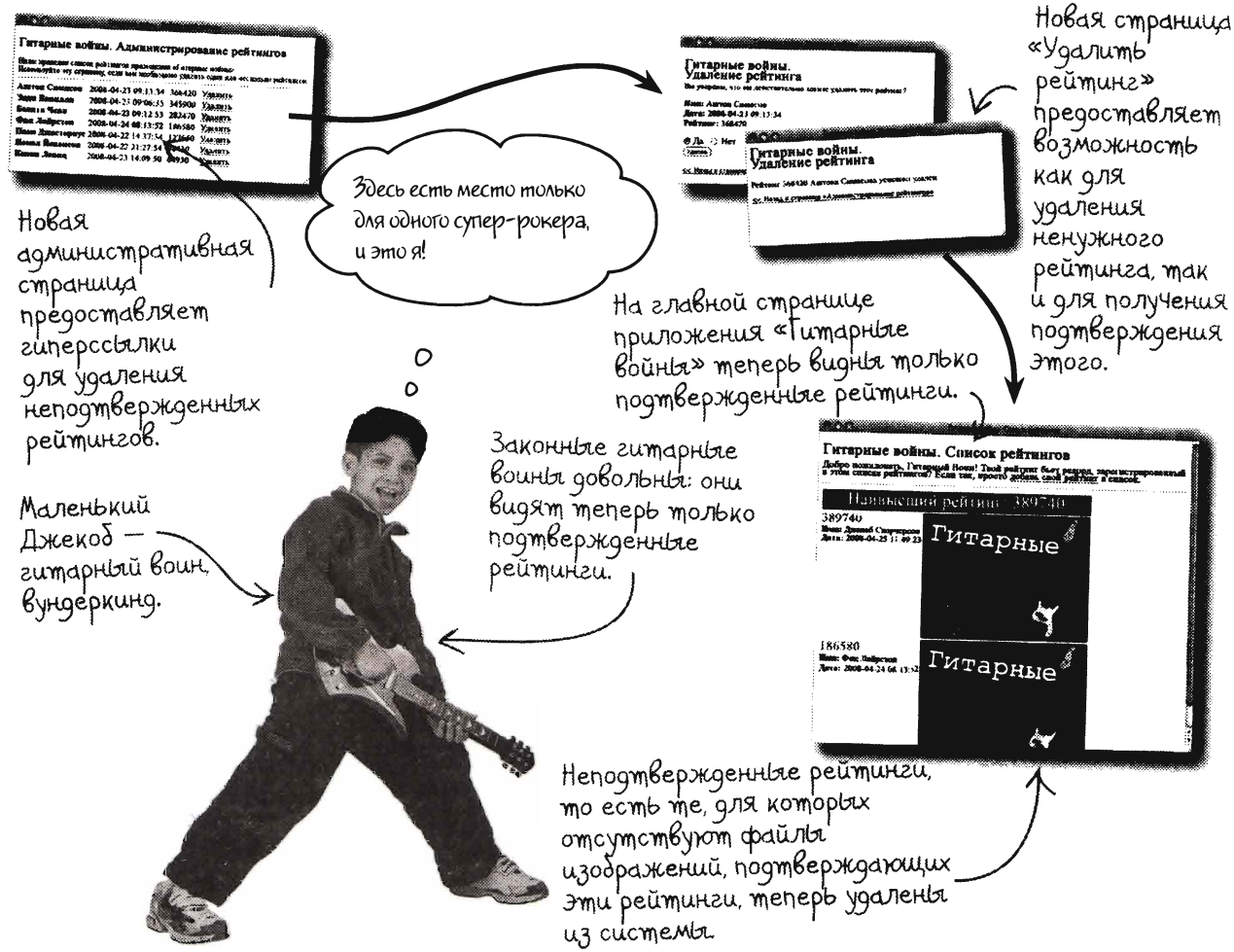


Небольшое количество магнитов осталось неиспользованным.

Тест-драйв

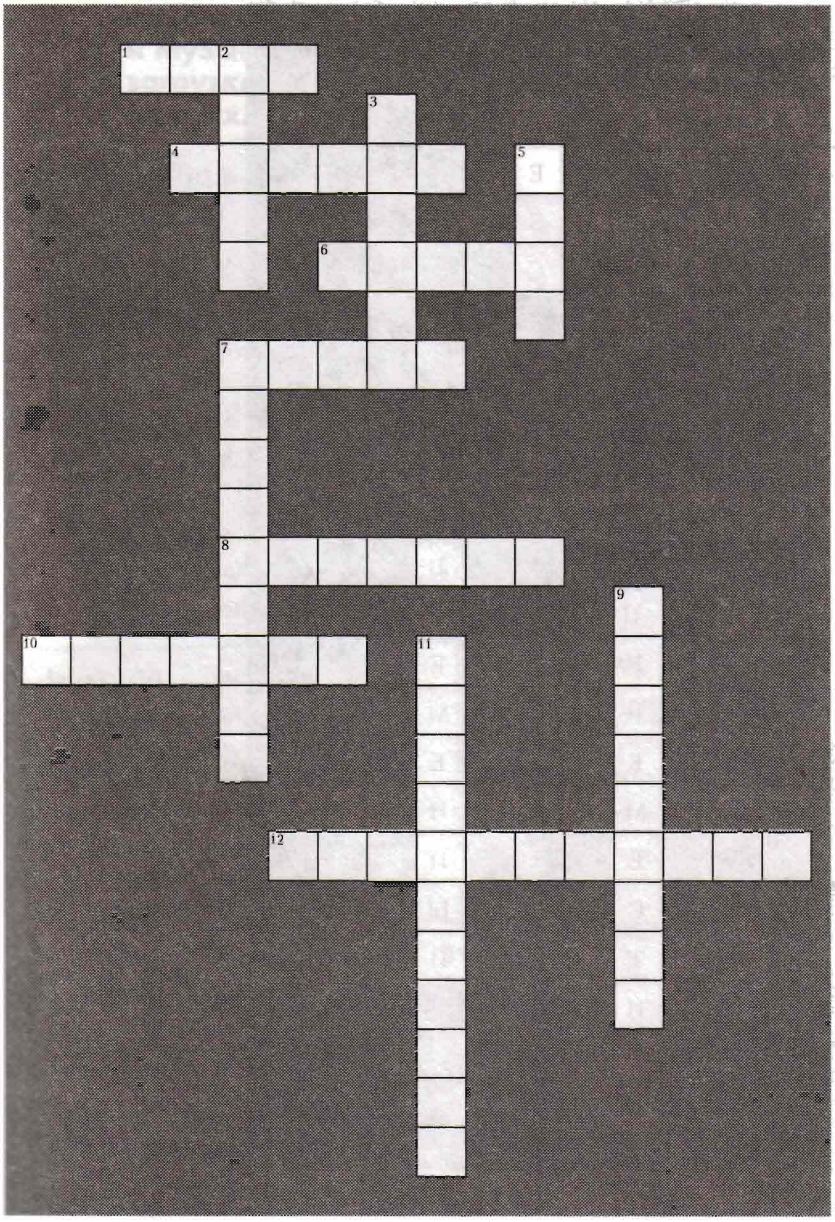
Добавьте сценарии «Удалить рейтинг» и «Админ» к приложению «Гитарные войны», чтобы появилась возможность удалять рейтинги.

Создайте два новых текстовых файла: `removescore.php` и `admin.php`, добавьте в них код, который мы с вами только что рассмотрели. Загрузите все эти файлы на ваш веб-сервер и откройте сценарий `admin.php` в браузере. Щелкните кнопкой мыши на гиперссылке «Удалить» для того рейтинга, который вы хотите удалить, и затем подтвердите удаление этого рейтинга в форме страницы подтверждения «Удалить рейтинг». Вернитесь на административную страницу, чтобы убедиться, что рейтинг, о котором шла речь, действительно удален. После этого откройте главную страницу приложения «Гитарные войны» (`index.php`) и посмотрите, какие изменения произошли на ней.



Кроссворд PHP и MySQL

Устали загружать файлы на сервер? А как насчет того, чтобы загрузить кое-какие знания в квадратики, собранные в головоломку?



По горизонтали

1. Это значение должно быть присвоено атрибуту `type` тега `<input>`, чтобы создать поле ввода имени загружаемого файла.

4. Разработчики веб-приложений обычно сохраняют на сервере файлы изображений в каталоге с именем _____.

6. Информация о загруженных файлах сохраняется в суперглобальном массиве $_____.

7. Выражение с этим ключевым словом используется в SQL-запросе, чтобы изменить структуру таблицы базы данных.

8. Выражение с этим ключевым словом используется в SQL-запросе, чтобы расположить записи в определенном порядке.

10. Это PHP-выражение используется для включения кода из другого сценария.

12. Добавляемые файлы очень удобно применять для _____ использования данных несколькими сценариями.

По вертикали

2. Выражение с этим ключевым словом используется в SQL-запросе для предотвращения удаления более чем одной записи из таблицы базы данных.

3. Это выражение PHP используется для инициализации константы.

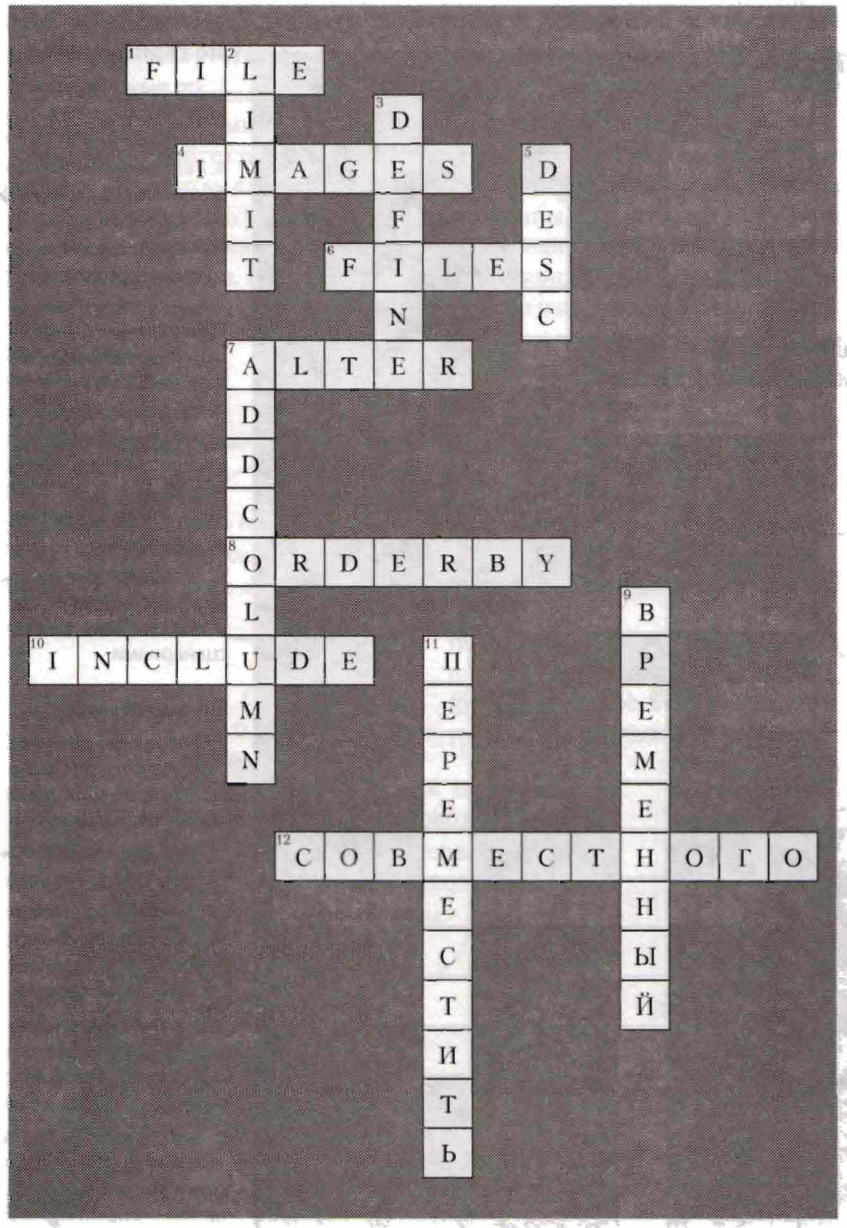
5. Это ключевое слово используется как часть SQL-выражения, обеспечивающего расположение записей в результате запроса в нисходящем порядке.

7. Выражение с этими ключевыми словами используется в SQL-запросе для добавления новой колонки при изменении структуры таблицы базы данных.

9. Когда файл загружается на сервер в результате обработки формы, он помещается во _____ каталог на этом сервере.

11. Весьма полезные действия, которые рекомендуется производить над только что загруженными на сервер файлами.

Кроссворд PHP и MySQL. Решение



Ваш инструментарий PHP и MySQL

Не стесняйтесь и возьмите в руки виртуальный ящик для инструментов. Вы не только стали любимым человеком для виртуальных гитаристов, но также накопили еще немного нового опыта о PHP и MySQL: изменение структуры таблиц, загрузка файлов на сервер, упорядочение данных...

ALTER TABLE *таблица*
ADD COLUMN *колонка тип*

Используйте этот запрос для добавления новой колонки в существующую таблицу базы данных. Колонка будет добавлена в конец перечня существующих колонок таблицы. Для всех записей, уже существующих на момент создания колонки, ее значения будут пустыми.

Этот каталог предоставляет удобное место для хранения изображений, которые использует приложение, в том числе те, которые были загружены пользователем.

$_FILES

Встроенный суперглобальный массив PHP $_FILES содержит информацию о файлах, которые были загружены на сервер с использованием формы загрузки файлов. Вы можете использовать этот массив для определения имени файла, места его временного хранения, размера, типа и других атрибутов этого файла.

DELETE FROM *таблица*
WHERE *колонка = значение*
LIMIT *число*

Используйте этот SQL-запрос для удаления записи из таблицы базы данных. Более одного условия может (и часто должно!) быть определено для того, чтобы повысить точность удаления, не говоря уже об ограничении количества удаляемых записей одной записью.

include, include_once, require, require_once

Эти PHP-выражения позволяют использовать код нескольким сценариям приложения совместно, что дает возможность избегать дублирования кода и облегчает техническую поддержку приложения.

ORDER BY *колонка*

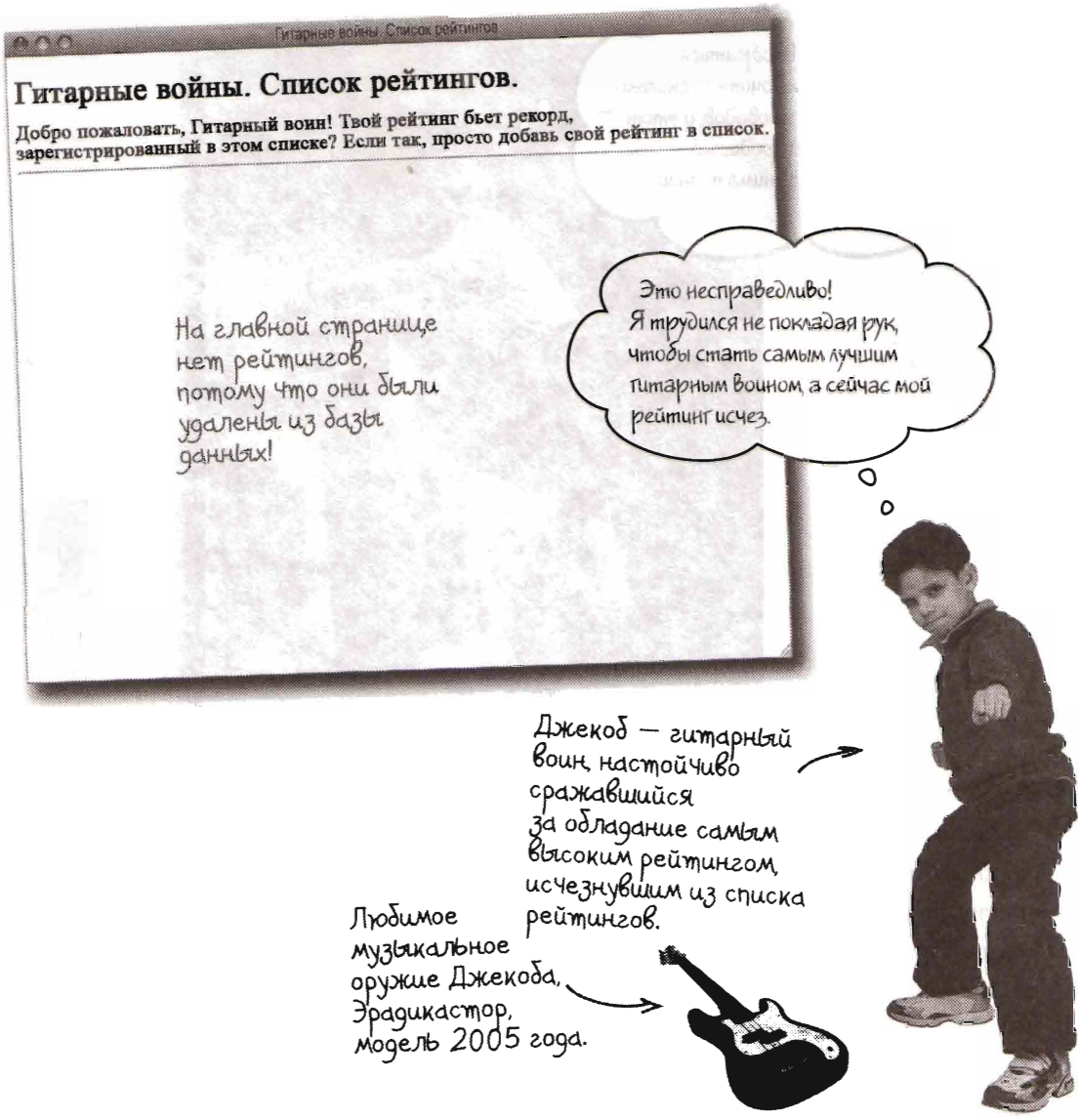
Это SQL-выражение упорядочивает записи в результате запроса в соответствии со значениями указанной колонки. Используйте ключевые слова ASC (от ASCending — «восходящий») или DESC (от DESCending — «нисходящий») в конце этого выражения, чтобы располагать записи в восходящем или нисходящем порядке соответственно. ASC является значением по умолчанию, следовательно, его указывать необязательно.

Считайте, что все они намерены воспользоваться вашими слабостями

Ваши родители были правы: никогда не разговаривайте с незнакомыми людьми. Или, по крайней мере, не доверяйте им. Ну и, конечно же, не давайте им ключи доступа к данным вашего приложения, полагая, что они будут делать то, что положено. Мы живем в жестоком мире, и вы не можете рассчитывать на то, что все окружающие заслуживают доверия. По сути, будучи разработчиком веб-приложения, вы должны быть частично циником, частично конспиратором-теоретиком. Да, люди в своем большинстве плохие, и все они определенно намерены воспользоваться вашими слабостями! Ладно, возможно, такой взгляд граничит с крайностью, но очень важно серьезно относиться к вопросам защиты своих приложений и разрабатывать их так, чтобы они могли противостоять любому, кто попытается нанести им вред.

День, когда умерла музыка

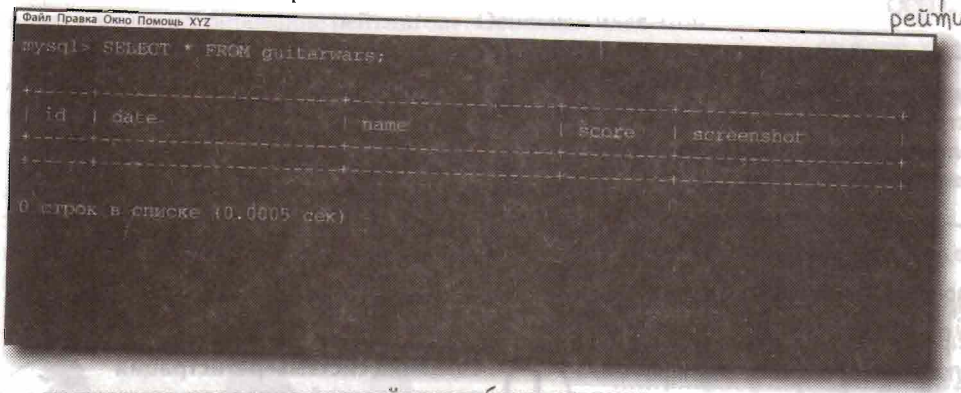
Эх, время, когда наш молодой виртуальный рок-вундеркинд купался в лучах славы, оказалось слишком коротким, потому что наивысший рейтинг «Гитарных войн», принадлежащий Джекобу, как-то вдруг исчез вместе со всеми другими рейтингами. Как будто какие-то дьявольские силы вмешались, чтобы разрушить приложение «Гитарные войны» и не дать возможности гитарным воинам соревноваться в Сети. Несчастные виртуальные гитаристы — это несчастные пользователи, и все это отражается на вас — несчастном разработчике приложения!



Куда делись рейтинги?

Мы знаем, что главная страница приложения «Гитарные войны» пуста, но означает ли это, что и база данных также пуста? Запрос SELECT может помочь с ответом на этот вопрос.

Запрос SELECT показывает, что таблица guitarwars совершенно пуста: все рейтинги исчезли!



Как-то получилось, что все записи рейтингов были удалены из таблицы guitarwars базы данных. Могло ли произойти, что кто-то использовал наш сценарий «Удаление рейтинга», чтобы совершить такое злодеяние? Нам необходимо защитить рейтинги!

Время поработать

Просмотрите предложенные ниже варианты, выберите те из них, которые вы могли бы использовать для защиты рейтингов приложения «Гитарные войны» от злобных виртуальных гитарных ненавистников, и напишите, почему вы выбрали именно эти варианты.

Защита паролем страницы «Администрирование рейтингов»: только человек, который знает пароль (вы!), сможет удалять рейтинги.

.....

Создание системы регистрации пользователей с предоставлением только отдельным пользователям (вам!) привилегий администратора.

.....

Проверка IP-адреса компьютера, пытающегося получить доступ к странице «Администрирование рейтингов», и предоставление такого доступа только определенным компьютерам (вашему!).

.....

Ликвидация самой возможности удаления рейтингов.

Решение задачи

Просмотрите предложенные ниже варианты, выберите те из них, которые вы могли бы использовать для защиты рейтингов приложения «Гитарные войны» от злобных виртуальных гитарных ненавистников, и напишите, почему вы выбрали именно эти варианты.

Защита паролем страницы «Администрирование рейтингов»: только человек, который знает пароль (вы!), сможет удалять рейтинги.

Защита паролем страницы «Администрирование рейтингов» — это хорошее решение, потому что оно не слишком сложно и быстро защищает сайт.

Все варианты так или иначе решают проблему, только одни более жизнеспособны, чем другие.

Создание системы регистрации пользователей с предоставлением только отдельным пользователям (вам!) привилегий администратора.

Это отличное решение, но его реализация требует тщательного планирования и больших усилий в кодировании... «Гитарным войнам» требуется защита сейчас!

Проверка IP-адреса компьютера, пытающегося получить доступ к странице «Администрирование рейтингов», и предоставление такого доступа только определенным компьютерам (вашему!).

Проверка IP-адреса вашего компьютера дает результат, но делает приложение зависимым от этого адреса, а он может очень легко измениться (в том числе по совершенно не зависящей от вас причине).

Ликвидация самой возможности удаления рейтингов.

Ликвидация самой возможности удалять рейтинги решает одну проблему, но, как вы помните, мы добавили возможность удалять рейтинги в предыдущей главе, чтобы улучшить техническую поддержку сайта.

Защита от нашествия банд злоумышленников

Простой и прямой способ быстрой защиты приложения «Гитарные войны» заключается в использовании HTTP-аутентификации с проверкой имени пользователя и его пароля при попытке открытия страницы «Администрирование рейтингов». Основная идея этой технологии заключается в том, что для получения доступа к критическим функциям приложения, таким, например, как удаление рейтингов, администратор должен ввести определенные секретные данные.

HTTP-аутентификация предоставляет простой способ защиты страницы с использованием PHP.

Если страница защищена с использованием HTTP-аутентификации, при попытке ее открытия появляется окно диалога, запрашивающее имя пользователя и его пароль. И только после ввода правильных данных будет предоставлен доступ к защищаемой странице. В случае «Гитарных войн» вы можете предоставить такой доступ ограниченному количеству пользователей, потенциально — только себе!

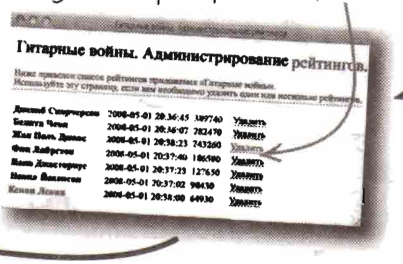
Теперь между пользователем и страницей «Администрирование рейтингов» находится окно HTTP-аутентификации.

После того как страница «Администрирование рейтингов» стала защищенной, защищенными стали также и рейтинги в базе данных.

guitarwars

id	date	name	score	screenshot
14	2008-05-01 20:36:07	Белита Чеви	282470	belitasscore.gif
15	2008-05-01 20:36:45	Джекоб Скорчерсон	389740	jascobsscore.gif
16	2008-05-01 20:37:02	Невил Йоханссон	98430	nevilsscore.gif
17	2008-05-01 20:37:23	Пако Джасторикус	127650	pacosscore.gif
18	2008-05-01 20:37:40	Фиц Пайрстон	186580	phizsscore.gif
19	2008-05-01 20:38:00	Кенни Левиц	64930	kennysscore.gif
20	2008-05-01 20:38:23	Жан Поль Джонс	243360	jeanpaulsscore.gif

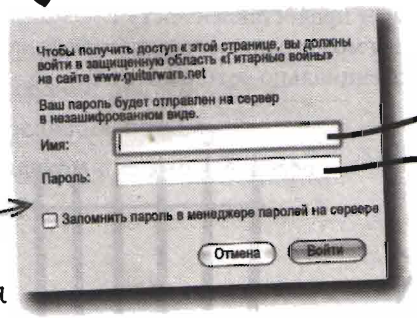
Переход по гиперссылке «Удалить» на странице «Администрирование рейтингов» теперь доступен только администратору «Гитарных войн».



Защита страницы «Администрирование рейтингов» приложения «Гитарные войны»

HTTP-аутентификация работает следующим образом. Когда пользователь пытается получить доступ к странице, защищенной методом HTTP-аутентификации, такой как наша страница «Администрирование рейтингов», появляется окно диалога, запрашивающее имя пользователя и его пароль.

Браузер использует такое окно для запроса имени пользователя и пароля, прежде чем предоставить доступ к защищенной странице.



Для упрощения пароль передается в незашифрованном виде.

Эта суперглобальная переменная содержит имя пользователя, введенное в окне диалога аутентификации.

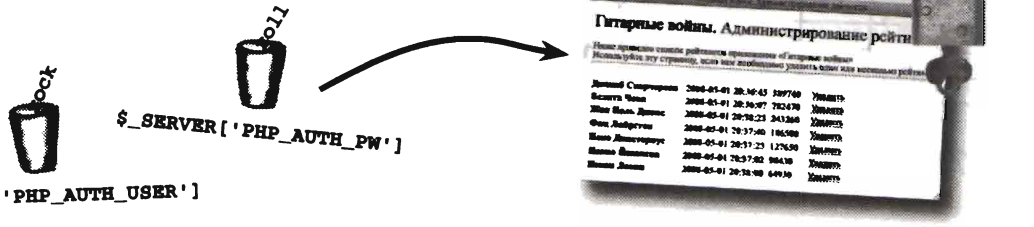
```
$_SERVER['PHP_AUTH_USER']
```

```
$_SERVER['PHP_AUTH_PW']
```

Эта суперглобальная переменная содержит пароль, введенный в окне диалога аутентификации.

PHP вступает в игру, когда возникает необходимость в доступе к имени пользователя и его паролю, которые он ввел в окне диалога аутентификации. Они сохранены в суперглобальном массиве `$_SERVER`, аналогичном другим суперглобальным переменным, которые мы уже использовали (`$_POST`, `$_FILES` и т. п.). PHP-сценарий может проанализировать имя пользователя и его пароль, введенные в окне диалога аутентификации, и на основании результатов такого анализа принять решение либо о предоставлении этому пользователю доступа к защищенной странице, либо об отклонении запроса. Предположим, доступ к странице «Администрирование рейтингов» разрешен пользователю с именем «рок» и паролем «ролл». Ниже показано, как будет открыта страница «Администрирование рейтингов».

Страница «Администрирование рейтингов» доступна только в том случае, если пользователь ввел правильное имя и пароль.



Не бывает глупых вопросов

В: • Полностью ли защищает HTTP-аутентификация?

О: И да, и нет. Все зависит от того, чего вы хотите достичь в защите своей информации. 100%-й защиты не существует в принципе. Поэтому все, о чем мы можем говорить, — это степень защиты. Для защиты рейтингов приложения «Гитарные войны» HTTP предоставляет приемлемый уровень. Вы можете несколько повысить уровень этой защиты, добавив шифрование пароля. Тем не менее такой уровень защиты, скорее всего, будет недостаточным для приложений, имеющих дело с более чувствительными данными — такими, например, как финансовые.

В: • Что произойдет, если имя пользователя и/или его пароль будут введены неправильно?

О: Браузер эмитирует небольшой электрический разряд через мышь. Нет, никакого болезненного воздействия не будет. Обычно выводится сообщение о том, что пользователь пытается получить доступ к защищенной странице, имеющей дело с вопросами, которые его не касаются. Насколько жесткой будет информация, содержащаяся в этом сообщении, зависит исключительно от вас.

В: • Требуется ли при HTTP-аутентификации вводить и имя пользователя, и его пароль? Что, если я введу только пароль?

О: Вам не требуется вводить и имя пользователя, и его пароль. Если вас интересует только пароль, проверяйте только суперглобальную переменную `$_SERVER['PHP_AUTH_PW']`. Больше о том, как проверяется эта переменная, чуть дальше…

В: • Как на практике защищается страница с помощью HTTP-аутентификации? Необходимо вызвать PHP-функцию?

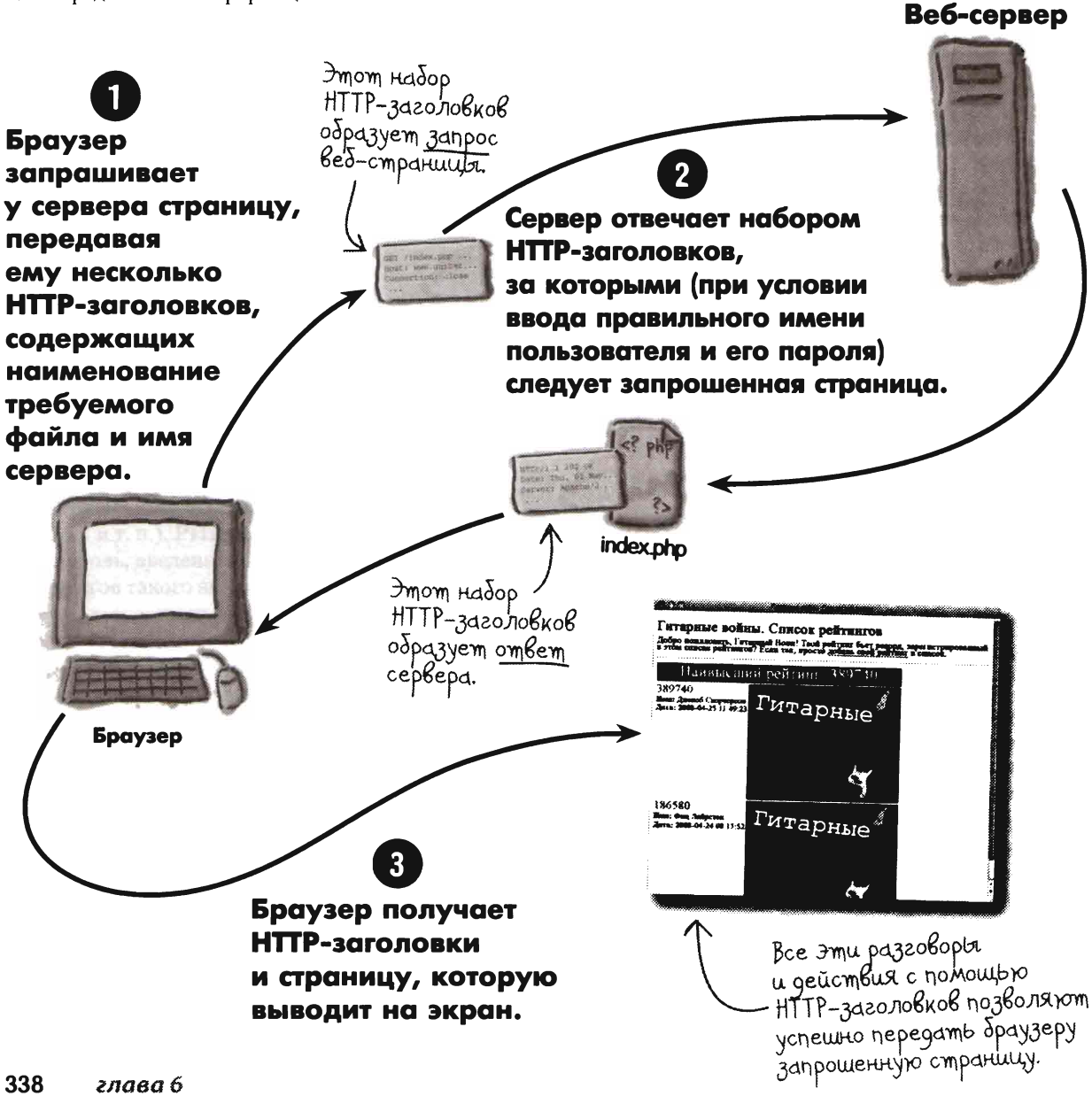
О: Да, нужно. HTTP-аутентификация включает обмен информацией между браузером и веб-сервером с использованием HTTP-заголовков. Вы можете рассматривать HTTP-заголовки как краткие диалоги браузера с сервером. Браузер и сервер достаточно часто используют HTTP-заголовки для обмена информацией вне контекста PHP, но PHP позволяет вам отправить HTTP-заголовок для осуществления HTTP-аутентификации. Нам придется углубиться немного больше в понимание HTTP-заголовков и их роли в HTTP-аутентификации применительно к PHP.

Когда конкретно должна состояться HTTP-аутентификация для доступа к странице «Администрирование рейтингов»?

HTTP-аутентификации требуются HTTP-заголовки

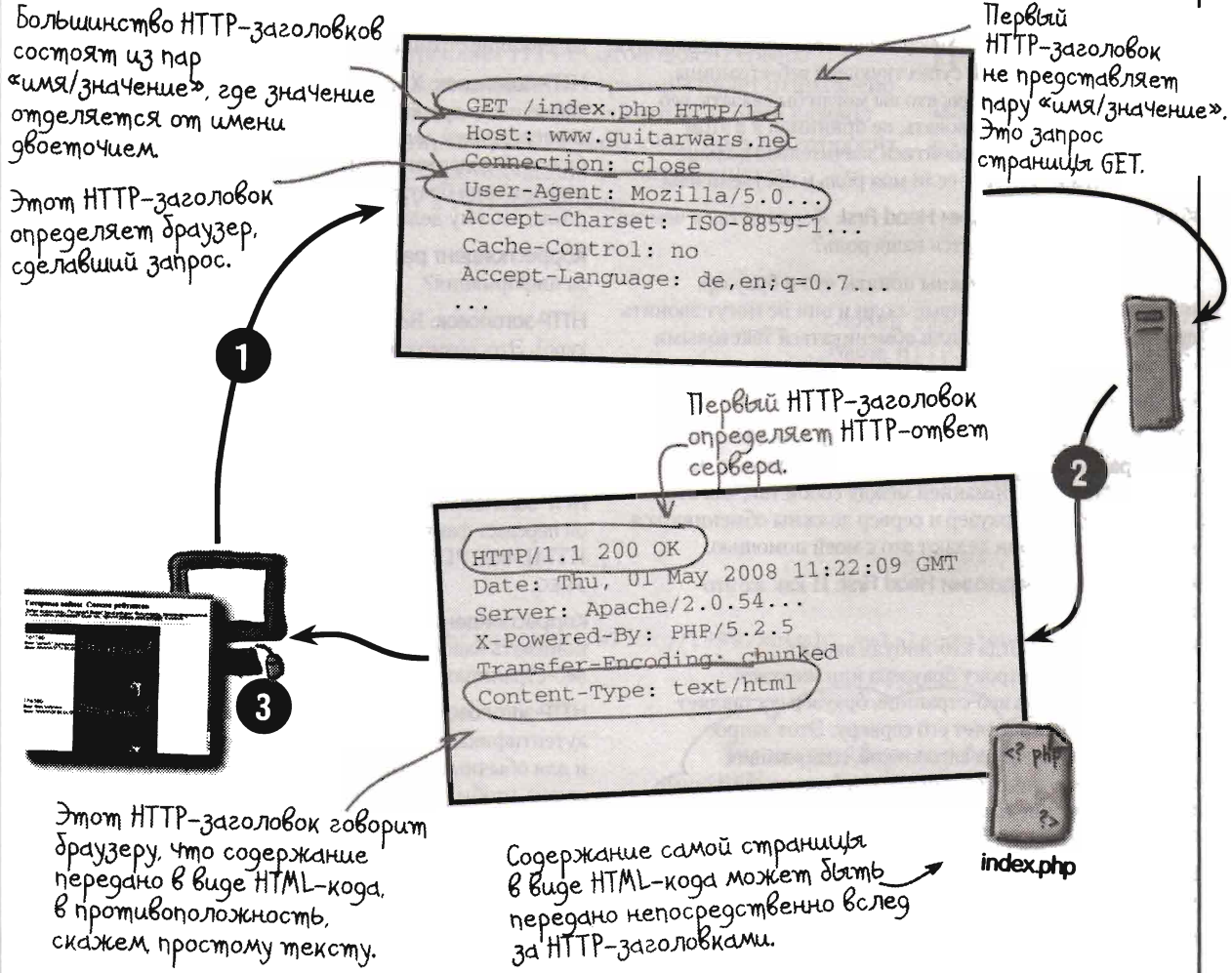
Основная идея, лежащая в основе HTTP-аутентификации, заключается в том, что в ответ на запрос браузера о передаче защищенной страницы сервер вначале запрашивает имя пользователя и его пароль и только после того, как пользователь введет правильное имя и пароль, отправляет страницу. Этот диалог между браузером и сервером происходит посредством HTTP-заголовков, которые представляют собой небольшие текстовые сообщения, содержащие особые инструкции о запрашиваемой или передаваемой информации.

Все веб-страницы передаются с помощью HTTP-заголовков.



Анатомия HTTP-заголовка

HTTP-заголовки тщательно следят за тем, какая информация и как проходит в прямом и обратном направлении между браузером и веб-сервером. Индивидуальный HTTP-заголовок чаще всего состоит из пары «имя/значение», которая определяет конкретные сведения — такие, например, как формат содержания веб-страницы (HTML). Определенная группа HTTP-заголовков посылается на сервер как часть запроса веб-страницы, а затем другая группа возвращается сервером как часть ответа на этот запрос. Давайте познакомимся поближе с этими группами HTTP-заголовков, чтобы понять, какая передается информация, когда клиент и сервер ведут диалог друг с другом.



Понимание действия HTTP-заголовков в приложении «Гитарные войны» имеет для нас значение потому, что через них реализуется механизм прерывания сервером передачи страницы и формирование требования о том, чтобы пользователь ввел свое имя и пароль прежде, чем страница будет передана. Иначе говоря, вы должны тщательно разобраться с этими HTTP-заголовками для того, чтобы защитить страницу с помощью HTTP-аутентификации.

HTTP-заголовок: пристальный взгляд

Интервью этой недели: По какому поводу вся эта суета?

Корреспондент редакции Head First: Похоже, вы привлекаете большое внимание, когда дело касается аутентификации веб-страниц. Это действительно оправданно или вы просто ищете повод для своей пятнадцатиминутной славы?

HTTP-заголовок: О, конечно, это оправданно. Вы наверняка считаете само собой разумеющимся, что я играю решающую роль в передаче каждой существующей веб-страницы. Поэтому я не сомневаюсь, что вы могли бы сказать: веб вряд ли смог бы существовать, не принимай я в этом участия. И без меня не обойтись значительно дольше пятнадцати минут, даже если моя роль и преувеличена.

Корреспондент редакции Head First: Хорошо, так в чем же на самом деле заключается ваша роль?

HTTP-заголовок: Вы должны понять, что и браузер, и веб-сервер — это не живые люди и они не могут звонить друг другу по телефону или обмениваться текстовыми сообщениями.

Корреспондент редакции Head First: Боже мой!

HTTP-заголовок: Да, я знаю, это звучит немного обескураживающе, но машины действительно не могут обмениваться информацией между собой так, как это делают люди. Но браузер и сервер должны обмениваться информацией, и они делают это с моей помощью.

Корреспондент редакции Head First: И как же это происходит?

HTTP-заголовок: Когда кто-нибудь вводит URL в адресную строку браузера или щелкает по гиперссылке на веб-странице, браузер составляет запрос `GET` и отправляет его серверу. Этот запрос состоит из нескольких заголовков, содержащих информацию о нем. Заголовки содержат такие данные, как наименование и адрес запрашиваемой страницы, тип браузера, с которого был отправлен запрос, и т. п.

Корреспондент редакции Head First: Я до сих пор не могу понять, почему это все так важно.

HTTP-заголовок: Ладно. Как вы думаете, является ли важной для официанта кафе информация о том, что вы хотите большую порцию ванильного эспрессо с молоком?

Корреспондент редакции Head First: Конечно, он же должен знать, чего я хочу.

HTTP-заголовок: То же самое и здесь. Браузер говорит серверу, чего он хочет, составляя запрос и отправляя его в виде заголовков.

Корреспондент редакции Head First: Интересно. Но я слышал, что и сервер также может отправлять заголовки. Я думал, что сервер отправляет только содержание страниц.

HTTP-заголовок: Хороший вопрос. Дело в том, что я важен также и для другой стороны этого диалога, потому что сервер должен делать больше, чем просто перегнать браузеру содержимое страницы. Без дополнительной информации браузер просто не будет иметь представления о том, что ему делать со всем тем, что он получил.

Корреспондент редакции Head First: Что это должна быть за информация?

HTTP-заголовок: Во-первых, формат содержания (Content type). Это, возможно, самая важная информация, но сервер также отправляет и другие данные, такие как размер страницы, дата и время передачи и т. д.

Корреспондент редакции Head First: А когда же отправляется сама страница?

HTTP-заголовок: Сразу после того как сервер отправит меня, он передает фактическое содержание страницы, будь то HTML-код, PDF-данные или изображения в формате `GIF`, `JPEG`...

Корреспондент редакции Head First: Хорошо, я начинаю понимать вашу роль, когда речь идет об обычных веб-страницах. А как обстоят дела с аутентификацией?

HTTP-заголовок: Для страниц, требующих аутентификации, мои функции остаются такими же, как и для обычных страниц, за исключением одного: я слежу за тем, чтобы дать знать браузеру, что запрашиваемая им страница защищена и требует аутентификации. В этом случае браузер запрашивает у пользователя данные для аутентификации.

Корреспондент редакции Head First: Вы имеете в виду имя и пароль?

HTTP-заголовок: Совершенно верно. А дальше уже PHP-код на сервере решает, правильные ли имя и пароль переданы с браузера, и, если все в порядке, отправляет запрошенную страницу.

Корреспондент редакции Head First: Великолепно! Благодарю вас за разъяснение.

HTTP-заголовок: Никаких проблем. Это часть моей работы.

Работа с HTTP-заголовками в PHP

Используя PHP, вы можете очень легко работать с заголовками, присланными сервером, что дает возможность выполнять задачи, управляемые заголовками, к которым относится и HTTP-аутентификация. Встроенная функция `header()` может быть использована в PHP-сценарии для передачи заголовка с сервера браузеру.

```
header('Content-Type: text/html');
```

Функция header() позволяет вам создавать и отправлять HTTP-заголовки из PHP-сценария.

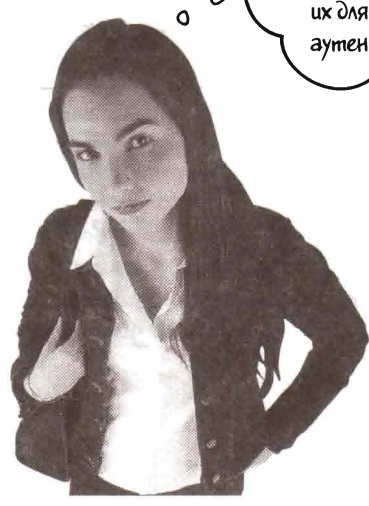
Функция `header()` немедленно отправляет HTTP-заголовок с сервера и должна вызываться до того, как само содержание страницы будет отправлено браузеру. Это требование является очень строгим. Если хотя бы один символ или даже пробел будет отослан до отправки HTTP-заголовка, браузер отбросит его и выведет сообщение об ошибке. Поэтому вызов функции должен предшествовать любому HTML-коду в PHP-сценарии:

Даже случайно вставленный символ пробела перед тегом <?php вызовет ошибку в этом примере сценария.

Пробелы внутри тегов <?php ?> не приведут ни к каким проблемам, так как они не посылаются браузеру непосредственно.

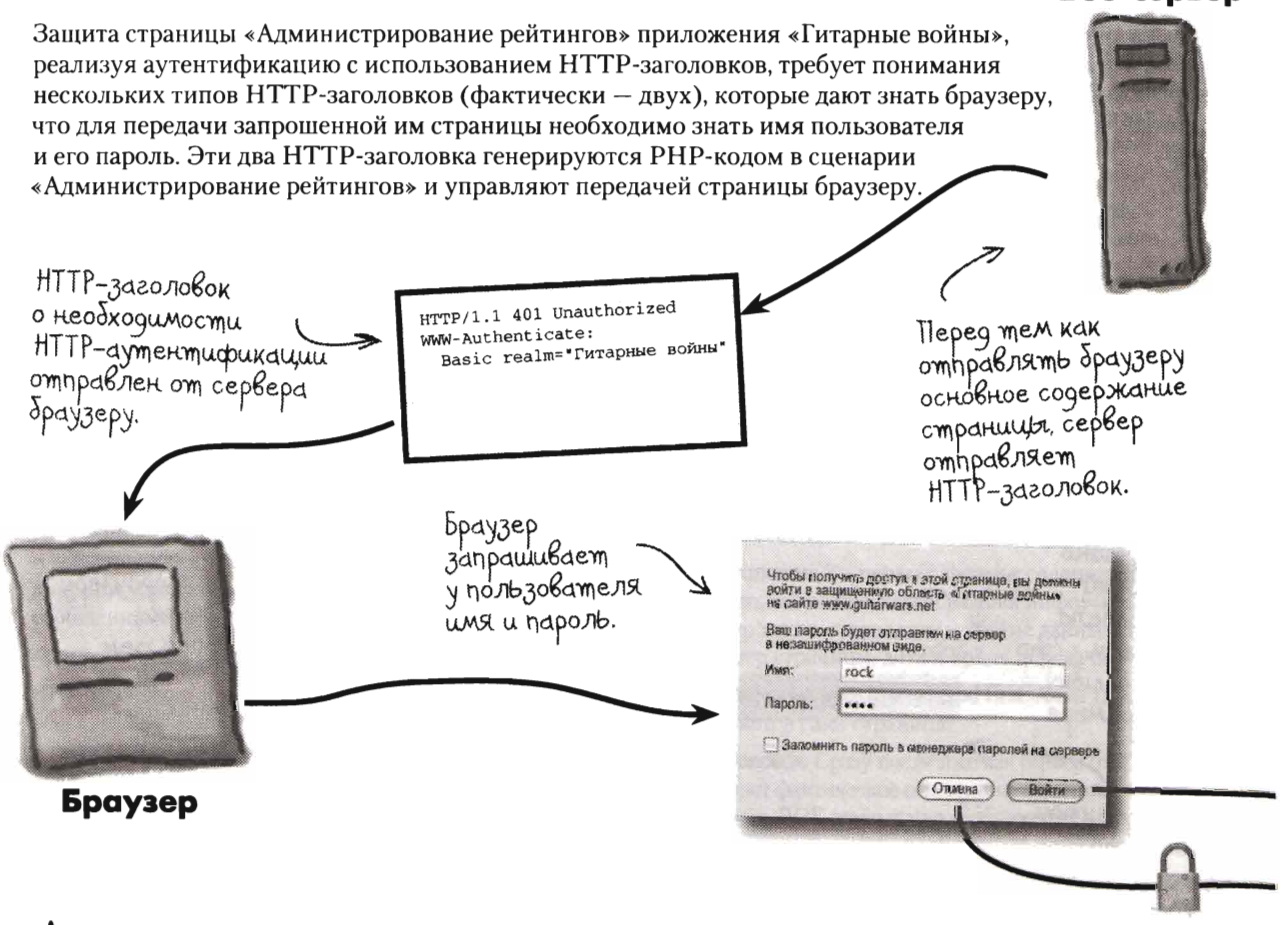
```
<?php
  header('Content-Type: text/html');
  ...
?>

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en" >

</html>
```

Сервер отправляет браузеру этот HTTP-заголовок для обработки перед тем, как отправлять любое содержание страницы.



Аутентификация с использованием HTTP-заголовков

Защита страницы «Администрирование рейтингов» приложения «Гитарные войны», реализуя аутентификацию с использованием HTTP-заголовков, требует понимания нескольких типов HTTP-заголовков (фактически — двух), которые дают знать браузеру, что для передачи запрошенной им страницы необходимо знать имя пользователя и его пароль. Эти два HTTP-заголовка генерируются PHP-кодом в сценарии «Администрирование рейтингов» и управляют передачей страницы браузеру.



Для запроса данных, необходимых для аутентификации, требуются два специальных HTTP-заголовка.

Чтобы инициировать аутентификацию, требуются два HTTP-заголовка, которые выполняют две специфические задачи:

```
HTTP/1.1 401 Unauthorized
```

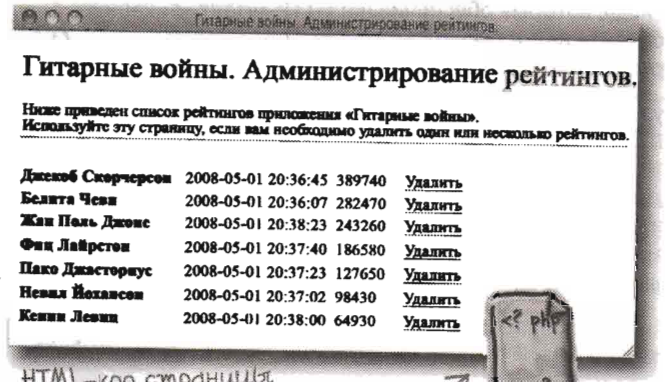
Этот HTTP-заголовок дает браузеру понять, что данному пользователю не предоставляется доступ к защищенной странице.

```
WWW-Authenticate: Basic realm="Гитарные войны"
```

Этот HTTP-заголовок просит браузер попытаться аутентифицировать пользователя, предложив ему ввести имя и пароль.

Фраза Basic realm (Защищенная область) всего лишь служит идентификатором этой конкретной аутентификации и появляется в окне диалога аутентификации.

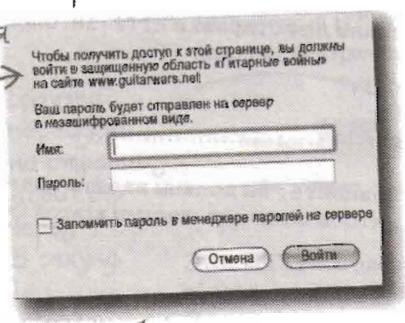
После обработки HTTP-заголовков аутентификации браузер ожидает результатов диалога с пользователем, который проходит с использованием окна аутентификации. При этом действия браузера могут разительно различаться в зависимости от действий пользователя...



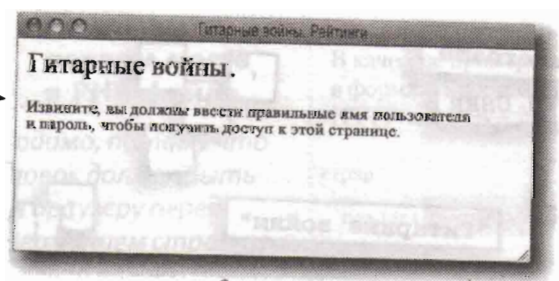
admin.php

HTML-код страницы «Администрирование рейтингов» будет передан только после того, как будут введены правильные имя и пароль.

Это защищенная область.



Если введены неправильные имя пользователя и пароль, окно диалога аутентификации будет выведено вновь.



Приложение также имеет возможность остановить выполнение сценария и вывести собственное сообщение о причинах отклонения запроса о доступе к странице, если пользователь прекратил аутентификацию.

Если пользователь введет правильное имя и пароль и затем нажмет кнопку «Войти», сервер отправит браузеру содержание страницы «Администрирование рейтингов», и пользователь получит возможность удалять рейтинги так же, как и в предыдущей незащищенной версии.

Если пользователь введет неправильное имя и/или пароль и затем нажмет кнопку «Войти», сервер сообщит браузеру, что необходимо запросить эти данные повторно. Браузер будет повторять эти действия в течение всего времени, пока пользователь будет вводить неправильное имя и/или пароль. Иначе говоря, если пользователь не знает правильного имени и пароля, все, что ему остается, — это только нажать кнопку «Отмена».

Если пользователь нажмет кнопку «Отмена» для того, чтобы прекратить аутентификацию, сервер отправит браузеру страницу с сообщением об отклонении запроса на доступ, и ничего больше. Страница «Администрирование рейтингов» послана не будет. Сообщение об отклонении запроса на доступ контролируется PHP-кодом в сценарии `admin.php`, который тесно связан с HTTP-заголовками. Этот код выводит сообщение и немедленно прекращает выполнение сценария:

```
exit('<h2>Гитарные войны</h2> Извините,
вы должны ввести правильные ' .
     'имя пользователя и пароль, чтобы
получить доступ к этой странице.');
```

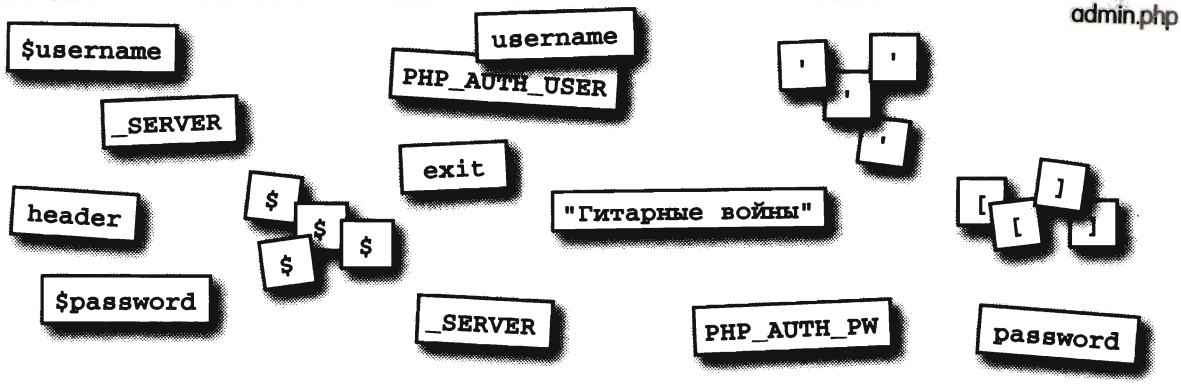
Магниты PHP

В сценарии «Администрирование рейтингов» приложения «Гитарные войны» отсутствуют несколько важных фрагментов кода, которые обеспечивают HTTP-аутентификацию. Используйте магниты, чтобы добавить недостающий код и с помощью HTTP-заголовков защитить страницу «Администрирование рейтингов».

Совет: отдельные магниты могут быть использованы только один раз.

```
<?php
  // Имя пользователя и его пароль для аутентификации
  .....= 'rock';

  .....= 'roll';

  if (!isset(.....) ||

     !isset(.....) ||

     ($_SERVER['PHP_AUTH_USER'] !=.....) ||

     ($_SERVER['PHP_AUTH_PW'] !=.....)) {

    // Имя пользователя/пароль не действительны для отправки HTTP-заголовков,
    // подтверждающих аутентификацию
    .....('HTTP/1.1 401 Unauthorized');

    ..... ('WWW-Authenticate: Basic realm=.....');

    .....('<h2> Гитарные войны h2> Извините, вы должны ввести правильные ' .
          'имя пользователя и пароль, чтобы получить доступ к этой странице.');
  }
?>

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ru" lang="ru">
  ...
</html>
```

admin.php



Конечно можно... HTTP-заголовки используются не только для безопасности.

Хотя именно процесс аутентификации демонстрирует непосредственное использование HTTP-заголовков, они достаточно гибки и могут быть использованы для решения многих других интересных задач. Достаточно просто вызвать функцию `header()` с соответствующей парой «имя/значение», как, например, показано ниже:

```
<?php
  header('Location: http://www.guitarwars.net/about.php');
?>
```

Этот HTTP-заголовок называется **location header** (HTTP-заголовок места расположения) и переадресовывает текущую страницу на страницу с именем `about.php` на том же сайте «Гитарные войны». А ниже показан аналогичный HTTP-заголовок, который переадресовывает текущую страницу на страницу с именем `about.php` через 5 секунд:

```
<?php
  header('Refresh: 5; url=http://www.guitarwars.net/about.php');
  echo 'Через 5 секунд вы перейдете на страницу «О сайте»';
?>
```

Этот HTTP-заголовок называется **refresh header** (HTTP-заголовок обновления), так как он обновляет страницу через указанное время. Вы часто сможете видеть, что такие HTTP-заголовки ссылаются на текущую страницу, то есть страница обновляет себя сама.

Один из последних HTTP-заголовков называется **content type header** (HTTP-заголовок формата содержания), потому что он контролирует тип формата содержания, передаваемого сервером. В качестве примера вы можете объявить, что содержание выводится в формате простого текста вместо HTML, используя следующий HTTP-заголовок при вызове функции `header()`:

```
<?php
  header('Content-Type: text/plain');
  echo 'Этот <strong>текст</strong> не будет выделен ' . ' жирным шрифтом';
?>
```

Содержание передано браузеру в виде простого текста.

В этом примере текст, выводимый в браузере, будет показан как есть, без HTML-форматирования. Иначе говоря, сервер говорит браузеру не интерпретировать текст как HTML-код, поэтому все содержание, включая HTML-теги, будет воспроизведено буквально, как простой текст.

Запомни!

HTTP-заголовки должны быть на самом первом месте в PHP-файле.

Это необходимо, потому что HTTP-заголовок должен быть отправлен браузеру перед любым содержанием страницы. Исключительно важно не допускать в PHP-сценарии даже единственного пробела вне PHP-кода перед вызовом функции header().

Магниты PHP. Решение

В сценарии «Администрирование рейтингов» приложения «Гитарные войны» отсутствуют несколько важных фрагментов кода, которые обеспечивают HTTP-аутентификацию. Используйте магниты, чтобы добавить недостающий код и с помощью HTTP-заголовков защитить страницу «Администрирование рейтингов».

Совет: отдельные магниты могут быть использованы только один раз.

Суперглобальная переменная $_SERVER предоставляет доступ к имени пользователя и его паролю, введенным в окно диалога аутентификации.

```
<?php
  // Имя пользователя и его пароль для аутентификации
  $username = 'rock';
  $password = 'roll';

  if (!isset($_SERVER['PHP_AUTH_USER']) ||
      !isset($_SERVER['PHP_AUTH_PW']) ||
      ($_SERVER['PHP_AUTH_USER'] != $username) ||
      ($_SERVER['PHP_AUTH_PW'] != $password)) {
    // Имя пользователя/пароль не действительны для отправки HTTP-заголовков,
    // подтверждающих аутентификацию
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Guitar Wars"');
    exit('<h2>Гитарные войны</h2>Извините, вы должны ввести правильные ' .
         'имя пользователя и пароль, чтобы получить доступ к этой странице.');
  }
?>

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ru" lang="ru">
  ...
</html>
```

Имя пользователя и его пароль сохранены в переменных в самом начале сценария.

Значения имени и пароля, введенные пользователем, сравниваются с заданными значениями.

В результате этих двух вызовов функции header() два HTTP-заголовка, переданные функции в качестве аргументов, будут отправлены браузеру.

admin.php

До тех пор пока не будут отправлены и обработаны HTTP-заголовки, никакого HTML-кода на браузер отправлено не будет.

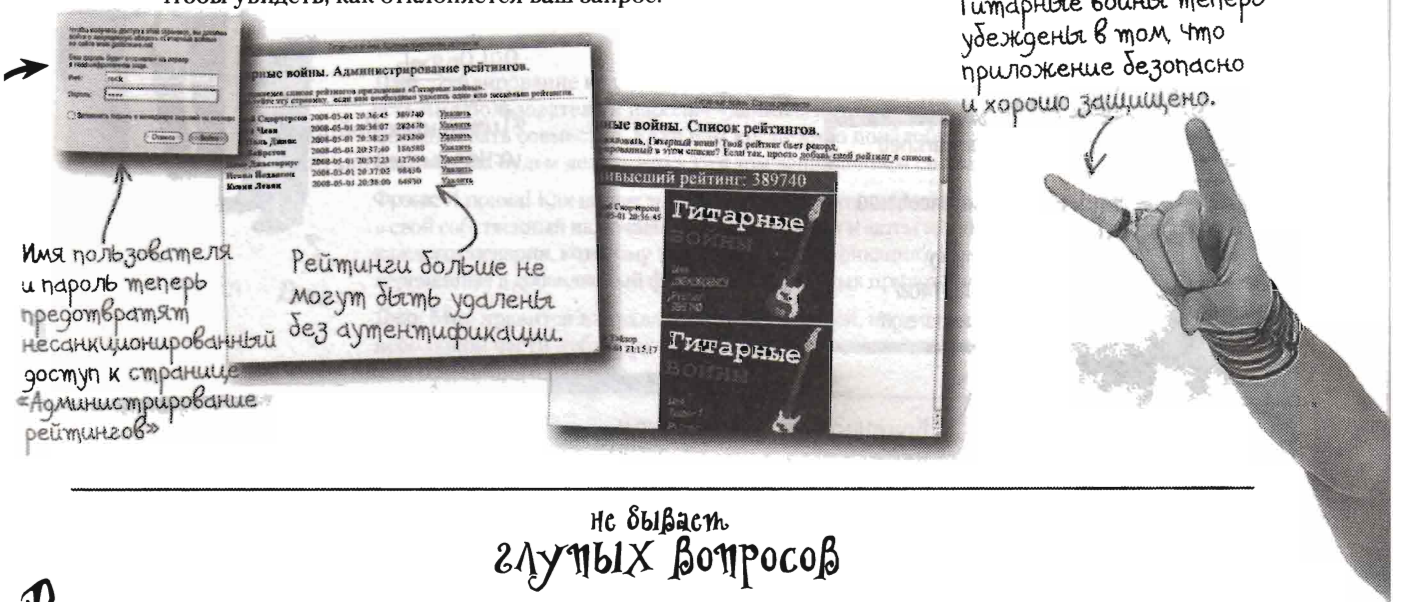
Функция exit() выводит сообщение об отклонении запроса на открытие страницы и предотвращает вывод любой другой информации в случае, если аутентификация закончится неудачно.

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate:
  Basic realm="Гитарные войны"
```

Тест-драйв

Добавьте HTTP-аутентификацию в сценарий «Администрирование рейтингов».

Внесите в сценарий `admin.php` изменения, обеспечивающие только вам доступ к этой странице. Загрузите сценарий на ваш веб-сервер и откройте страницу в браузере. Вначале попробуйте ввести неправильное имя и/или пароль, чтобы увидеть, как отклоняется ваш запрос.



не бывает глупых вопросов

В: Когда вызывается функция exit() в сценарии «Гитарные войны»?

О: Несмотря на то что функция `exit()` находится в PHP-коде сразу же за функциями `header()`, она вызывается только в том случае, если пользователь отменяет аутентификацию, нажимая кнопку «Отмена». В случае же неудавшейся аутентификации сервер прекращает интерпретацию кода после вызовов функций `header()` и отправления HTTP-заголовков и ожидает повторного ввода имени пользователя и его пароля. Только в том случае, если пользователь нажмет кнопку «Отмена», вызывается функция `exit()`, сервер отправляет содержание сообщения, переданного этой функции в качестве аргумента, и ничего больше. Если аутентификация проходит успешно, функция `exit()` не вызывается, потому что условное выражение управляющей конструкции `if` принимает значение `false`, и код, заключенный в ее фигурных скобках, выполняться не будет. Это условное выражение принимает значение `true` только в случаях, когда пользователь либо не введет свое имя и/или пароль, либо введет их неправильно.

В: Имеет ли какое-нибудь практическое значение выражение basic realm (защищенная область) в системе HTTP-аутентификации?

О: Да. Оно определяет безопасную зону, защищенную определенным именем и паролем. После того как имя пользователя и его пароль были успешно введены для данной зоны, браузер запоминает эти значения и больше не выводит окно диалога аутентификации для всех последующих случаев получения от сервера HTTP-заголовков аутентификации, касающихся этой зоны. Иначе говоря, basic realm позволяет браузеру запомнить, что вы отвечаете требованиям безопасности для данного набора страниц. Просто укажите одно и то же значение basic realm HTTP-заголовков аутентификации для всех страниц этого набора.

Отлично! К счастью, я сохранила страницу «Удаление рейтингов» в закладках своего браузера, а затем, чуть-чуть подправив дату, составила этот огромный URL.

http://www.guitarwars.net/removescore.php?id=10&name=Jacob%20Scorcherson&date=2008-05-01%2020:36:45&score=389740&screenshot=jacobsscore.gif

Похоже, злодей, покушающийся на наши «Гитарные войны», нашел способ обойти защиту нашего приложения.

Гитарные войны. Список рейтингов

ные войны. Список рейтингов.

...ро пожаловать, Гитарный воин! Твой рейтинг бьет рекорд, ...регистрированный в этом списке? Если так, просто добавь свой рейтинг в список.

Ой! Опять из приложения «Гитарные войны» исчезли рейтинги.

Конечно, это не слишком просто, но с помощью URL страницы removescore.php можно обойти защиту страницы admin.php.

Так, может быть, приложение «Гитарные войны» не защищено?

Подумайте о быстротечной славе. Злоумышленнику не понадобилось много времени, чтобы нанести удар вновь, выкинув рейтинги из приложения, и уж в который раз повергнуть в уныние толпы соперничающих исполнителей. Похоже, защиты одной страницы «Администрирование рейтингов» недостаточно, так как сценарий «Удаление рейтинга» остается пока еще доступным напрямую...

Напишите, как, по вашему мнению, мы сможем защититься от этой атаки и предотвратить несанкционированное удаление рейтингов:

.....

Нам необходимо защитить сценарий «Удаление рейтинга», и, я уверен, мы опять сможем использовать для этого HTTP-аутентификацию.

Джо: В этом есть смысл. Я имею в виду то, что HTTP-аутентификация отлично защищает страницу «Администрирование рейтингов».

Фрэнк: Это точно. Поэтому все, что нам нужно сделать, — это поместить тот же самый код, использующий HTTP-заголовки аутентификации, в сценарий «Удаление рейтинга», и все будет в порядке, так?

Джил: Да, это определенно будет работать. Но меня беспокоит то, что нам приходится повторять весь этот код аутентификации в двух местах. Что произойдет, если позднее мы добавим еще одну страницу, которую тоже потребуется защитить? Повторять этот код опять?

Джо: Дублирование кода — это определенно проблема. Особенно потому, что у нас есть имя пользователя и пароль — данные, которые защищаемые сценарии должны использовать совместно. Если нам когда-либо понадобится изменить эти данные, мы должны будем делать это в каждом защищенном сценарии.

Фрэнк: Я понял! Как насчет того, чтобы поместить переменные `$username` и `$password` в свой собственный включаемый (include) файл и затем включить имя этого файла в код каждого сценария, которому требуется аутентификация? Мы даже можем включить эти переменные в добавляемый файл для переменных приложения `appvars.php`.

Джо: Мне нравится направление твоих мыслей, но это решение касается только небольшой части дублирующегося кода. Не забывай, что мы ведем речь о фрагменте кода приличного размера.

```
<?php
  // Имя пользователя и его пароль для аутентификации
  $username = 'rock';
  $password = 'roll';

  if (!isset($_SERVER['PHP_AUTH_USER']) ||
    !isset($_SERVER['PHP_AUTH_PW']) ||
    ($_SERVER['PHP_AUTH_USER'] != $username) ||
    ($_SERVER['PHP_AUTH_PW'] != $password)) {
    // Имя пользователя/пароль не действительны для отправки HTTP-заголовков,
    // подтверждающих аутентификацию
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Гитарные войны"');
    exit('<h2> Гитарные войны h2> Извините, вы должны ввести правильные ' .
      'имя пользователя и пароль, чтобы получить доступ к этой странице.');
  }
?>
<html
```

admin.php

Джил: Вы оба правы. Поэтому, я думаю, нам необходимо создать новый добавляемый файл и поместить в него весь код аутентификации, а не только переменные `$username` и `$password`.

Фрэнк: Точно, и мы сможем просто включить этот файл в любой сценарий, который захотим защитить с помощью HTTP-аутентификации.

Джо: Все правильно! Мы только должны следить за тем, чтобы включать его в самом начале, так как он полагается на HTTP-заголовки в процессе аутентификации.

Создание сценария «Аутентификация»

У нас уже есть весь код, необходимый для создания нового сценария «Аутентификация». Дело заключается только в том, чтобы перенести его из сценария `admin.php` в новый файл сценария (`authorize.php`) и заменить прежний код выражением `require_once`.

Мы извлекаем этот код из сценария admin.php для того, чтобы поместить его в его собственный файл сценария authorize.php.

```
<?php
   // Имя пользователя и его пароль для аутентификации
   $username = 'rock';
   $password = 'roll';

   if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW']) ||
      ($_SERVER['PHP_AUTH_USER'] != $username) ||
      ($_SERVER['PHP_AUTH_PW'] != $password)) {
      // Имя пользователя/пароль не действительны для отправки HTTP-заголовков, подтверждающих аутентификацию
      header('HTTP/1.1 401 Unauthorized');
      header('WWW-Authenticate: Basic realm="Гитарные войны"');
      exit('<h2> Гитарные войны h2> Извините, вы должны ввести правильные ' .
           'имя пользователя и пароль, чтобы получить доступ к этой странице.');
   }
?>

<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
      <title>Гитарные войны. Администрирование рейтингов.</title>
      <link rel="stylesheet" type="text/css" href="style.css" />
   </head>
   <body>
      <h1>Гитарные войны. Администрирование рейтингов.</h1>

      <p>
         Ниже приведен список рейтингов приложения «Гитарные войны». Используйте эту страницу,
         если вам необходимо удалить один или несколько рейтингов.
      </p>
      <hr />

<?php
   require_once('appvars.php');
   require_once('connectvars.php');

   // Соединение с базой данных
   $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

   // Извлечение данных-рейтингов из базы MySQL
   $query = "SELECT * FROM guitarwars ORDER BY score DESC, date ASC";
   $data = mysqli_query($dbc, $query);

   // Извлечение данных из массива рейтингов в цикле.
   // Форматирование данных записей в виде кода HTML
   echo '<table>';
   while ($row = mysqli_fetch_array($data)) {
      // Вывод данных рейтинга
      echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
      echo '<td>' . $row['date'] . '</td>';
      echo '<td>' . $row['score'] . '</td>';
      echo '<td><a href="removescore.php?id=' . $row['id'] . '&amp;date=' . $row['date'] .
           '&amp;name=' . $row['name'] . '&amp;score=' . $row['score'] .
           '&amp;screenshot=' . $row['screenshot'] . '">Удалить</a>';
      echo '</td></tr>';
   }
   echo '</table>';

   mysqli_close($dbc);
?>
   </body>
</html>
```

admin.php

```
<?php
  // Имя пользователя и его пароль для аутентификации
  $username = 'rock';
  $password = 'roll';

  if (!isset($_SERVER['PHP_AUTH_USER']) ||
   !isset($_SERVER['PHP_AUTH_PW']) ||
   ($_SERVER['PHP_AUTH_USER'] != $username) ||
   ($_SERVER['PHP_AUTH_PW'] != $password)) {
   // Имя пользователя/пароль не действительны для отправки HTTP-заголовков,
   // подтверждающих аутентификацию
   header('HTTP/1.1 401 Unauthorized');
   header('WWW-Authenticate: Basic realm="Гитарные войны"');
   exit('<h2>Гитарные войны h2> Извините, вы должны ввести правильные ' .
      'имя пользователя и пароль, чтобы получить доступ к этой странице.');
  }
?>
```

authorize.php

Так как сценарий «Аутентификация» используется совместно двумя сценариями, вы можете быть уверены, что обе страницы будут принадлежать одной защищенной области. Это означает, что они используют одну и ту же пару «имя пользователя/пароль».

Совместно используемый сценарий размещается в самом начале сценария, так как он вызывает функции header().

```
<?php
 require_once('authorize.php');
?>

<html>
```

admin.php

```
<?php
 require_once('authorize.php');
?>

<html>
```

removescore.php

Код аутентификации в сценарии «Администрирование рейтингов» замещен одной строкой PHP-кода.

КЛЮЧЕВЫЕ МОМЕНТЫ

- PHP-сценарий может использовать HTTP-заголовки для того, чтобы управлять процессом передачи информации от сервера браузеру.
- Встроенная PHP-функция `header()` используется для передачи браузеру HTTP-заголовков, которые могут использоваться для переадресации страницы, управления форматом содержания страницы или чтобы запрашивать данные для аутентификации.
- Если HTTP-заголовок отправляется браузеру с помощью функции `header()`, ее вызов должен производиться перед отправкой любого другого содержания.
- Если страница защищается с использованием HTTP-аутентификации, имя и пароль, введенные пользователем, сохраняются в суперглобальном массиве `$_SERVER`.
- Basic realm (защищенная область) HTTP-аутентификации — это защищенная зона, ассоциируемая с определенным именем пользователя и его паролем и позволяющая защищать несколько страниц одновременно.
- Встроенная PHP-функция `exit()` останавливает действие сценария, предотвращая выполнение любого кода, следующего за ней.

не бывает глупых вопросов

В: Я все еще не могу понять, как Этел обошла защиту приложения «Гитарные войны». Что она сделала?

О: Она использовала слабость, вытекающую из того, что в приложении была защищена только одна страница («Администрирование рейтингов»), в то время как функция удаления рейтинга фактически зависит от двух страниц («Администрирование рейтингов» и «Удаление рейтинга»). На странице «Администрирование рейтингов» представлен список гиперссылок, переадресовывающих на страницу «Удаление рейтинга». Данные, определяющие, какой конкретно рейтинг должен быть удален, передаются в составе URL, давая возможность сценарию «Удаление рейтинга» получить к ним доступ через суперглобальный массив `$_GET`. Если у вас есть возможность составить действительный URL для открытия страницы «Удаление рейтинга», вы можете удалить рейтинг даже без открытия страницы «Администрирование рейтингов». Именно это и сделала Этел.

В: Но как она узнала, как составить действительный URL для открытия страницы «Удаление рейтинга»?

О: Она достаточно изобретательна, но для решения этой задачи совсем не обязательно быть гением. Вспомните ее упоминание о том, что она сохранила в закладках браузера страницу «Удаление рейтинга» в то время, когда сайт не был защищен. Но закладка — это не что иное, как URL страницы, на базе которого она и получила возможность создать URL для прямого доступа к странице «Удаление рейтинга» без необходимости переходить на нее по гиперссылке через страницу «Администрирование рейтингов».

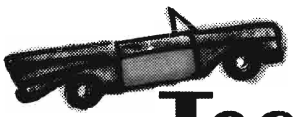
В: Хорошо, но ведь рейтинг был введен заново после предыдущей атаки. Разве это не означает, что старый URL стал уже недействительным? Данные-то ведь изменились.

О: Очень верное замечание. Но не забывайте, что Этел достаточно изобретательна. Ей не составило большого труда посмотреть главную страницу приложения «Гитарные войны», чтобы увидеть новые данные, которые она затем вставила в старый URL, что и позволило ей без проблем удалить новый рейтинг. Очень важно никогда не недооценивать способности людей в «творческом» подходе к вашим сценариям и использованию любых их слабых мест.

В: Хорошо. Защита страниц «Администрирование рейтингов» и «Удаление рейтинга» останавливает Этел, но разве она не создает дополнительных препятствий для удаления рейтинга на законных основаниях?

О: Совсем нет. Без использования принципа защищенной области удаление рейтинга на законных основаниях действительно превратилось бы в непростую задачу, потому что вам пришлось бы вводить имя пользователя и его пароль отдельно и в странице «Администрирование рейтингов», и в странице «Удаление рейтинга». Но не забывайте, что эта область определена для обеих страниц, то есть обе эти страницы принадлежат к одной зоне безопасности. И если вы прошли процедуру аутентификации для одной страницы данной защищенной области, введенные вами имя пользователя и пароль запоминаются для всех сценариев, принадлежащих этой области. Конечный результат заключается в том, что достаточно однажды успешно ввести имя пользователя и его пароль, чтобы разблокировать обе страницы.

Никогда не недооценивайте способности людей в «творческом» подходе к вашим сценариям и использованию любых их слабых мест.

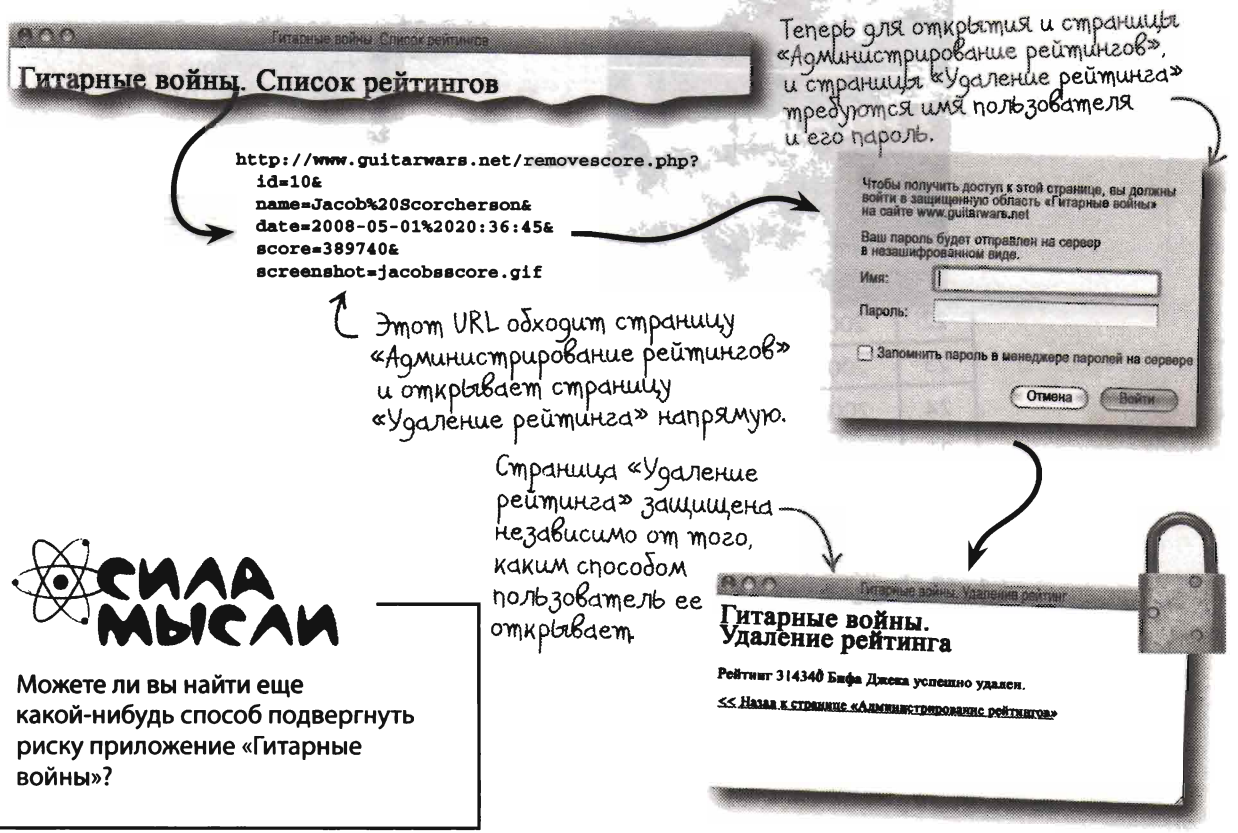


Тест-драйв

Создайте сценарий «Аутентификация» и включите его в сценарии «Администрирование рейтингов» и «Удаление рейтинга», чтобы защитить их.

Создайте новый текстовый файл с именем `authorize.php` и введите в него код аутентификации. Затем замените в сценарии `admin.php` код фактической аутентификации на код включения в него сценария «Аутентификация». Добавьте такое же выражение `require_once` в начало сценария `removescore.php`, чтобы защитить и его с помощью HTTP-аутентификации.

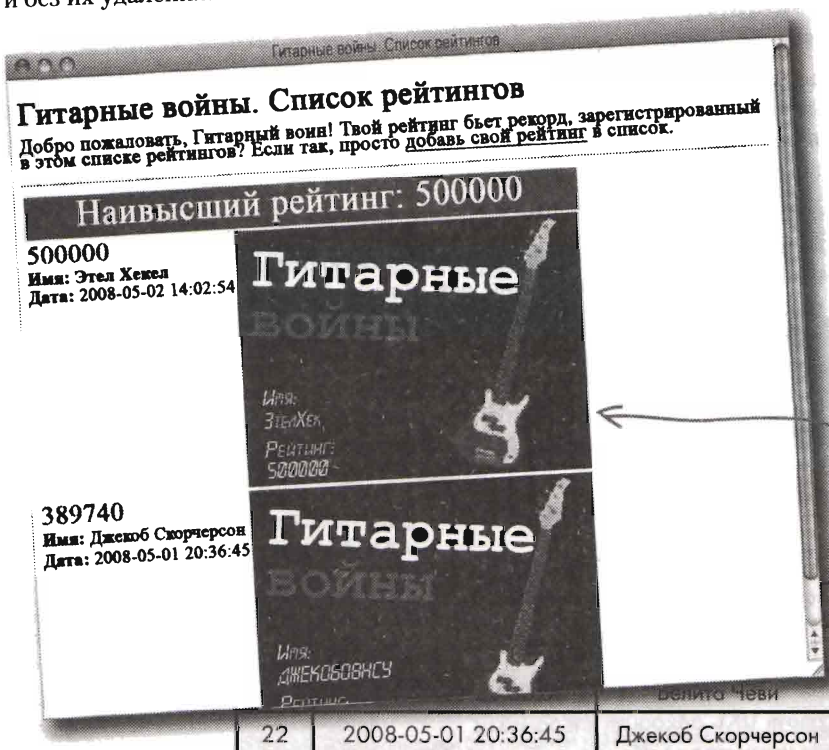
Загрузите все сценарии на ваш веб-сервер и попытайтесь напрямую открыть в браузере сценарий «Удаление рейтинга». Возможно, вам придется закрыть все предыдущие сессии, открытые в вашем браузере с HTTP-аутентификацией. Большинство браузеров запоминают параметры защищенной области, чтобы вам не приходилось многократно вводить имя пользователя и его пароль.



Можете ли вы найти еще какой-нибудь способ подвергнуть риску приложение «Гитарные войны»?

«Гитарные войны». Эпизод II: атака рейтинговых клонов

К сожалению, счастье в мире «Гитарных войн» длилось недолго, потому что на месте законных рейтингов появляются поддельные, сея гнев и раздор во всем пространстве приложения. Очевидно, что изменить список рейтингов приложения «Гитарные войны» вполне возможно и без их удаления. Но как?

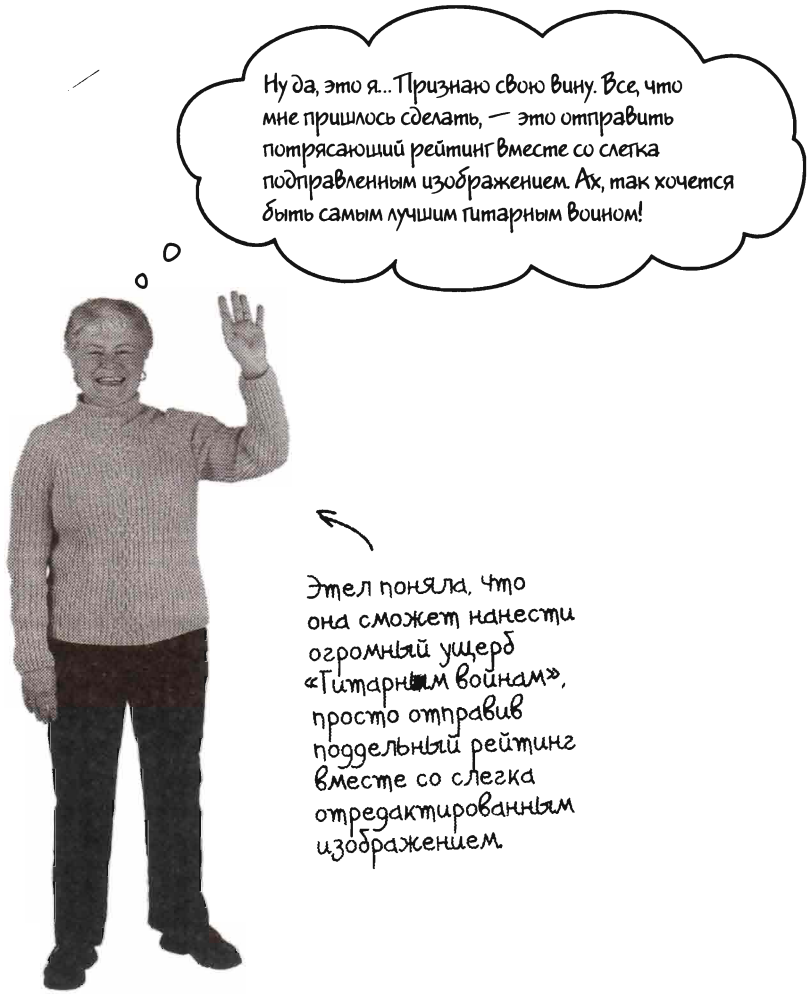


Рейтинг Этел вызывает подозрение из-за грубо подделанного изображения и того, что число круглое: ровно 500 000.

			score	screenshot
		Белита Чеви	282470	belitasscore.gif
22	2008-05-01 20:36:45	Джекоб Скорчерсон	389740	jacobsscore.gif
23	2008-05-01 20:37:02	Невил Йохансон	98430	nevilsscore.gif
24	2008-05-01 20:37:23	Пако Джасториус	127650	pacosscore.gif
25	2008-05-01 20:37:40	Фиц Лайрстон	186580	phizsscore.gif
26	2008-05-01 20:38:00	Кенни Левиц	64930	kennysscore.gif
27	2008-05-01 20:38:23	Жан Поль Джонс	243260	jeanpaulsscore.gif
28	2008-05-01 21:14:56	Леди Ги	308710	leddysscore.gif
29	2008-05-01 21:15:17	Тибоун Тэйлор	354190	tbonesscore.gif
30	2008-05-02 14:02:54	Этел Хекел	500000	ethelsscore.gif

Вычитание сложением

До сих пор мы работали исходя из предположения, что все рейтинги, загруженные вместе с изображениями, считаются подлинными. Но сейчас у нас есть все основания усомниться в этом. И нам совершенно ясно, кого можно в этом обвинить...



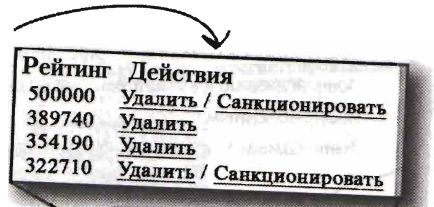
Напишите, как бы вы смогли решить проблему, связанную с тем, что у людей имеется возможность отправлять поддельные рейтинги в приложение «Гитарные войны»:

Безопасность требует присутствия человека

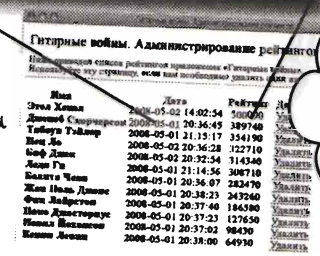
Даже в компьютеризированном мире, в котором мы живем, иногда невозможно обойтись без живого, дышащего, думающего гомо сапиенса. Трудно заменить человека в случаях, когда нужно проанализировать какую-то информацию и определить, подлинна она или нет. Мы ведем речь об арбитраже, когда на человека возлагается ответственность санкционировать прием информации, отправленной для веб-приложения, прежде чем выставлять ее на всеобщее обозрение.

Наделение человека полномочиями арбитра — отличный способ повысить достоверность данных, поступающих от пользователей.

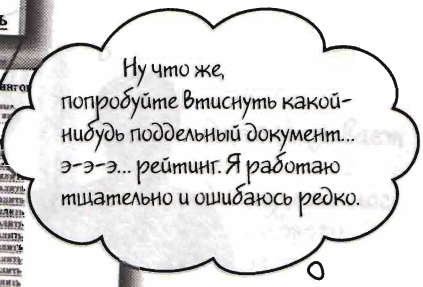
В случае применения арбитража новый рейтинг добавляется в базу данных, но широкой публике не виден до тех пор, пока арбитр не санкционирует это.

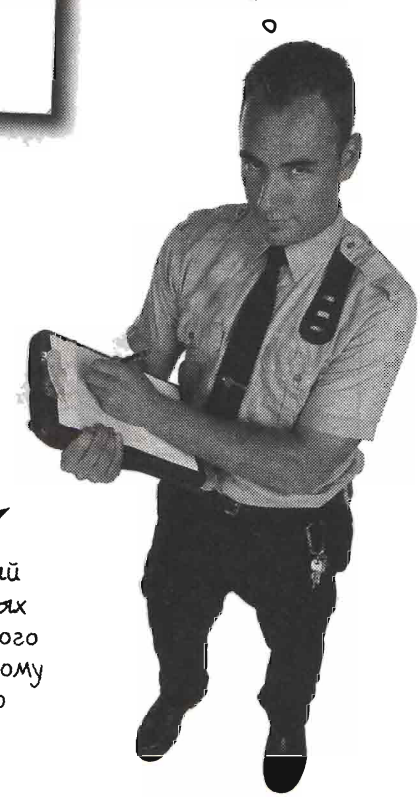


Изменение сценария «Администрирование рейтингов» предполагает добавление гиперссылки «Санкционировать» для каждого нового рейтинга, чтобы была возможность сделать его доступным для широкой публики.



Простое добавление нового рейтинга больше не означает, что он сразу же будет выставлен на обозрение широкой публике.





Наш бесстрашный арбитр «Гитарных войн»... Нет такого рейтинга, которому бы он полностью доверял.

Для приложения «Гитарные войны» полезно использование арбитража, проводимого человеком. Конечно, вполне возможно, что кто-нибудь подделает изображение рейтинга так тщательно, что это не сможет заметить и арбитр. Но сделать подобное не так просто, и это не умаляет значения арбитража как сдерживающего фактора. Не забывайте, что защита информации в PHP-приложениях основывается прежде всего на предупредительных мерах.

План арбитража в приложении «Гитарные войны»

Добавление арбитража, проводимого человеком, в приложение «Гитарные войны» — задача непростая, потому что ее решение связано с несколькими частями приложения. Нужно внести изменения в базу данных; должен быть создан новый сценарий для санкционирования; в страницу «Администрирование рейтингов» необходимо добавить соответствующие гиперссылки для каждого рейтинга; наконец, на главную страницу теперь должны выводиться только санкционированные рейтинги. При таком большом количестве необходимых изменений важно составить план и выполнять все действия поэтапно.

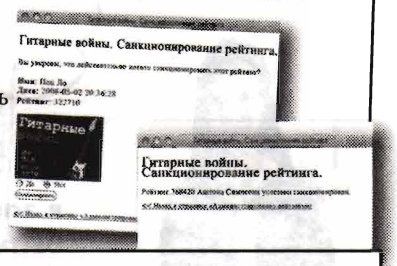
1 Используйте SQL-запрос ALTER для добавления колонки approved (санкционировано) в таблицу.

Давайте начнем с таблицы базы данных, которой необходима новая колонка, чтобы отслеживать, санкционирована публичная демонстрация рейтинга или нет.

id	date	name	rating	screenshot	approved
24	2008-05-01 21:14:56	Лидл Ти	308710	ledbetter.gif	0
25	2008-05-01 21:15:17	Тибоуи Тэйлор	354190	thibodeaux.gif	0
26	2008-05-02 14:02:54	Этел Хекел	500000	ethelcorn.gif	0
27	2008-05-02 20:32:54	Биф Джек	314340	bifjackson.gif	0
28	2008-05-01 20:36:28	Пол Ло	322710	paulo.gif	0

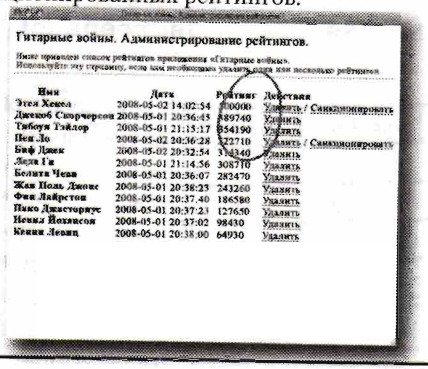
2 Создайте сценарий «Санкционирование рейтинга», который будет санкционировать новый рейтинг (заносить единицу в колонку approved).

Кроме того, что ваша база данных получила возможность сохранять информацию о санкционировании рейтинга, вам необходим также сценарий для управления этим процессом. Сценарий «Санкционирование рейтинга» должен будет находить определенную запись с данными рейтинга и изменять для нее значение колонки approved.



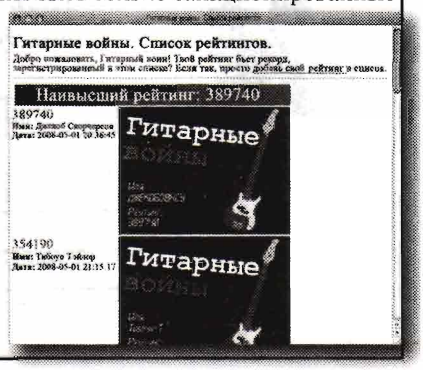
3 Измените страницу «Администрирование рейтингов», добавив в нее гиперссылки «Санкционировать» для каждого еще не санкционированного рейтинга.

Сценарий «Санкционирование рейтинга» — это внутренний сценарий, который не должен быть доступен напрямую. Вместо этого доступ к нему осуществляется через гиперссылки «Санкционировать», генерируемые сценарием «Администрирование рейтингов»; при этом на экран браузера выводятся гиперссылки только для еще не санкционированных рейтингов.



4 Измените SQL-запрос, в результате выполнения которого из базы извлекаются данные, чтобы на главную страницу выводились только санкционированные рейтинги.

На последнем этапе мы должны убедиться в том, что все действия, направленные на решение вопроса, связанного с санкционированием рейтингов, отразились на списке демонстрируемых рейтингов. В нем теперь должны быть только санкционированные рейтинги. И если это не так, все наши предыдущие изменения были бесполезными.



Предоставьте место для санкций с помощью SQL-запроса ALTER

Для того чтобы добавить в таблицу `guitarwars` новую колонку `approved`, необходимо сделать один SQL-запрос `ALTER TABLE`, который мы уже использовали ранее.

```
ALTER TABLE guitarwars
ADD COLUMN approved TINYINT
```

Имя типа данных MySQL BOOL является альтернативным именем (псевдонимом) для типа данных TINYINT, и вы можете использовать любое из них.

Новая колонка имеет тип `TINYINT`, и данные, содержащиеся в ней, могут иметь значения 0 (ноль) для несанкционированных рейтингов или 1 для санкционированных. Поэтому все новые рейтинги должны иметь первоначальное значение этой колонки, равное нулю, показывая тем самым, что они **изначально не санкционированы**.

СДЕЛАНО

1. **Используйте SQL-запрос ALTER для добавления колонки approved (санкционировано) в таблицу.**

Минуточку. Я сомневаюсь, что вы можете просто добавить колонку в таблицу без изменения сценария «Добавление рейтинга». Разве SQL-запрос INSERT не должен добавлять данные и в новую колонку?

Это так: появление новой колонки в таблице означает, что в запросе INSERT сценария «Добавление рейтинга» должны быть указаны данные для нее.

Очень важно не упустить из вида тот факт, что PHP-приложение — это большой, хорошо слаженный ансамбль, состоящий из многих разнородных частей: базы данных, включающей таблицы с колонками и записями, PHP-кода, HTML-кода и обычно CSS-кода. Добавление новой колонки `approved` в таблицу `guitarwars` вызывает необходимость изменения SQL-запроса `INSERT` в сценарии «Добавление рейтинга»:

Во всех вновь добавляемых записях рейтингов значение колонки approved устанавливается равным нулю... Не санкционировано!

```
INSERT INTO guitarwars
VALUES (0, NOW(), '$name', '$score','$screenshot', 0)
```

id	date	name	score	screenshot	approved
...					
30	2008-05-02 14:02:54	Этел Хекел	500000	ethelsscore.gif	0
31	2008-05-02 20:32:54	Биф Джек	314340	biffsscore.gif	0
32	2008-05-02 20:36:38	Пец Ло	322710	pezsscore.gif	0

Когда добавляется новая запись, значение ее колонки approved устанавливается равным нулю, поэтому и соответствующий этой записи рейтинг изначально является несанкционированным.

Время поработать

Сценарий «Санкционирование рейтинга» по своей структуре очень похож на сценарий «Удаление рейтинга» с той лишь разницей, что его функцией является не удаление, а санкционирование. Добавьте в сценарий «Санкционирование рейтинга» пропущенный код, обеспечивающий защиту страницы и санкционирование рейтинга, данные которого были переданы в составе URL.

```
<?php
.....;
?>
...
<?php
  require_once('appvars.php');
  require_once('connectvars.php');
  ...
  if (isset($_POST['submit'])) {
    if (.....) {
      // Соединение с базой данных
      $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

      // Санкционирование рейтинга путем установки значения 1
      // для колонки approved таблицы guitarwars

      $query = "UPDATE guitarwars SET.....";
      mysqli_query($dbc, $query);
      mysqli_close($dbc);

      // Вывод на экран пользователя подтверждения об успешном санкционировании
      echo .....
    }
    else {
      echo .....
    }
  }
  ...
  echo '<p><a href="          ">&lt;&lt; Назад к странице &quot;Админ&quot;</a></p>';
?>
...
```

Решение задачи

Сценарий «Санкционирование рейтинга» по своей структуре очень похож на сценарий «Удаление рейтинга» с той лишь разницей, что его функцией является не удаление, а санкционирование. Добавьте в сценарий «Санкционирование рейтинга» пропущенный код, обеспечивающий защиту страницы и санкционирование рейтинга, данные которого были переданы в составе URL.

```
<?php
        require_once('authorize.php') ;
?>
...
<?php
  require_once('appvars php');
  require_once('connectvars.php');
  ...
  if (isset($_POST['submit'])) {
    if ( $_POST['confirm'] == 'Yes' ) {
      // Соединение с базой данных
      $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

      // Санкционирование рейтинга путем установки значения 1
      // для колонки approved таблицы guitarwars
      $query = "UPDATE guitarwars SET  approved = 1 WHERE id = '$id' ";
      mysqli_query($dbc, $query);
      mysqli_close($dbc);
      // Вывод на экран пользователя подтверждения об успешном санкционировании
      echo  echo '<p>Рейтинг '. $score . ' для пользователя ' . $name . ' успешно санкционирован.';
    }
    else {
      echo  echo '<p class="error">Извините, возникли проблемы с санкционированием рейтинга.</p>';
    }
  }
  ...
  echo '<p><a href="  admin.php ">&lt;&lt; Назад к странице &quot;Админ&quot;</a></p>';
?>
...
```

Включение сценария «Аутентификация» — это все, что необходимо, чтобы защитить страницу «Санкционирование рейтинга» с помощью имени пользователя и его пароля, но это должно быть сделано в самом начале, так как сценарий «Аутентификация» полагается на HTTP-заголовки.

2 **Создайте сценарий «Санкционирование рейтинга», который будет санкционировать новый рейтинг (заносить единицу в колонку approved).**

СДЕЛАНО

Для того чтобы санкционировать нужный рейтинг, должна быть выбрана запись с соответствующим идентификатором.

Занесение единицы в колонку approved санкционирует рейтинг.

Подтвердите пользователю успешное санкционирование рейтинга выводом на экран браузера его имени и санкционированного рейтинга.

Очень важно сообщить обо всех случаях, когда санкционирование рейтинга не удалось, аналогично тому, как в других сценариях приложения «Гитарные войны» выводятся сообщения об ошибках.

Для более удобного возвращения на страницу «Администрирование рейтингов» на нее указывает эта гиперссылка.

не бывает глупых вопросов

В: Почему при санкционировании рейтинга не передается также и имя файла изображения, подтверждающего рейтинг?

О: Потому что при санкционировании рейтинга нужны данные только для того, чтобы найти в таблице запись с необходимым рейтингом. Даты, имени и рейтинга вполне достаточно, чтобы найти нужную запись и изменить с нуля на единицу значение колонки approved.

В: Использование в качестве значений ноля и единицы для колонки approved выглядит не совсем понятным. Нет ли других способов представления этих данных?

О: Есть. Тип MySQL-данных ENUM, сокращенное от enumerated (пронумерованный), позволяет вам создать колонку с ограниченным перечнем возможных значений. Поэтому вместо добавления колонки approved с типом данных TINYINT, в которой вы сохраняли бы 0 или 1, можете добавить колонку с типом данных ENUM и сохранять в ней yes (да) или no (нет), как показано ниже:

```
ALTER TABLE guitarwars
ADD COLUMN approved ENUM('yes', 'no')
```

Время поработать

Данные рейтинга, который необходимо санкционировать, передаются в сценарий «Санкционирование рейтинга» через гиперссылки «Санкционировать», создаваемые сценарием «Администрирование рейтингов». Добавьте пропущенный код в сценарий «Администрирование рейтингов» для создания этих гиперссылок.

```
...
  // Извлечение данных из массива рейтингов в цикле.
   // Форматирование данных записей в виде кода HTML
   echo '<table>';
   echo '<tr><th>Имя</th><th>Дата</th><th>Рейтинг</th><th>Действие</th></tr>';
   while ($row = mysqli_fetch_array($data)) {
     // Вывод данных рейтинга
     echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
     echo '<td>' . $row['date'] . '</td>';
     echo '<td>' . $row['score'] . '</td>';
     echo '<td><a href="removescore.php?id=' . $row['id'] . '&amp;date=' . $row['date'] .
          '&amp;name=' . $row['name'] . '&amp;score=' . $row['score'] .
          '&amp;screenshot=' . $row['screenshot'] . '">Удалить</a>';
     if ( ..... ) {
       echo .....
            .....
            .....
     }
     echo '</td></tr>';
   }
  echo '</table>';
...
```

Совет: гиперссылка «Санкционировать» должна быть только у несанкционированных рейтингов.

Решение задачи

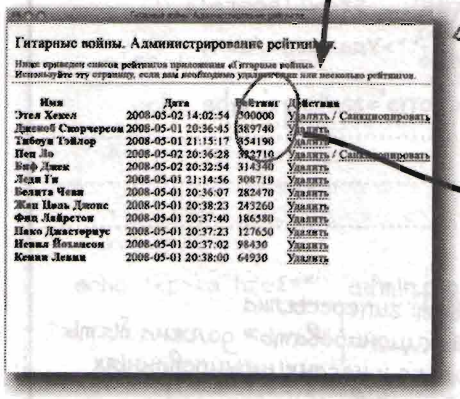
Данные рейтинга, который необходимо санкционировать, передаются в сценарий «Санкционирование рейтинга» через гиперссылки «Санкционировать», создаваемые сценарием «Администрирование рейтингов». Добавьте пропущенный код в сценарий «Администрирование рейтингов» для создания этих гиперссылок.

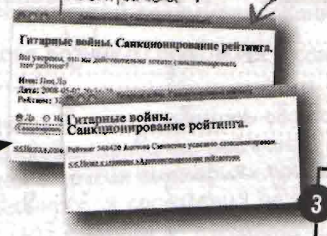
```
// Извлечение данных из массива рейтингов в цикле.
// Форматирование данных записей в виде кода HTML
echo '<table>';
echo '<tr><th>Имя</th><th>Дата</th><th>Рейтинг</th><th>Действие</th></tr>';
while ($row = mysqli_fetch_array($data)) {
    // Вывод данных рейтинга
    echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
    echo '<td>' . $row['date'] . '</td>';
    echo '<td>' . $row['score'] . '</td>';
    echo '<td><a href="removescore.php?id=' . $row['id'] . '&amp;date=' . $row['date'] .
      '&amp;name=' . $row['name'] . '&amp;score=' . $row['score'] .
      '&amp;screenshot=' . $row['screenshot'] . '">Удалить</a>';
    if ( $row['approved'] == '0' ) {
      echo ' / <a href="approvescore.php?id=' . $row['id'] . '&amp;date=' . $row['date'] .
        '&amp;name=' . $row['name'] . '&amp;score=' . $row['score'] . '&amp;screenshot=' .
        $row['screenshot'] . '">Санкционировать </a>';
    }
    echo '</td></tr>';
}
echo '</table>';
...
```

Прежде чем создавать гиперссылку «Санкционировать», производится проверка санкционирована эта запись или нет

Создание гиперссылки «Санкционировать» таким образом, чтобы идентификатор записи, дата регистрации, имя пользователя, рейтинг и имя файла изображения, подтверждающего его, были отправлены в составе URL.

Гиперссылка «Санкционировать» связывает страницу «Администрирование рейтингов» со страницей «Санкционирование рейтинга».





СДЕЛАНО

3 **Измените страницу «Администрирование рейтингов», добавив в нее гиперссылки «Санкционировать» для каждого еще не санкционированного рейтинга.**

Несанкционированные рейтинги недостойны внимания

Итак, вся инфраструктура для арбитража в приложении «Гитарные войны» уже готова. Чего еще не хватает, так это заключительного этапа, на котором должна быть обеспечена демонстрация на главной странице только санкционированных рейтингов. Для этого необходимо подправить SQL-запрос `SELECT` так, чтобы в результате его выполнения извлекались только записи со значением колонки `approved`, равным 1, то есть санкционированные. Это достигается включением в запрос условного выражения `WHERE`.

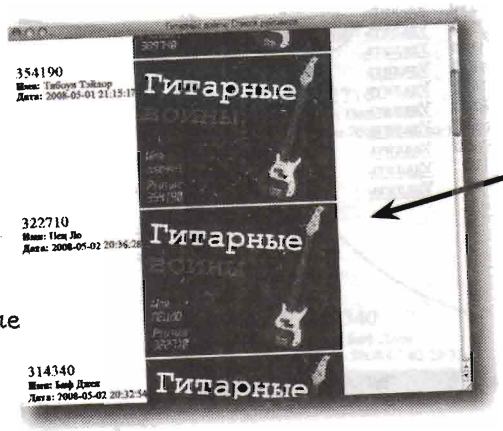
Используйте условное выражение WHERE для того, чтобы извлечь данные, соответствующие определенному значению какой-либо колонки.

```
SELECT * FROM guitarwars
WHERE approved = 1
ORDER BY score DESC, date ASC
```

Если колонка approved имеет любое значение, но не 1, рейтинг не будет включен в видимый широкой публикой перечень.

Включение в этот запрос условного выражения `WHERE` изымает из результата запроса любые несанкционированные рейтинги, в состав которых входят также все новые рейтинги. Это дает арбитру возможность просмотреть их и решить, удалить их из базы данных или сделать видимыми для широкой публики (санкционировать).

id	date	name	score	screenshot	approved
		...			
28	2008-05-01 21:14:56	Леди Ги	308710	leddysscore.gif	1
29	2008-05-01 21:15:17	Тибоун Тэйлор	354190	tbonesscore.gif	1
30	2008-05-02 14:02:54	Этел Хекел	500000	ethelsscore.gif	0
31	2008-05-02 20:32:54	Биф Джек	314340	biffsscore.gif	1
32	2008-05-02 20:36:38	Пец Ло	322710	pezsscore.gif	1



Теперь только санкционированные рейтинги выводятся на главной странице.

4 **Измените SQL-запрос, в результате выполнения которого из базы извлекаются данные, чтобы на главную страницу выводились только санкционированные рейтинги.**

СДЕЛАНО

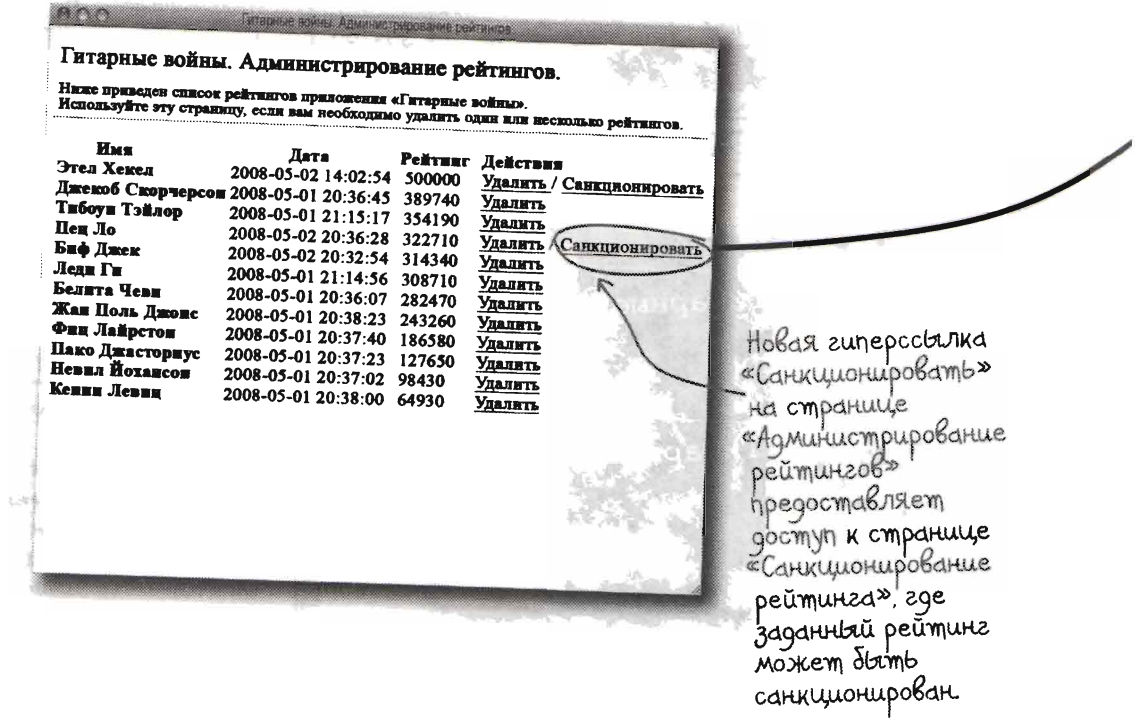
Тест-драйв

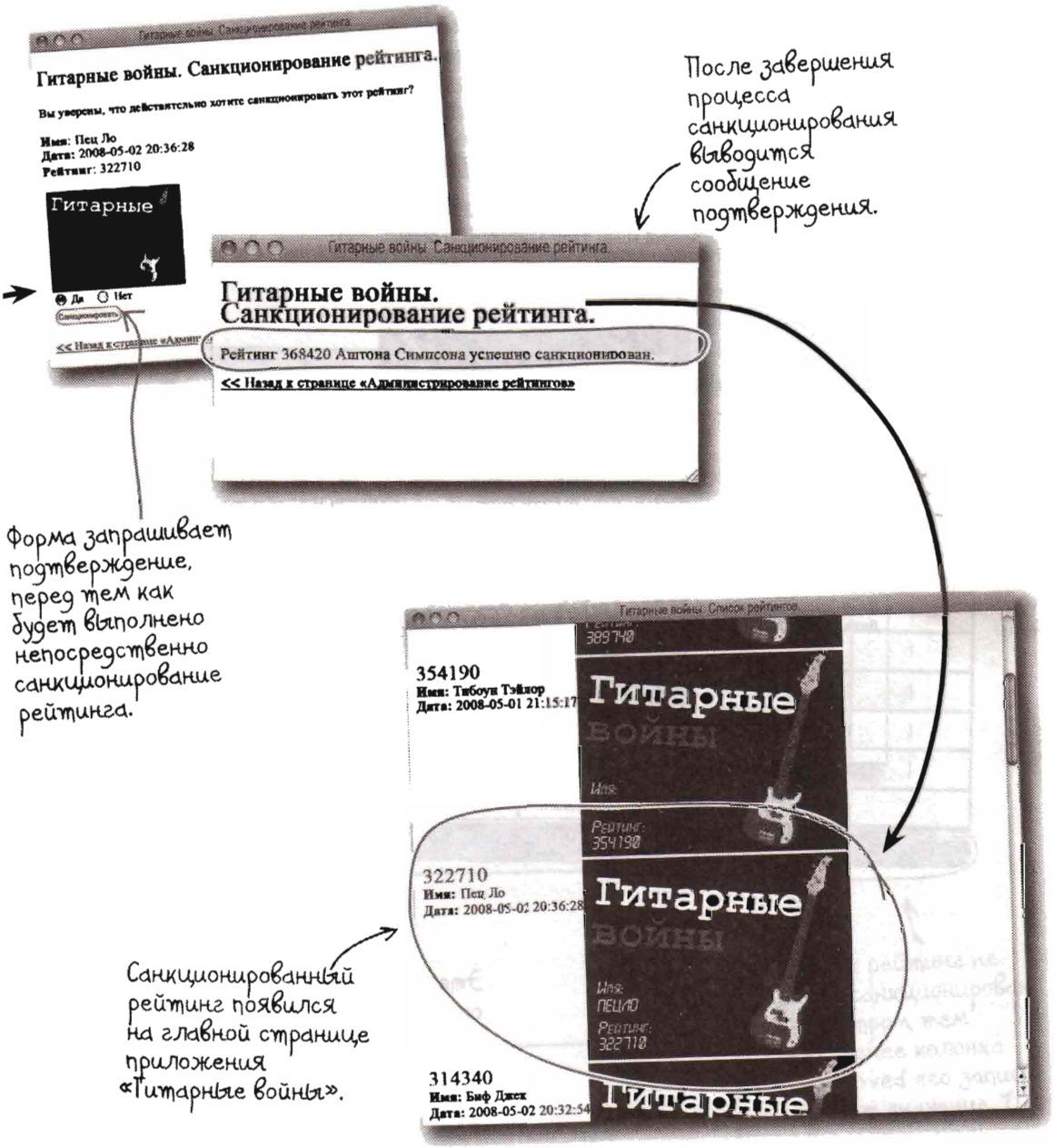
Создайте сценарий «Санкционирование рейтинга» и откорректируйте остальные части приложения «Гитарные войны», чтобы использовать этот сценарий.

Используя инструментальную программу MySQL, выполните SQL-запрос `ALTER`, чтобы добавить к таблице `guitarwars` новую колонку `approved`. Затем измените запрос `INSERT` в сценарии `addscore.php` так, чтобы при добавлении новых записей колонке `approved` присваивалось бы значение «ноль».

Потом создайте новый текстовый файл с именем `approvedscore.php` и введите в него код сценария «Санкционирование рейтинга». Добавьте в сценарий `admin.php` код, создающий гиперссылку «Санкционировать» для тех рейтингов, которые еще не санкционированы. И, наконец измените SQL-запрос `SELECT` в сценарии `index.php` так, чтобы на главной странице были видны только санкционированные рейтинги.

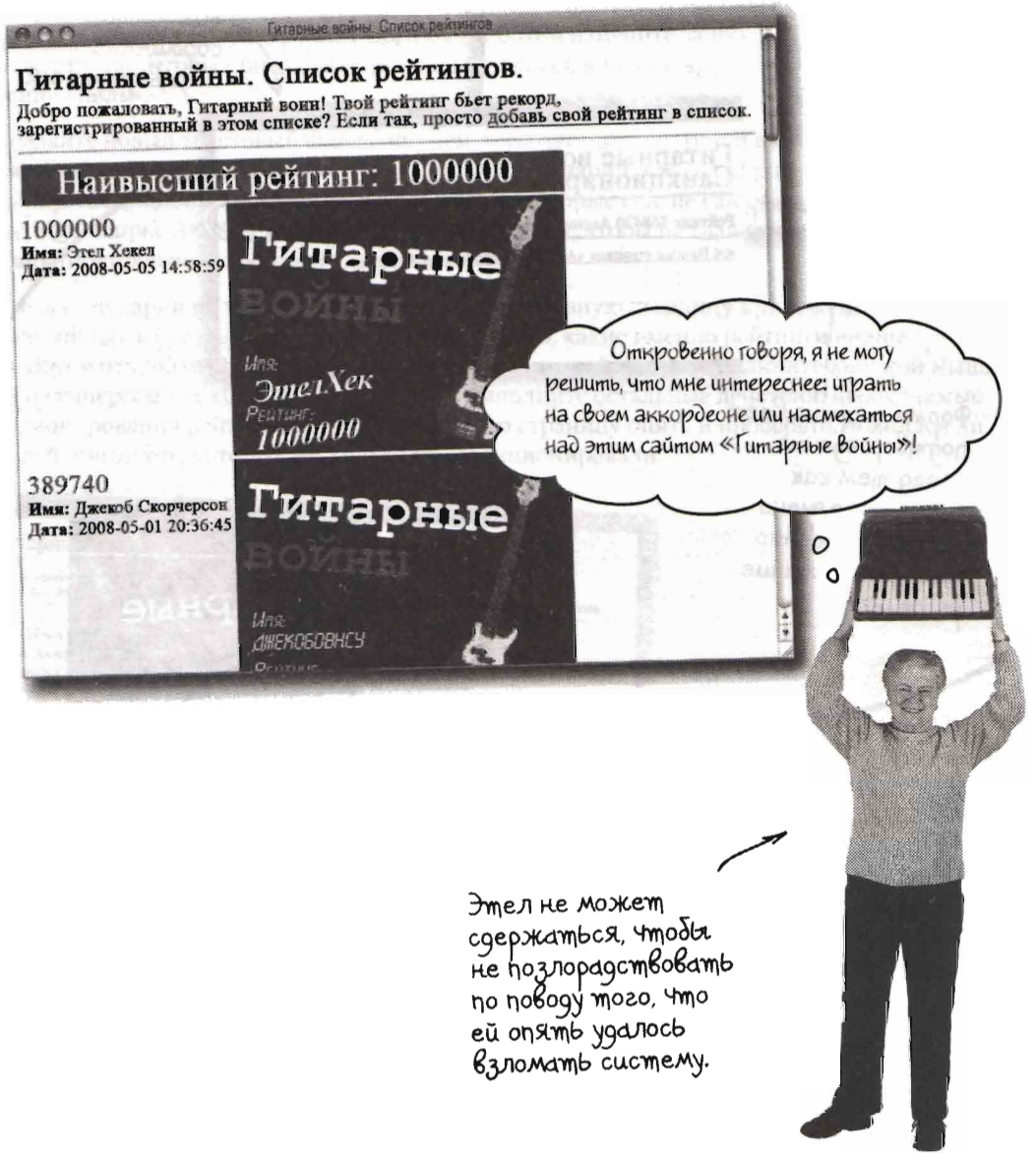
Загрузите все сценарии на ваш веб-сервер и откройте главную страницу приложения «Гитарные войны» в браузере. Обратите внимание на то, какие именно рейтинги видны в списке. Затем откройте страницу «Администрирование рейтингов», щелкните кнопкой мыши на одной из гиперссылок «Санкционировать» и выполните остальные действия, необходимые для санкционирования рейтинга. Откройте главную страницу опять и проверьте, появился ли в списке рейтингов тот, который вы только что санкционировали.




Гитарные войны. Санкционирование рейтинга.
Вы уверены, что действительно хотите санкционировать этот рейтинг?
Имя: Пец Ло
Дата: 2008-05-02 20:36:28
Рейтинг: 322710
Гитарные
Да
Нет
Санкционировать
<< Назад к странице «Администрирование рейтингов»
Гитарные войны. Санкционирование рейтинга.
Гитарные войны.
Санкционирование рейтинга.
Рейтинг 368420 Аштона Симпсона успешно санкционирован.
<< Назад к странице «Администрирование рейтингов»
После завершения процесса санкционирования выводится сообщение подтверждения.
Форма запрашивает подтверждение, перед тем как будет выполнено непосредственно санкционирование рейтинга.
Гитарные войны. Список рейтингов
354190
Имя: Тибоун Тэйлор
Дата: 2008-05-01 21:15:17
322710
Имя: Пец Ло
Дата: 2008-05-02 20:36:28
314340
Имя: Биф Джек
Дата: 2008-05-02 20:32:54
Гитарные
Войны
Санкционированный рейтинг появился на главной странице приложения «Гитарные войны».

Хакерская атака в миллион очков

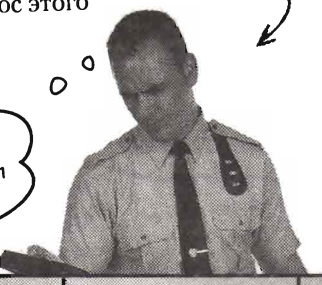
Введение арбитража в приложение «Гитарные войны» значительно повысило его защиту, но ему еще далеко до того, чтобы стать пуленепробиваемым. Похоже, нашему коварному лазутчику удалось найти новое слабое место в приложении и как-то втиснуть свой рейтинг, минуя арбитра. Этел нужно остановить навсегда, чтобы восстановить доверие на всем пространстве «Гитарных войн».



Все зависит от модерирования?

Хотя у арбитра нет ни тени сомнения в том, что он никогда не санкционировал рейтинг Этел, этот злополучный рейтинг, тем не менее, здесь, выставлен на обозрение широкой публике, со значением колонки `approved`, равным единице. Мы знаем, что при выполнении сценария «Добавление рейтинга» значение колонки `approved` для каждого нового добавляемого рейтинга устанавливается равным нулю, потому что мы только что внесли соответствующие изменения в SQL-запрос этого сценария. Происходит что-то совершенно непонятное!

Арбитр «Гитарных войн» не может понять, что случилось.



id	date	name	score	screenshot	approved
21	2008-05-01 20:36:07	Белита Чеви	282470	belitasscore.gif	1
22	2008-05-01 20:36:45	Джекоб Скорчерсон	389740	jacobsscore.gif	1
23	2008-05-01 20:37:02	Невил Йохансон	98430	nevilsscore.gif	1
24	2008-05-01 20:37:23	Пако Джасториус	127650	pacosscore.gif	1
25	2008-05-01 20:37:40	Фиц Лайрстон	186580	phizsscore.gif	1
26	2008-05-01 20:38:00	Кенни Левиц	64930	kennysscore.gif	1
27	2008-05-01 20:38:23	Жан Поль Джонс	243260	jeanpaulsscore.gif	1
28	2008-05-01 21:14:56	Леди Ги	308710	leddysscore.gif	1
29	2008-05-01 21:15:17	Тибоун Тэйлор	354190	tbonesscore.gif	1
31	2008-05-02 20:32:54	Биф Джек	314340	biffsscore.gif	1
32	2008-05-02 20:36:38	Пец Ло	322710	pezsscore.gif	1
33	2008-05-05 14:58:59	Этел Хекел	1000000	ethelsscore2.gif	1

Этот рейтинг не был санкционирован арбитром, тем не менее колонка `approved` его записи имеет значение 1, в результате чего он виден широкой публике.

СИЛА МЫСЛИ

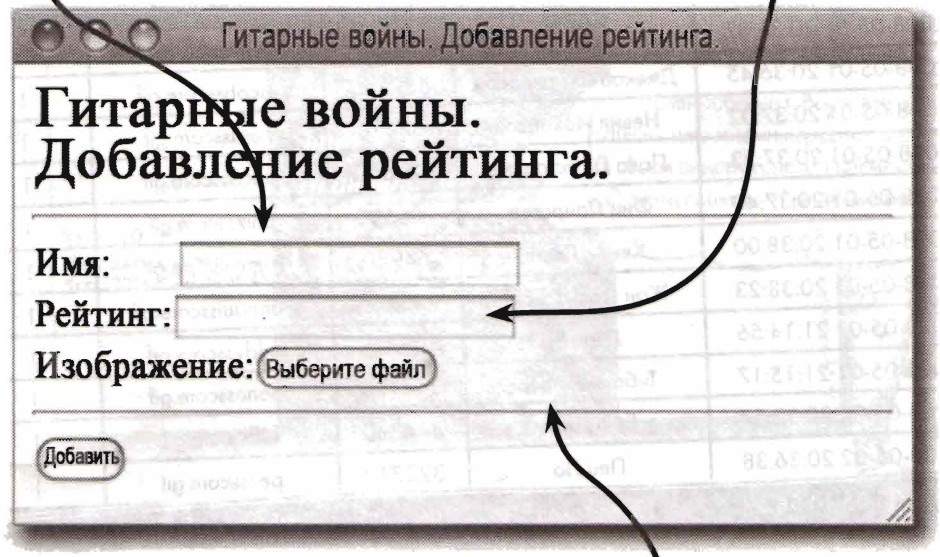
Как, по вашему мнению, фальшивый рейтинг Этел миновал арбитра?

ТЯЖЕЛАЯ АТЛЕТИКА ДЛЯ МЫСЛИ

Как оказалось, атака Этел в миллион очков не имеет ничего общего с формой «Санкционирование рейтинга». Ее злой умысел направлен исключительно на форму «Добавление рейтинга». Ниже приведен фрагмент данных, которые Этел ввела в форму «Добавление рейтинга» для достижения своей цели. Введите такие же данные в свою форму и добавьте рейтинг. Что произойдет, как вы думаете?

Не забудьте про пробел после символов «--» здесь.

```
Этел Хекел                                    1000000', 'ethelsscore2.gif', 1) --
```

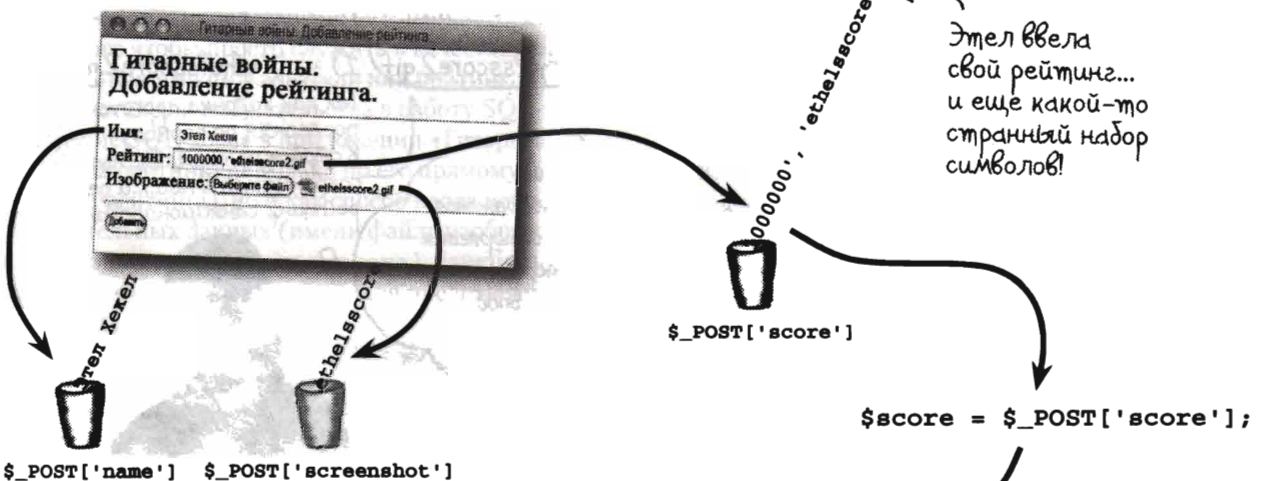


Это может быть любой файл изображения в формате GIF или JPEG размером не более 32 Кбайт.

ethelsscore2.gif

Как именно она сделала это?

Для того чтобы понять, что происходит во время этой изощренной атаки, давайте проследим весь путь, проходимый данными через сценарий «Добавление рейтинга».



Необычное значение поля формы Score сохраняется в переменной $score и переносится непосредственно в запрос INSERT.

```
INSERT INTO guitarwars
VALUES (0, NOW(), '$name', '$score', '$screenshot', 0)
```

Поле данных «Рейтинг» формы «Добавление рейтинга» ожидает единственное целочисленное значение — такое, например, как 1 000 000, но вместо этого получает несколько значений, заключенных в одинарные кавычки, отделенные друг от друга запятыми, со странным двойным дефисом в конце. Очень странно.

Эти данные сохраняются в переменной `$score`, после чего вставляются в SQL-запрос `INSERT`. В результате мы получим совершенно бессмысленный рейтинг, так? Или здесь происходит что-то, не предвещающее ничего хорошего?

Время поработать

Используя данные, представленные на предыдущей странице, напишите полностью SQL-запрос для атаки в миллион очков. Убедитесь, что переменные в запросе заменены на данные формы. Прокомментируйте, к чему, по вашему мнению, все это приведет.

.....

.....

Решение задачи

Используя данные, представленные на предыдущей странице, напишите полностью SQL-запрос для атаки в миллион очков. Убедитесь, что переменные в запросе заменены на данные формы. Прокомментируйте, к чему, по вашему мнению, все это приведет.

```
INSERT INTO guitarwars
VALUES (0, NOW(), 'Этел Хекел', '1000000', 'ethelsscore2.gif', 1) -- ', 'ethelsscore2.gif', 0)
```

Этел каким-то образом создала свою собственную версию SQL-запроса, которая заменяет оригинальную.

Так как колонка approved последняя по порядку в структуре таблицы, эта единица сохраняется в ней... и делает рейтинг санкционированным.

Этот запрос выглядит очень странно. Имя файла изображения, подтверждающего рейтинг, повторяется дважды. Кроме того, я не знаю, что делать с этим двойным дефисом... Это вообще работающий запрос?

Манипуляции сервером баз данных с помощью комментариев

Главным виновником атаки Этел в миллион очков является, как это ни странно, SQL-комментарий. Текст от пробела, следующего за двойным дефисом, до конца строки игнорируется при интерпретации SQL-кода и обычно используется программистом для записи комментариев. Для правильной интерпретации комментария за двойным дефисом обязательно должен следовать пробел. А теперь давайте посмотрим на весь запрос Этел, включая этот изощренный прием, в корне меняющий содержание запроса.

Весь текст после символа комментария (--) до конца строки игнорируется интерпретатором SQL.

```
INSERT INTO guitarwars
VALUES (0, NOW(), 'Этел Хекел', '1000000', 'ethelsscore2.gif', 1) -- '
```

Выглядит более осмысленно? Символ комментария аккуратно удаляет остаток SQL-кода, не давая серверу повода генерировать сообщение об ошибке и позволяя запросу версии Этел проскользнуть без сучка, без задоринки. В результате новый запрос немедленно санкционируется, не оставляя арбитру ни малейшего шанса перехватить и проверить его.

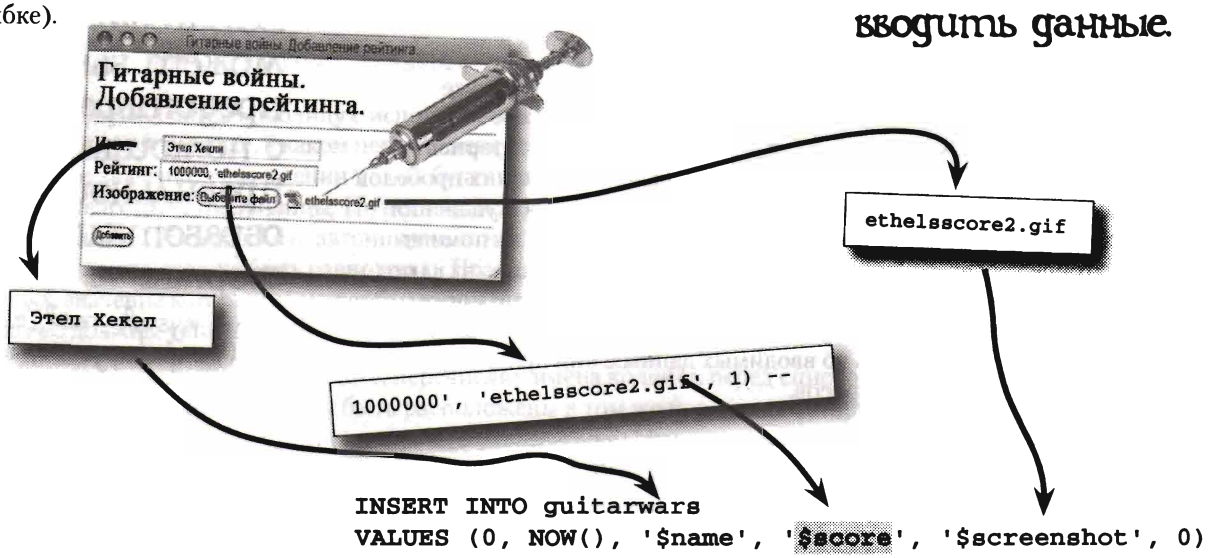
В результате выполнения сервером запроса, подделанного Этел, ее рейтинг был санкционирован.

id	date	name	score	screenshot	approved
...					
33	2008-05-05 14:58:59	Этел Хекел	1000000	ethelsscore2.gif	1

Форма «Добавление рейтинга» была подвергнута атаке «Внедрение SQL-кода»

Поля ввода данных формы являются слабым с точки зрения защиты местом приложения, потому что они предоставляют пользователю возможность вводить данные.

Атака Этел известна под названием «Внедрение SQL-кода» и представляет собой исключительно изощренный трюк, в котором данные формы используются как средство для радикального изменения содержания запроса. Вместо того чтобы просто служить в качестве средства передачи данных, таких, например, как имя пользователя или рейтинг, поля ввода данных формы используются для вмешательства в работу SQL-запроса. В случае атаки «Внедрение SQL-кода» в приложении «Гитарные войны» Этел использует поле ввода рейтинга не только по его прямому назначению (как средство для ввода рейтинга), но и как способ ввода непосредственно в SQL-запрос дополнительных данных (имени файла изображения, подтверждающего рейтинг, значения колонки `approval`, санкционирующего фальшивый рейтинг, и символа комментария в конце для предотвращения генерации сообщения об ошибке).



Не бывает глупых вопросов

В: Существуют ли в SQL другие символы комментариев кроме двойного дефиса (--)?

О: Да. Другой вариант однострочного комментария предполагает использование символа «хэш» (`#`). Аналогично тому, как это происходит при использовании символа двойного дефиса (`--`), весь текст после символа «хэш» (`#`) до конца строки игнорируется интерпретатором SQL. SQL также поддерживает многострочные комментарии. Так же как и в PHP, интерпретатором игнорируются многострочные комментарии, расположенные между символами `/*` и `*/`.

В: Достигнет ли успеха атака Этел «Внедрение SQL-кода», если колонка approved не будет последней в структуре таблицы?

О: Нет, и это действительно очень важное замечание. Этот конкретный запрос `INSERT` располагает данные в соответствии с порядком следования колонок в структуре таблицы. Расположение единицы в конце запроса выполняет поставленную задачу просто потому, что колонка `approved` является последней и размещена непосредственно после колонки `screenshot`.

Защитите свои данные от атаки «Внедрение SQL-кода»

Слабым местом, которое используется в атаке «Внедрение SQL-кода», является отсутствие проверки введенных в форму данных на наличие опасных символов. Опасными символами являются любые символы, которые потенциально могут изменить содержание SQL-запроса. К ним, в частности, относятся запятые, кавычки, двойные дефисы. Даже символы пробелов в конце данных могут представлять опасность. Пробелы в начале и в конце строк могут быть очень легко удалены с помощью встроенной PHP-функции `trim()`: достаточно просто вызвать ее с потенциально опасной строкой, переданной ей в качестве аргумента, перед тем как использовать эту строку в SQL-запросе.

Функция trim() удаляет пробелы в начале и в конце этих данных формы.

```
$name = trim($_POST['name']);
$score = trim($_POST['score']);
$screenshot = trim($_FILES['screenshot']['name']);
```

Но пробелами в начале и в конце строк не исчерпываются все проблемы, связанные с опасными символами. Вы можете также столкнуться с запятыми, кавычками, символами комментариев и еще, и еще... Поэтому дополнительно к удалению лишних пробелов в начале и в конце строк вам необходим способ поиска и удаления всех остальных проблемных символов. PHP приходит на помощь с другой встроенной функцией — `mysqli_real_escape_string()`, экранирующей потенциально опасные символы, которые могут неблагоприятно повлиять на выполнение SQL-запроса. Эти символы могут быть использованы во вводимых данных: они просто не будут изменять содержание запросов.

Атака «Внедрение SQL-кода» может быть предотвращена с помощью правильной обработки данных формы.

Совместный вызов функций `trim()` и `mysqli_real_escape_string()` устанавливает надежную линию обороны против атак «Внедрение SQL-кода».

Функция mysqli_real_escape_string() экранирует опасные символы, то есть преобразует их в такой вид, в котором они больше не рассматриваются SQL-интерпретатором как специальные символы.

```
$name = mysqli_real_escape_string($dbc, trim($_POST['name']));
$score = mysqli_real_escape_string($dbc, trim($_POST['score']));
$screenshot = mysqli_real_escape_string($dbc, trim($_FILES['screenshot']['name']));
```

Функция mysqli_real_escape_string() является функцией базы данных и поэтому ожидает в качестве первого аргумента ссылку на соединение с базой данных аналогично тому, как это происходит при вызове функции выполнения запроса.

Обработка данных этих трех полей ввода формы «Гитарные войны» с помощью функций `trim()` и `mysqli_real_escape_string()` значительно снижает шансы на успех атак «Внедрение SQL-кода». Но этих двух функций недостаточно. Возможно, существует способ сделать сам запрос менее уязвимым...

Более безопасный запрос INSERT (с использованием имен колонок)

Запрос INSERT может быть составлен в варианте, при котором совершенно точно указывается, какие значения сохраняются в каких колонках.

Кроме использования слабой защиты данных формы атака Этел «Внедрение SQL-кода» также полагается на тот факт, что колонка `approved` следует в структуре таблицы непосредственно после колонки `screenshot`. Это как раз то, что дало ей возможность сделать свое дело простым добавлением в конец запроса `INSERT` единицы, отправленной затем в колонку `approved`. Проблема этого запроса заключается в том, что он требует данных для всех колонок, а это повышает риск.

В идеале нам нет необходимости задавать значения колонок id и approved, так как они могут просто получать значения по умолчанию.

```
INSERT INTO guitarwars
VALUES (0, NOW(), '$name', '$score', '$screenshot', 0)
```

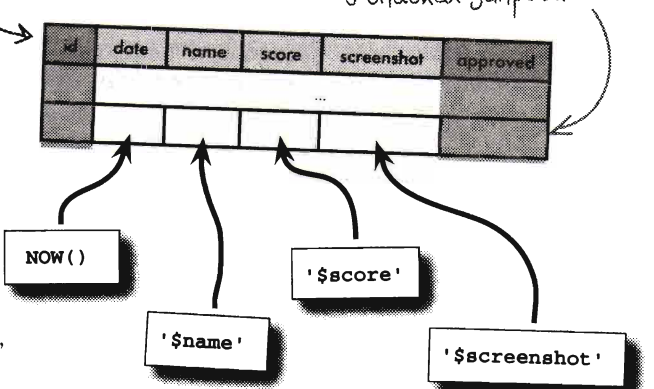
Когда данные добавляются в таблицу с использованием такого варианта запроса `INSERT`, они должны перечисляться в том порядке, в каком перечислены соответствующие колонки в структуре таблицы. Поэтому пятое значение в выражении `VALUE` будет отправлено в колонку `screenshot`: эта колонка находится на пятом месте в структуре таблицы. Но нет никакой нужды вводить значения колонок `id` и `approved`, поскольку колонка `id` создана как автоинкрементная, а значение колонки `approved` при добавлении новой записи всегда должно быть равно нулю. Правильнее было бы сосредоточиться на добавлении только тех данных, значение которых не может быть определено самим приложением. Значения же колонок `id` и `approved` должны устанавливаться по умолчанию как `AUTO_INCREMENT` и `НОЛЬ` соответственно.

Мы должны переписать запрос `INSERT`, перечислив имена колонок перед списком их значений. При этом значения во втором списке должны быть расположены в том же порядке, в котором перечислены колонки в первом списке. Это снижает риск установки пользователем значения колонки `approved`, так как этой колонки просто нет в запросе. Если такой вариант запроса `INSERT` показался вам знакомым, то это потому, что мы уже использовали его ранее в других примерах.

```
INSERT INTO guitarwars (date, name, score, screenshot)
VALUES (NOW(), '$name', '$score', '$screenshot')
```

Колонка id может быть исключена из списка, потому что она автоинкрементная.

В колонку approved не будет записано ничего, так как ее нет в списках запроса.



Эта версия составлена в варианте, при котором совершенно точно указывается, какие значения сохраняются в каких колонках, позволяя вам добавлять данные в таблицу, не беспокоясь о том, в каком порядке перечислены соответствующие им колонки в ее структуре. Фактически использование такого варианта запроса `INSERT` считается более предпочтительным стилем программирования, потому что вы явно указываете, куда именно должны быть записаны данные, а не полагаетесь в этом на структуру таблицы.

Подождите секундочку.
Я впервые слышу о том, что в таблицах MySQL можно применять значения по умолчанию.
Действительно ли существует такая возможность?

Это не только возможно, но и очень рекомендуется — присваивать колонкам значения по умолчанию, где это только возможно.

Выражение с ключевым словом DEFAULT позволяет вам указать для колонки значение по умолчанию. Если для колонки установлено значение по умолчанию, вы можете не указывать его в запросе INSERT, будучи уверенным, что при добавлении новой записи колонка получит это значение автоматически. Такой подход исключительно удобен применительно к колонке approved таблицы guitarwars. Теперь все, что нам необходимо, — это модифицировать таблицу guitarwars еще один раз, а именно установить для колонки approved значение, равное нулю (несанкционированно), в качестве значения по умолчанию.

Так как колонка approved уже существует, в этом запросе ALTER TABLE мы должны использовать выражение MODIFY COLUMN вместо ADD COLUMN.

```
ALTER TABLE guitarwars
MODIFY COLUMN approved TINYINT
DEFAULT 0
```

Выражение «DEFAULT 0» устанавливает, что колонка approved всегда будет принимать значение 0, если не будет явно указано другое.

Вы должны указать тип данных для колонки, чтобы дать понять, что это именно та колонка, которую вы первоначально создавали.

Так как колонка approved изменена таким образом, что для нее установлено значение по умолчанию, в результате выполнения нового и улучшенного запроса INSERT записи рейтингов теперь будут добавляться в таблицу даже без упоминания этой колонки в запросе. Такой стиль программирования считается хорошим, потому что отпадает необходимость ввода в явном виде данных, значения которых могут быть присвоены по умолчанию. Кроме того, это немного повышает уровень защиты приложения, так как колонка approved становится недоступной для потенциальных атак.

Проверка данных формы на достоверность никогда не может быть чрезмерной

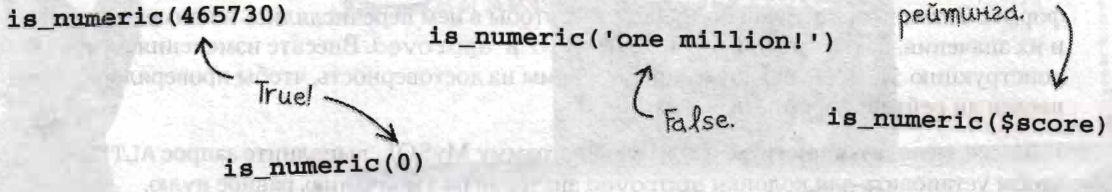
Последняя ступень в минимизации риска атаки «Внедрение SQL-кода» включает проверку данных, введенных в форму сценария «Добавление рейтинга», на достоверность. Прежде чем проверять, соответствует ли тип или размер файла изображения, подтверждающего рейтинг, значениям, определенным приложением, необходимо проверить, не являются ли пустыми значения трех полей ввода.

```
if (!empty($name) && !empty($score) && !empty($screenshot)) {

}
```

Эта управляющая конструкция if проверяет, все ли поля формы заполнены.

Этот код выглядит достаточно неплохо, но, когда речь идет о вопросах защиты приложения, нужно руководствоваться еще и, можно сказать, чувством долга. Раз поле ввода рейтинга ожидает целочисленной величины, имеет смысл убедиться в том, что значение этого поля не только не пустое, но и является целым числом. PHP-функция `is_numeric()` производит такую проверку, возвращая `true`, если переменная, переданная ей в качестве аргумента, является целым числом, и `false`, если это переменная любого другого типа. Последовательное выполнение таких простых действий, как проверка, введено ли в поле формы целое число, когда вы ожидаете ввода данных именно такого типа, делает ваше приложение максимально защищенным от хакерских атак.

True или false зависит от того, ввел пользователь целое число или нет в поле ввода рейтинга.



Везде, где можно, настаивайте, чтобы данные были в том формате, в котором вы их запрашиваете.

УПРАЖНЕНИЕ

Перепишите форму в сценарии «Добавление рейтинга», добавив управляющую конструкцию `if` с использованием функции `is_numeric()` так, чтобы разрешалось вводить рейтинг только в виде целого числа.

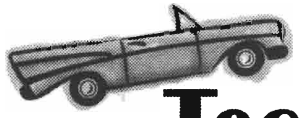
.....

.....

.....

Перепишите форму в сценарии «Добавление рейтинга», добавив управляющую конструкцию `if` с использованием функции `is_numeric()` так, чтобы разрешалось вводить рейтинг только в виде целого числа.

```
if (!empty($name) && is_numeric($score) && !empty($screenshot)) {
    ...
}
```



Тест-драйв

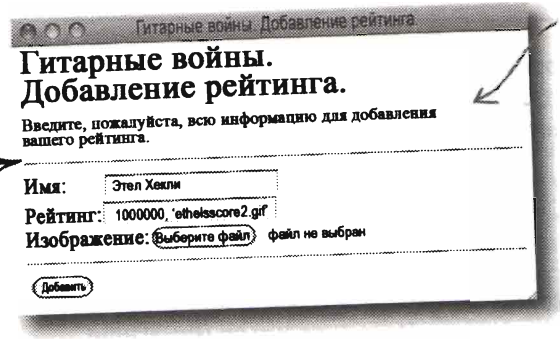
Ужесточите обработку данных формы в сценарии «Добавление рейтинга».

Организуйте процесс присвоения значений переменным сценария «Добавление рейтинга» так, чтобы, используя функции `trim()` и `mysqli_real_escape_string()`, «подчистить» данные формы. Затем измените запрос `INSERT` так, чтобы в нем перечислялись и наименования колонок, и их значения, исключив при этом колонки `id` и `approved`. Внесите изменения в управляющую конструкцию `if`, проверяющую данные формы на достоверность, чтобы проверялось также и то, введен ли рейтинг в формате целого числа.

Наконец, используя инструментальную программу MySQL, выполните запрос `ALTER TABLE` так, чтобы установить для колонки `approved` значение по умолчанию, равное нулю.

Загрузите новый сценарий «Добавление рейтинга» на свой веб-сервер, перейдите к нему в браузере и попробуйте опять провести атаку «Внедрение SQL-кода».

Теперь поле ввода рейтинга принимает только целые числа и ничего более.

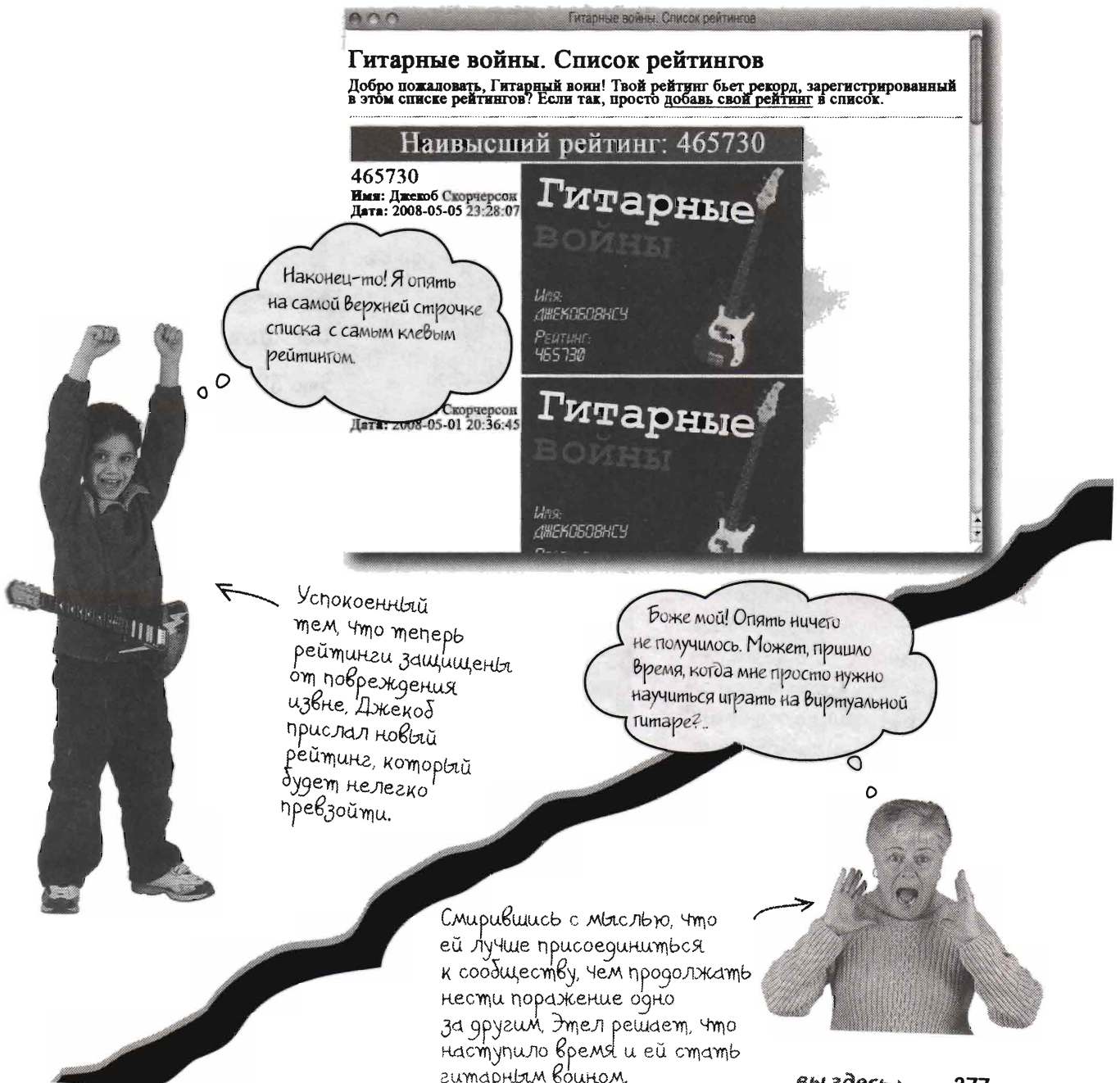


Конечно, это сообщение об ошибке могло бы быть поконкретнее, но в таком виде оно выводится без подключения в сценарий дополнительной логики.

Проверка данных формы на достоверность не ограничивается рамками базы данных. Глава 10 рассматривает это значительно подробнее...

Пожар потушен!

Похоже, желание Этел вредить рейтингам «Гитарных войн» было в конце концов сломлено благодаря улучшениям, внесенным в приложение и сделавшим его невосприимчивым к атакам «Внедрение SQL-кода». Непобедимый чемпион «Гитарных войн» ответил на это регистрацией нового, еще более высокого рейтинга.



Ваш инструментарий PHP и MySQL

В дополнение к повышению уровня приложения «Гитарные войны» вы приобрели некоторое количество новых инструментов и технологий. Давайте вспомним наиболее важные из них.

header()

Встроенная PHP-функция, используемая для отправки HTTP-заголовка с сервера браузеру. Позволяет вам выполнять такие задачи, как переадресация страницы, установка определенного формата содержания или HTTP-аутентификация.

$_SERVER

Среди других данных этот встроенный суперглобальный массив PHP содержит имя и пароль, введенные пользователем, пытающимся открыть страницу, которая защищена с помощью HTTP-аутентификации. Вы можете проверить соответствие этих данных значениям, установленным с целью защиты страницы от несанкционированного доступа.

exit()

Эта встроенная PHP-функция вызывает немедленную остановку выполнения сценария. Как только сценарий достигнет этой функции, выполнение любого PHP-кода и отправка браузеру любого HTML-кода прекращается.

is_numeric()

Встроенная PHP-функция, используемая для проверки, является ли переменная, переданная ей в качестве аргумента, целым числом. Эта функция полезна для проверки, действительно ли в целочисленные поля формы введены целые числа.

DEFAULT *значение*

Это SQL-выражение устанавливает для колонки в таблице значение по умолчанию. При добавлении новой записи без указания конкретного значения для этой колонки ей будет автоматически присвоено значение по умолчанию.

trim(), mysqli_real_escape_string()

Эти две встроенные PHP-функции удобны для обработки данных формы и изоляции проблемных символов, которые могут быть введены пользователями, а также для предотвращения попадания этих символов в SQL-запросы (это может исказить смысл запроса и стать причиной серьезных проблем).

Арбитраж с помощью человека

Все через арбитра! Здесь это означает, что человек часто обеспечивает наилучший рубеж обороны при определении и устранении нежелательного содержания, присланного извне. Автоматические технологии защиты, конечно, остаются очень важными, но они часто не в силах тягаться с живым, дышащим человеком с головой на плечах.

Атака «Внедрение SQL-кода»

Брешь в системе защиты приложения, позволяющая злоумышленнику изменить смысл SQL-запроса, чтобы получить доступ к базе данных. Большинство атак «Внедрение SQL-кода» заключаются в использовании веб-форм для передачи опасных данных непосредственно коду, ответственному за динамическое создание SQL-запросов. Поэтому проверка достоверности данных формы часто решает проблему.

Запрос типа «колонка/значение»

Тип SQL-запроса INSERT, в котором колонки и их значения перечисляются двумя списками в строгом соответствии друг с другом. Это альтернатива тому, чтобы, не перечисляя в запросе имен колонок, полагаться на их порядок, определенный структурой таблицы.

Проверка достоверности данных формы

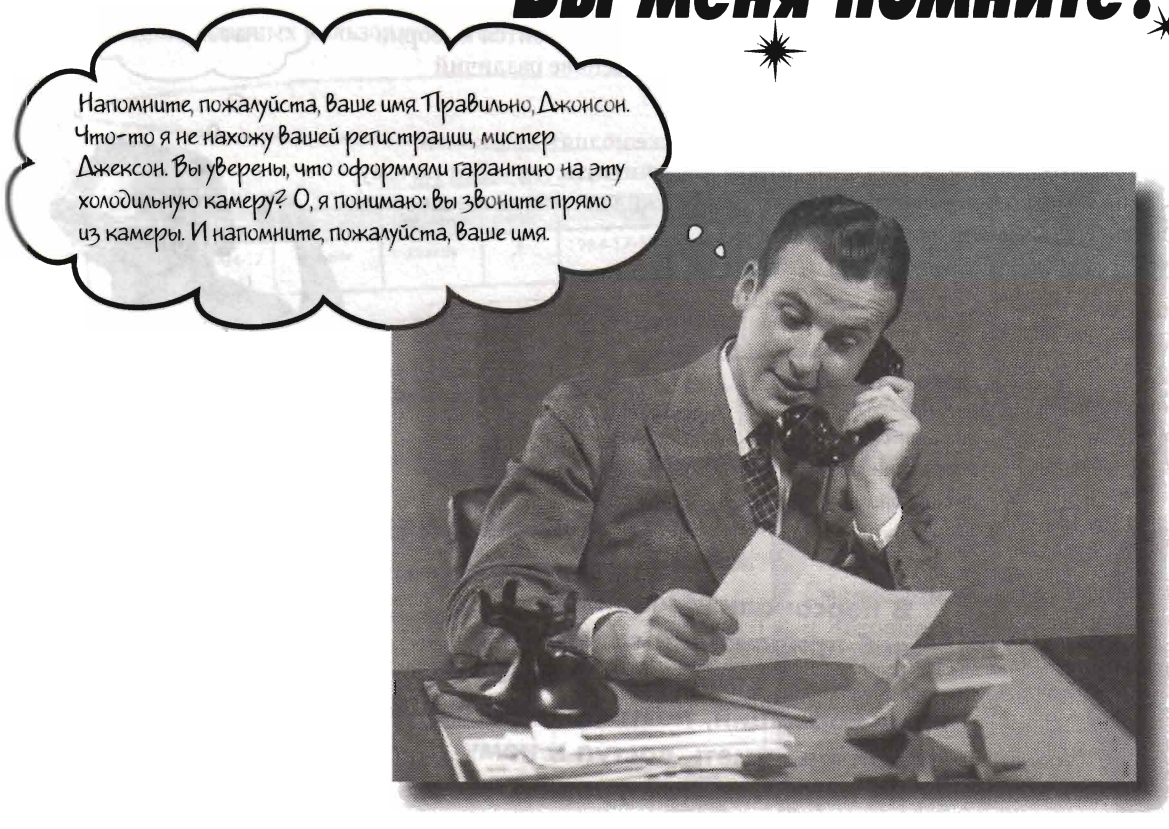
Процесс проверки всех данных, введенных пользователем в форму, чтобы убедиться в том, что они введены в ожидаемом формате. Кроме того, что это облегчает пользователю работу с формой, проверка достоверности данных формы может помочь в повышении уровня защиты веб-приложения, не позволяя пользователю вводить нежелательные данные.

HTTP-аутентификация

Простая технология защиты приложения, основанная на ограничении доступа к веб-странице или сценарию путем использования имени пользователя и пароля. Хотя и не предназначенная для приложений, требующих высокого уровня защиты чувствительных данных, HTTP-аутентификация может быть весьма удобной для быстрого добавления приемлемой для веб-приложений степени защиты.

7 создание персонализированного веб-приложения

Вы меня помните?

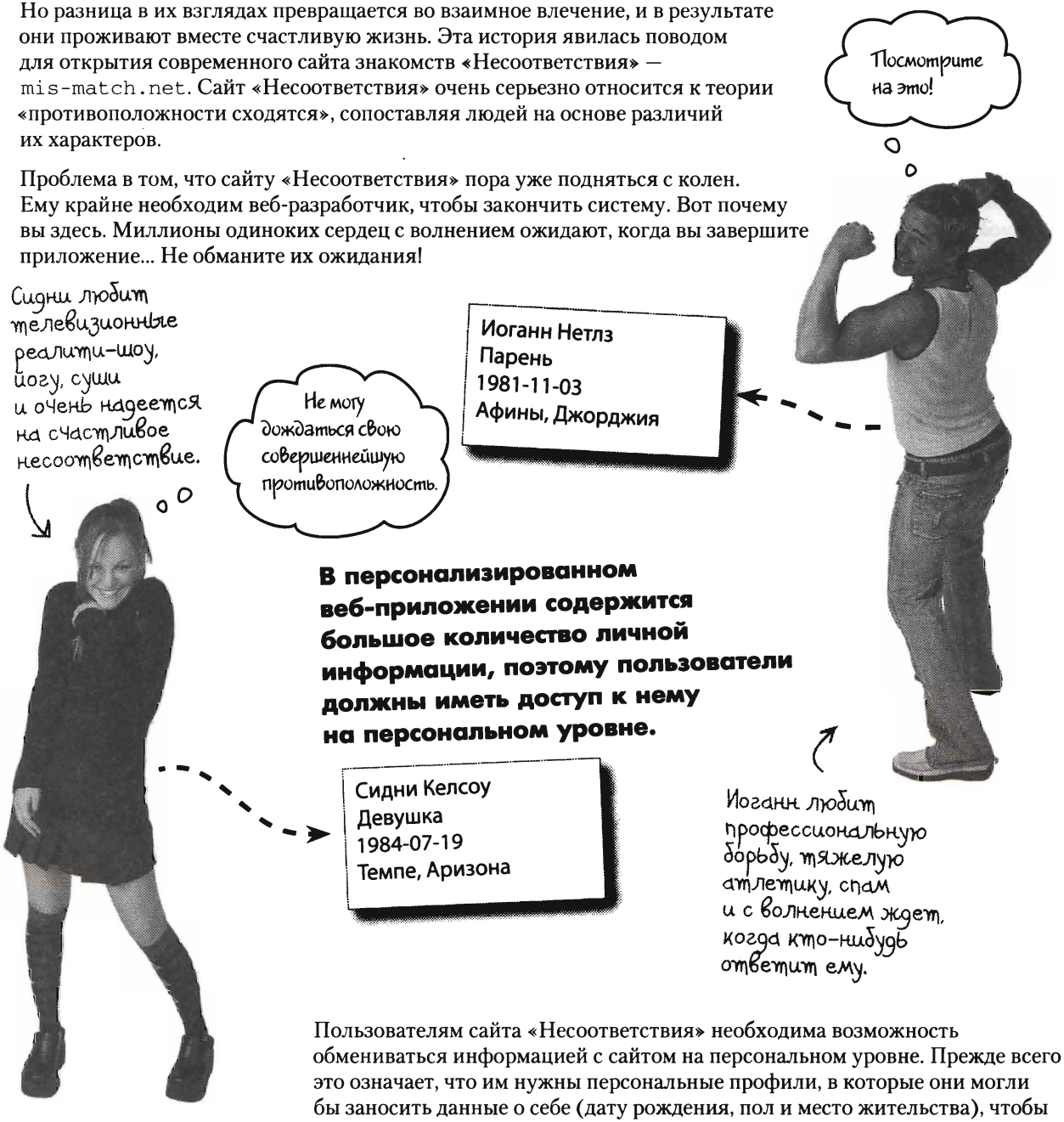


Никому не нравится, когда о нем забывают, особенно это не любят пользователи веб-приложений. Если приложение ориентировано на работу с пользователями как с членами своего рода сообщества и пользователи обмениваются с ним информацией в определенной степени личного характера, такое приложение должно помнить своих пользователей. Вам определенно не понравилось бы представляться своей семье каждый раз, появляясь на пороге дома. Вам нет необходимости делать это, потому что у всех членов вашей семьи есть такая замечательная штука, как **память**. Но веб-приложение не запоминает людей автоматически. Это забота толкового разработчика приложения, использующего доступные средства (может быть, PHP и MySQL?), — **создавать персонализированные веб-приложения, способные запоминать пользователей.**

Говорят, противоположности сходятся

История вековой давности: парень встречает девушку. Девушка считает парня круглым идиотом. Парень считает, что у девушки имеются проблемы. Но разница в их взглядах превращается во взаимное влечение, и в результате они проживают вместе счастливую жизнь. Эта история явилась поводом для открытия современного сайта знакомств «Несоответствия» — `mis-match.net`. Сайт «Несоответствия» очень серьезно относится к теории «противоположности сходятся», сопоставляя людей на основе различий их характеров.

Проблема в том, что сайту «Несоответствия» пора уже подняться с колен. Ему крайне необходим веб-разработчик, чтобы закончить систему. Вот почему вы здесь. Миллионы одиноких сердец с волнением ожидают, когда вы завершите приложение... Не обманите их ожидания!



В персонализированном веб-приложении содержится большое количество личной информации, поэтому пользователи должны иметь доступ к нему на персональном уровне.

Пользователям сайта «Несоответствия» необходима возможность обмениваться информацией с сайтом на персональном уровне. Прежде всего это означает, что им нужны персональные профили, в которые они могли бы заносить данные о себе (дату рождения, пол и место жительства), чтобы обмениваться ими с другими пользователями сайта «Несоответствия».

Все эти несоответствия являются, по существу, личными данными

Итак, задача сайта «Несоответствия» сводится к организации обмена персональными данными. Этот процесс должен происходить внутри сообщества пользователей, каждый из которых может обмениваться информацией с сайтом и управлять своими персональными данными. Таблица `mismatch_user` базы данных `mismatch` используется для хранения персональных данных пользователей сайта «Несоответствия».

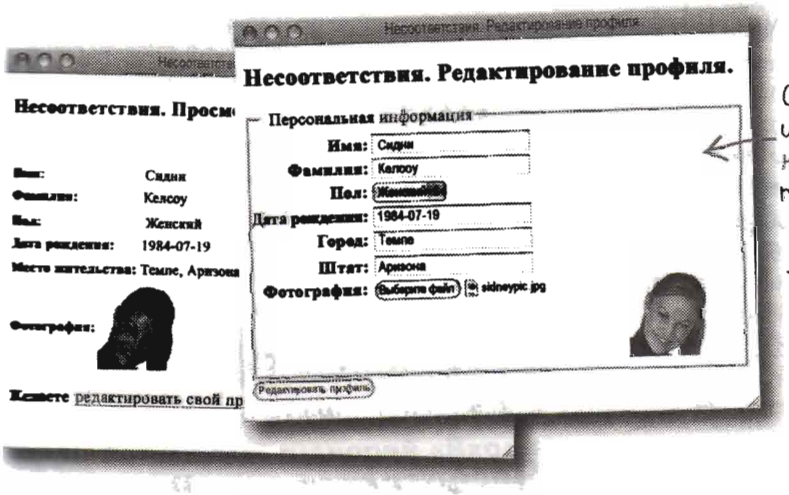
Это база данных mismatch.

В таблице mismatch_user базы данных mismatch хранятся персональные данные пользователей.

mismatch_user

user_id	join_date	first_name	last_name	gender	birthdate	city	state	picture
1	2008-04-17 09:43:11	Сидни	Келсоу	F	1984-17-19	Темпе	Аризона	sidneypic.jpg
...								
11	2008-05-23 12:24:06	Иоган	Нетлз	M	1981-11-03	Афины	Джорджия	johanpic.jpg
...								

В каждой записи таблицы mismatch_user хранятся персональные данные одного пользователя.



Страницам «Просмотр профиля» и «Редактирование профиля» необходимо знать, к профилю какого пользователя получать доступ.

В дополнение к просмотру профиля пользователь сайта «Несоответствия» должен также иметь возможность редактировать свой собственный профиль. Но проблема в том, что приложение должно знать, профиль какого пользователя может редактироваться. Страница «Редактирование профиля» должна как-то определять, какой пользователь запрашивает доступ к ней для редактирования профиля.

Как сделать, чтобы страница «Редактирование профиля» открывалась только для своего пользователя?

Пользователям приложения «Несоответствия» необходимо войти в приложение, прежде чем они получат доступ к профилям

Решение проблемы доступа к персональным данным заключается во входе пользователей в приложение после его открытия. При этом вопрос о предоставляемом уровне доступа будет решаться на основании данных о себе, введенных каждым конкретным пользователем при входе в приложение. Зная, кто есть кто, приложение «Несоответствия» получает возможность предоставлять полный доступ (редактирование) к персональной информации только ее владельцу и ограниченный доступ (просмотр) всем остальным пользователям, также вошедшим в приложение. Итак, пользователь, вошедший в приложение, будет иметь полную возможность редактировать свой профиль, но при этом только просматривать профили всех остальных. Принцип входа пользователей в приложение — это ключевой момент на пути создания персонализированного приложения «Несоответствия».

Принцип входа позволяет веб-приложению взаимодействовать с пользователями на персональном уровне.

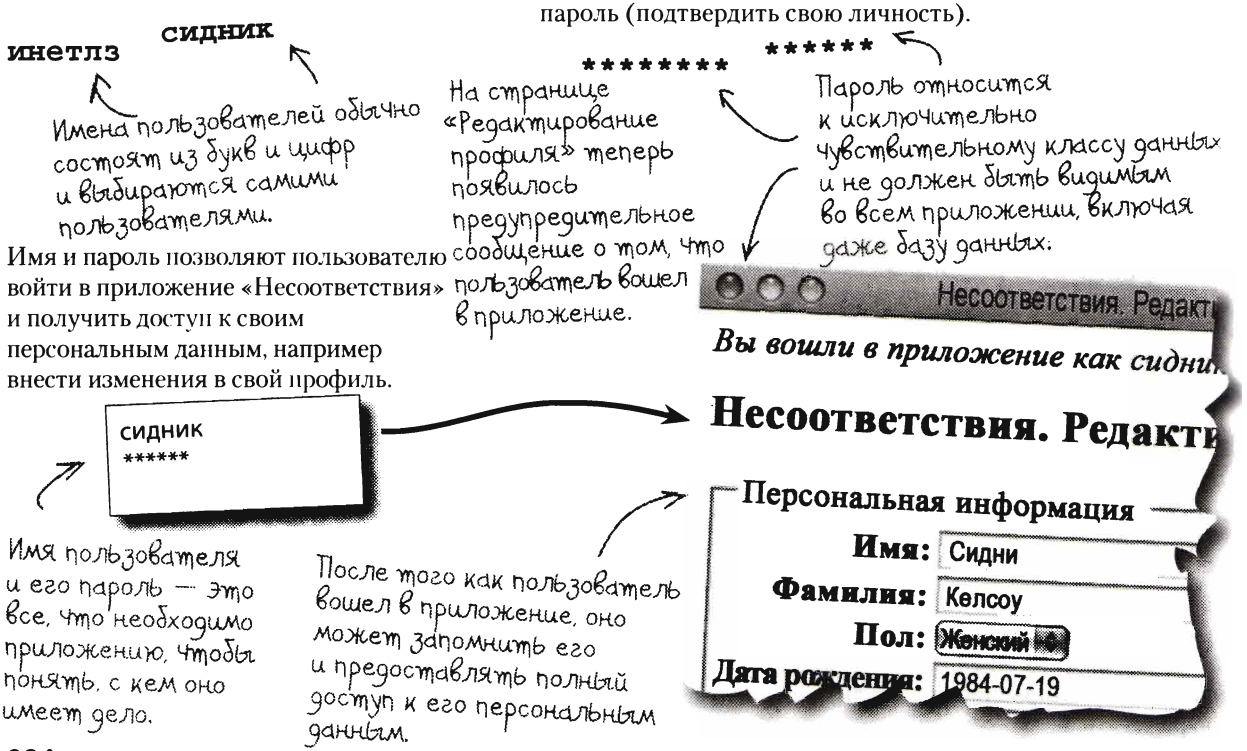
При входе в приложение, чтобы оно могло понять, с кем имеет дело, пользователю обычно достаточно ввести свое имя и пароль.

Имя пользователя

Задача имени заключается в предоставлении пользователю уникального идентификатора, используя который система сможет найти информацию, связанную именно с этим пользователем. Пользователи смогут также обмениваться информацией друг с другом, используя эти имена.

Пароль

Пароль несет ответственность за предоставление определенного уровня секретности при входе пользователя в приложение, что позволяет приложению быть уверенным, что оно имеет дело именно с тем пользователем, за которого тот себя выдает, и защитить его персональные данные. Поэтому при входе в приложение пользователь должен ввести не только свое имя (представиться), но также и свой пароль (подтвердить свою личность).

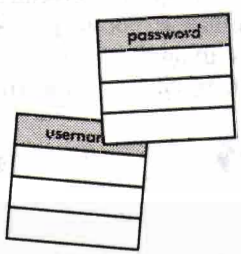


Более детальная разработка плана для входа пользователей в приложение

Добавить в приложение «Несоответствия» поддержку процедуры входа в него пользователей — это не просто трюк, поэтому, перед тем как писать конкретный код и запросы к базам данных, очень важно определить, какие части приложения вовлечены в сферу действия этой процедуры. Мы знаем, что уже существует таблица, в которой содержатся данные о пользователях, поэтому первым шагом будет такое изменение ее структуры, чтобы она могла сохранять также и входные данные этих пользователей. Нам также необходимо предоставить пользователям способ для ввода своих входных данных. Они должны быть как-то интегрированы с другими частями приложения «Несоответствия», чтобы персональные страницы, как , например, «Редактирование профиля», были доступны только после успешного входа в приложение. Ниже приведены этапы разработки кода, обеспечивающего вход пользователей в приложение.

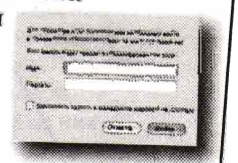
1 **Используйте SQL-запрос ALTER TABLE, чтобы добавить в таблицу колонки username (имя пользователя) и password (пароль).**

Таблице необходимы новые колонки, чтобы сохранять в них входные данные для каждого пользователя. Это колонки `username` и `password`.



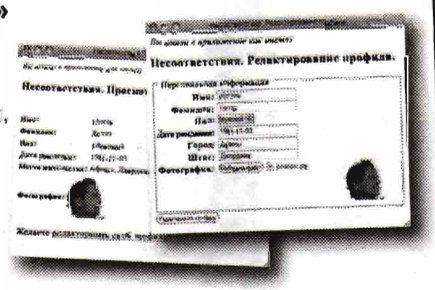
2 **Создайте новый сценарий для запроса имени пользователя и его пароля.**

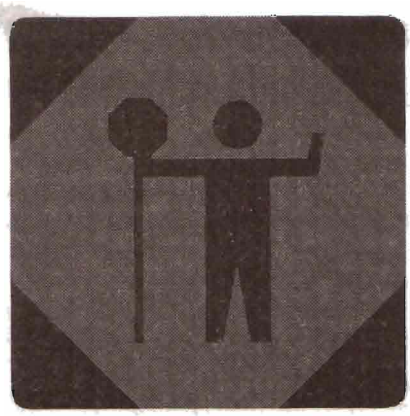
Форма входа в приложение — это то, что полностью защитит страницы с персональной информацией путем запроса действующего имени пользователя и его пароля. Только после правильного ввода этой информации приложение сможет предоставить доступ к персональным данным пользователей. Таким образом, этот сценарий должен ограничить доступ к страницам с персональной информацией, чтобы они не были видны без предварительной процедуры входа в приложение.



3 **Согласуйте сценарий «Вход в приложение» с остальными частями приложения.**

Страницы «Просмотр профиля» и «Редактирование профиля» должны быть доступны только пользователям, прошедшим процедуру входа в приложение. Поэтому, прежде чем предоставить доступ к этим страницам, нам необходимо убедиться, что пользователь, запрашивающий их, вошел в приложение.

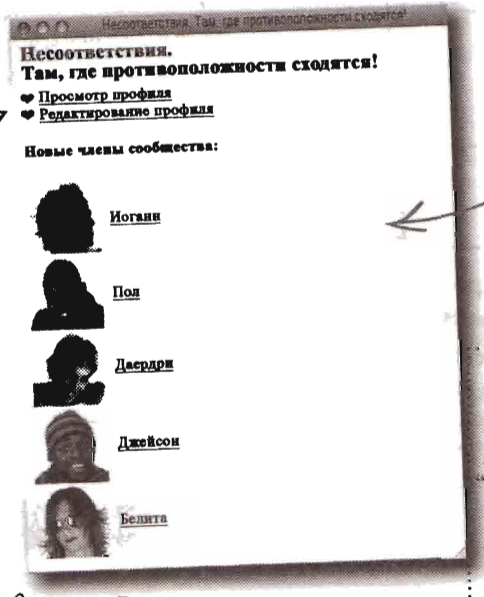




Прежде чем двигаться дальше, найдите время познакомиться с приложением «Несоответствия» поближе, чтобы получить представление о том, как оно работает.

Загрузите код для приложения «Несоответствия» с сайта по адресу `www.headfirstlabs.com/books/hfphp`. Отправьте весь код, кроме файлов с суффиксом `.sql`, на ваш веб-сервер. В файлах с суффиксом `.sql` содержатся SQL-запросы для создания всех таблиц, необходимых приложению «Несоответствия». Убедитесь в том, что вы сделали все запросы, используя вашу инструментальную MySQL-программу, и создали таким образом все таблицы, необходимые для начала работы.

После того как все это сделано, перейдите в своем браузере к странице `index.php` и посмотрите, как работает приложение. Имейте в виду, что страницы «Просмотр профиля» и «Редактирование профиля» первоначально не работают, потому что они полностью зависят от процедуры входа пользователей в приложение, в то время как мы еще только на полпути к ней.



Эти две гиперссылки ведут в персонализированную область приложения.

Главная страница приложения «Несоответствия» дает вам возможность видеть имена и фотографии последних членов сообщества, но не более того без прохождения процедуры входа в приложение.

Загрузите это!

Весь код для приложения «Несоответствия» доступен для загрузки на сайте «Лаборатория "Очертя голову"» по адресу:

`www.headfirstlabs.com/books/hfphp`

Подготовка базы данных к процедуре входа в приложение

Итак, продолжим работу. Таблица `mismatch_user` уже выполняет полезную функцию, сохраняя данные профиля для каждого пользователя, но этого недостаточно для того, чтобы пользователь смог войти в приложение. Если выражаться точнее, в таблице не хватает колонок для сохранения имени пользователя и его пароля.

Таблице mismatch_user необходимы колонки для сохранения имени пользователя и его пароля, чтобы обеспечить процедуру входа в приложение.

mismatch_user

user_id	join_date	first_name	last_name	gender	birthdate	city	state	picture

И имя пользователя, и его пароль записываются в формате простого текста, поэтому для новых колонок `username` и `password`, предназначенных для сохранения имени пользователя и его пароля в таблице `mismatch_user`, можно предусмотреть уже знакомый нам тип данных MySQL `VARCHAR`. Но в отличие от других данных профиля пользователя эти колонки не должны содержать пустые значения (`NULL`).

Колонки username и password содержат данные в формате простого текста и не должны быть пустыми.

password
****
*****
*****
...

username
jnettles
baldpaul
dierdre
...

Не бывает глупых вопросов

В: Почему бы нам не использовать user_id вместо username для уникальной идентификации пользователя?

О: При желании это можно сделать. Фактически назначение `user_id` заключается в том, чтобы предоставить уникальный и эффективный способ идентифицировать запись конкретного пользователя. Но числовые идентификаторы не так легко запомнить, а, кроме того, пользователям просто нравится создавать свои собственные имена для доступа к персонализированным веб-приложениям. Поэтому это, скорее, вопрос простоты и удобства использования — дать возможность Иоганну войти в приложение как «инетлз», а не «11». Мало кто захочет быть низведенным до уровня простого числа!

Время поработать

Мало кто захочет попытаться запомнить пароль, содержащий более 16 символов.

Закончите SQL-запрос для добавления колонок `username` и `password`, расположив их так, как показано ниже. При этом колонка `username` должна содержать до 32 символов, а колонка `password` — до 16 символов, и ни одна из них не должна быть пустой.

mismatch_user

user_id	username	password	join_date	first_name	last_name	gender	birthdate	city	state	picture

Решение задачи

Закончите SQL-запрос для добавления колонок `username` и `password`, расположив их так, как показано ниже. При этом колонка `username` должна содержать до 32 символов, а колонка `password` — до 16 символов, и ни одна из них не должна быть пустой.

SQL-запрос ALTER TABLE используется для добавления новых колонок к существующей таблице.

```
ALTER TABLE mismatch_user ADD username VARCHAR(32) NOT NULL AFTER user_id,
ADD password VARCHAR(16) NOT NULL AFTER username
```

Ключевое слово AFTER определяет место в структуре таблицы, на которое будет помещена новая колонка.

Колонка username уже добавлена, поэтому ссылка на нее здесь вполне допустима.

mismatch_user

user_id	username	password	join_date	first_name	last_name	gender	birthdate	city	state	picture

Место в структуре таблице, на котором находится колонка, особого значения не имеет, но с организационной точки зрения лучше помещать впереди наиболее важные из данных.

СДЕЛАНО

1. **Используйте SQL-запрос ALTER TABLE, чтобы добавить в таблицу колонки username (имя пользователя) и password (пароль).**

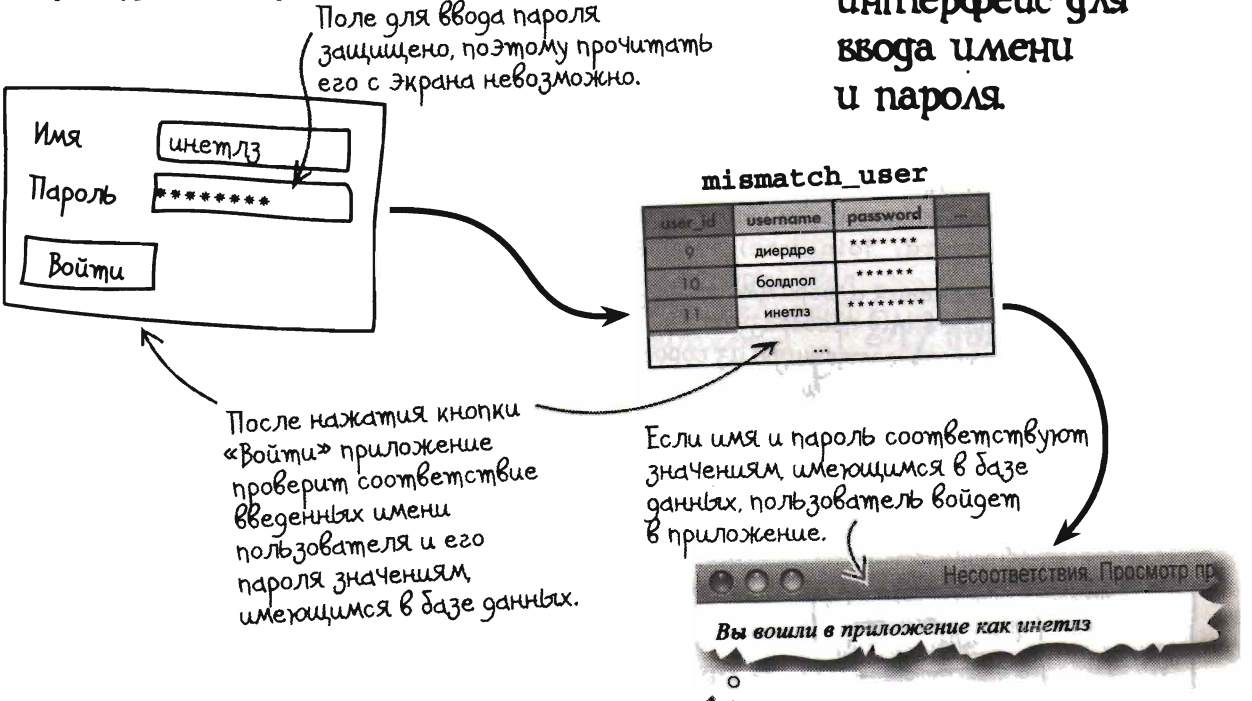
Совершенно правильное замечание... Пароль необходимо зашифровать.

Шифрование в приложении «Несоответствия» при сохранении пароля в базе данных включает процедуру преобразования его в формат, не позволяющий определить его первоначальное значение. Любое приложение с поддержкой процедуры входа в него пользователей должно обязательно предусматривать шифрование паролей, чтобы пользователь чувствовал уверенность, что его пароль держится в секрете в безопасном месте. Доступность пароля даже в базе данных недопустима. Поэтому нам необходим способ шифрования пароля перед тем, как он будет помещен в таблицу `mismatch_user`. Но проблема в том, что шифрование не поможет нам до тех пор, пока у пользователя отсутствует возможность ввести свое имя и пароль для входа в приложение...

Создание интерфейса для входа пользователя в приложение

После того как база данных получила возможность сохранять входные данные пользователя, нам необходимо предоставить возможность пользователю вводить эти данные и входить в приложение. Нужно создать интерфейс, включающий текстовые поля для ввода имени пользователя и пароля, а также кнопку для выполнения самой процедуры входа в приложение.

Для входа пользователя в приложение необходим интерфейс для ввода имени и пароля.



не бывает глупых вопросов

В: Но ведь в базе данных сохраняются не символы звездочек, так?

О: Это верно. Символы звездочек выводятся в поле формы только лишь для обеспечения видимой его защиты, не давая возможности кому-нибудь прочитать пароль, заглядывая пользователю через плечо. При передаче формы на сервер вместе с ней передается сам пароль, а не символы звездочек. Вот почему важно шифровать пароль перед сохранением его в базе данных.

Если вас волнует вопрос о том, как пользователь сможет войти в приложение, если мы еще не назначили ему имя и пароль... не волнуйтесь.

Мы перейдем к вопросу создания для пользователя имени и пароля чуть позднее. Пока важно заложить основу самого процесса входа в приложение, хотя у нас есть еще задачи, которые предстоит решить, прежде чем все это выстроится в единую систему.

Шифрование пароля с помощью функции SHA()

Интерфейс входа пользователей в приложение достаточно прост и понятен, но он не предусматривает шифрования паролей. MySQL предлагает функцию `SHA()`, которая зашифровывает строку текста, используя алгоритм шифрования SHA. Функция, получив в качестве аргумента строку текста любой длины, возвращает строку текста длиной в 40 цифр, выраженных в шестнадцатеричной системе счисления. Эта строка называется хеш-функцией, или дайджестом исходной строки. В данном случае, если функции `SHA()` передать в качестве аргумента пароль пользователя, она сгенерирует 40-символьный код, уникально представляющий этот пароль.

MySQL-функция SHA() шифрует строку текста, преобразуя ее в уникальный 40-символьный код.

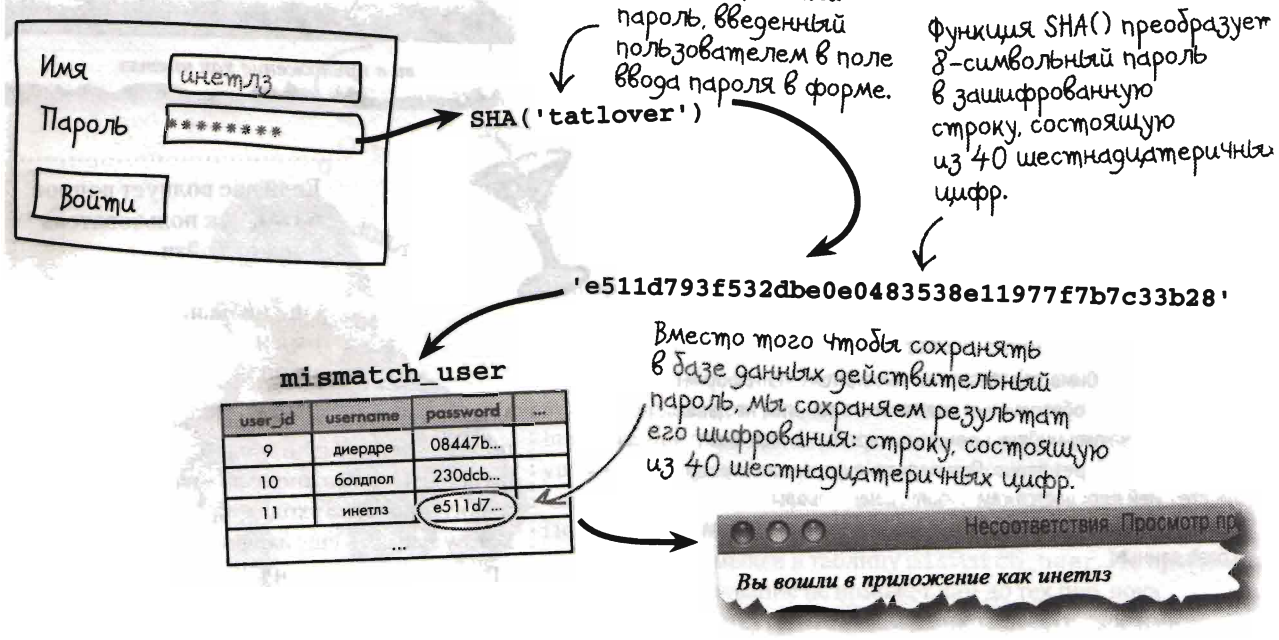
Так как функция `SHA()` является функцией MySQL, а не PHP, вам необходимо вызывать ее в составе SQL-запроса, который используется при добавлении пароля в таблицу или его изменении. Например, этот код добавляет запись для нового пользователя в таблицу `mismatch_user`, зашифровывая при этом пароль с использованием функции `SHA()`:

```
INSERT INTO mismatch_user
  (username, password, join_date) VALUES ('инетлз', SHA('tatlover'), NOW())
```

Путем применения алгоритма шифрования SHA функция SHA() преобразует пароль в строку, состоящую из сорока шестнадцатеричных цифр.

Это действительный пароль, то есть тот, который был введен в поле ввода пароля в форме.

Эта же самая функция SHA() используется на другом конце процесса входа в приложение — при проверке соответствия пароля, введенного пользователем, паролю, находящемуся в базе данных.



Сравнение ~~Дешифрование~~ паролей

Как только вы зашифровали какую-то информацию, у вас совершенно инстинктивно возникает мысль о том, как вы сможете ее дешифровать, когда в этом возникнет необходимость. Но функция `SHA()` выполняет однонаправленное шифрование — без возможности восстановления информации в первоначальном, незашифрованном виде. Это повышает уровень защиты, так как если даже кому-то удалось взломать вашу базу данных и получить доступ к паролям, он не сможет дешифровать их. Но как тогда пользователю войти в приложение, если мы не можем дешифровать его пароль?

Функция SHA() использует алгоритм однонаправленного шифрования. Вы не можете дешифровать зашифрованные данные.

Вам нет необходимости знать пароль пользователя в его исходном, незашифрованном виде; важно только, чтобы пользователь ввел его правильно, потому что функция `SHA()` всегда возвращает один и тот же 40-символьный код при условии, что вы всегда передаете ей в качестве аргумента одну и ту же строку текста. Поэтому вам достаточно просто зашифровать пароль, введенный пользователем в форму, и сравнить полученный 40-символьный код с тем, что имеется у вас в колонке `password` таблицы `mismatch_user` для записи этого пользователя. Это может быть достигнуто выполнением единственного SQL-запроса, который пытается выбрать запись пользователя, основываясь на его пароле.

Это пароль, который ввел пользователь, желающий войти в приложение.

```
SELECT * FROM mismatch_user
  WHERE password = SHA('tatlover')
```

Функция SHA() вызывается для того, чтобы зашифровать пароль и передать его в выражение WHERE в виде 40-символьного кода.

В результате выполнения этого запроса из таблицы `mismatch_user` выбираются все записи, для которых значение колонки `password` соответствует паролю, введенному пользователем в зашифрованном виде, хеш-функции строки `tatlover` в данном случае. Так как мы сравниваем зашифрованные версии пароля, совершенно необязательно знать его оригинальное незашифрованное значение. Запрос, в результате выполнения которого мы находим запись пользователя, пытающегося войти в приложение, использует функцию `SHA()`, но нам также понадобится выбирать нужную нам запись и с использованием идентификатора пользователя, что мы скоро увидим.

Добавление места для зашифрованного пароля

Шифрование паролей с помощью функции `SHA()` влечет за собой проблему для приложения «Несоответствия», связанную с тем, что длина зашифрованного пароля в 40 символов превышает максимальную длину строк, сохраняемых в колонке `password` (16 символов). Для того чтобы расширить колонку в пределах, позволяющих сохранять пароль в зашифрованном виде, мы можем использовать SQL-запрос `ALTER TABLE`.

```
ALTER TABLE mismatch_user
  CHANGE password password VARCHAR(40) NOT NULL
```

Размер колонки password изменен до 40, что позволит сохранять пароль в зашифрованном виде.

Не бывает глупых вопросов

В: Откуда происходит название функции SHA()?

О: `SHA` — это сокращение от Secure Hash Algorithm (алгоритм криптографического хеширования). «Хеш» — это термин, используемый в программировании и означающий (в узком смысле) строку фиксированной длины, однозначно представляющую строку текста произвольной длины. В случае функции `SHA()`, использующей алгоритм `SHA`, хеш — это строка, состоящая из 40 цифр, выраженных в шестнадцатеричной системе счисления, и однозначно определяющая строку, переданную ей в качестве аргумента (пароль).

В: Существуют ли другие способы шифрования паролей?

О: Да. MySQL предлагает другую, подобную `SHA()` функцию, называемую `MD5()`, которая использует похожий способ шифрования. Но алгоритм `SHA` считается немного более защищенным, поэтому лучше использовать функцию `SHA()`. PHP также предлагает эквивалентные функции (`sha1()` и `md5()`) на тот случай, если вам понадобится выполнять операции шифрования в PHP-коде, а не в SQL-запросе.

Тест-драйв

Добавьте колонки username и password в таблицу mismatch_user и проведите испытания.

Используя инструментальную программу MySQL, сделайте запрос, в результате выполнения которого в таблице `mismatch_user` появятся новые колонки `username` и `password`.

```
ALTER TABLE mismatch_user ADD username VARCHAR(32) NOT NULL AFTER user_id,
  ADD password VARCHAR(16) NOT NULL AFTER username
```

Но наша колонка `password` в действительности должна допускать сохранение в ней зашифрованных строк длиной 40 символов, поэтому выполните запрос `ALTER TABLE` еще раз, чтобы добавить места для сохранения зашифрованных паролей.

```
ALTER TABLE mismatch_user
  CHANGE password password VARCHAR(40) NOT NULL
```

Не забудьте зашифровать пароль, вызвав функцию SHA().

Теперь, для того чтобы проверить новые колонки, давайте выполним запрос `INSERT` для нового пользователя.

```
INSERT INTO mismatch_user
  (username, password, join_date) VALUES ('джими', SHA('heyjoe'), NOW())
```

Для более тщательной проверки, действительно ли пароль сохранен в базе данных в зашифрованном виде, выполните запрос `SELECT` для нового пользователя.

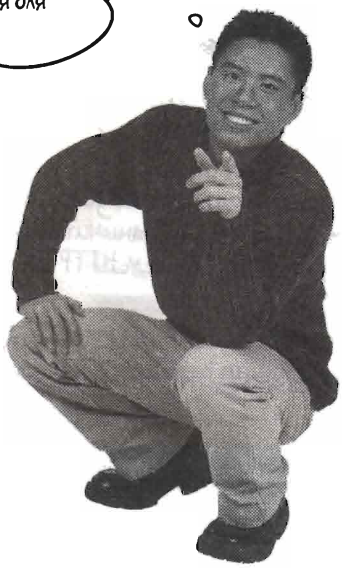
```
SELECT password FROM mismatch_user WHERE username = 'джими'
```

И, наконец, вы можете сымитировать проверку входа пользователя в приложение путем выполнения запроса `SELECT` для имени пользователя и использования функции `SHA()` с паролем в качестве аргумента в условном выражении `WHERE`.

```
SELECT username FROM mismatch_user WHERE password = SHA('heyjoe')
```

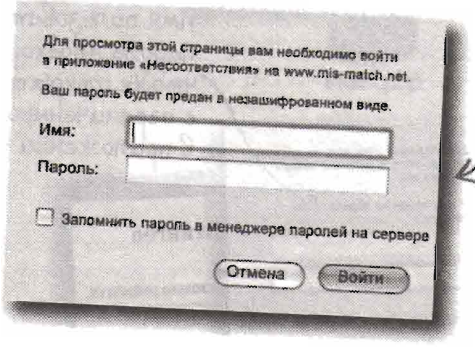
Для успешного входа пользователя в приложение эт должен быть тот же пароль, которы использовался п добавлении данн записи.

Итак, теперь пароль зашифрован, но нам необходимо еще создать форму входа в приложение. Можем ли мы просто использовать HTTP-аутентификацию, ведь она также требует ввода имени пользователя и его пароля для доступа к защищенным страницам?



Конечно! HTTP-аутентификация определенно сможет сыграть роль простой системы входа пользователя в приложение.

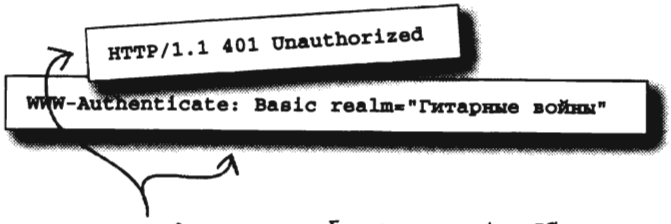
Если вы помните, в приложении «Гитарные войны», описанном в предыдущих главах, HTTP-аутентификация использовалась для ограничения доступа к определенным страницам приложения путем запроса имени пользователя и его пароля. Примерно такие же функции необходимы приложению «Несоответствия» за исключением того, что здесь у нас имеется множество пар «имя пользователя/его пароль», в то время как для всего приложения «Гитарные войны» была только одна такая пара. Пользователи приложения «Несоответствия» могут использовать то же самое окно HTTP-аутентификации, только каждый из них должен будет вводить свои собственные имя и пароль.



Стандартное окно HTTP-аутентификации, внешнее оформление которого зависит от браузера, может быть использовано в качестве простого интерфейса входа пользователей в приложение.

Аутентификация пользователей с помощью HTTP

Как было показано в приложении «Гитарные войны», чтобы ограничить доступ к странице с использованием окна диалога HTTP-аутентификации, необходимо отправить два заголовка. В результате их обработки браузер запрашивал имя пользователя и его пароль, чтобы решить вопрос о предоставлении доступа к странице «Администрирование рейтингов» приложения «Гитарные войны».



Эти два заголовка должны быть посланы для того, чтобы ограничить доступ к странице с использованием HTTP-аутентификации.

Отправка заголовков для HTTP-аутентификации требует двух строк кода — вызовов функции `header()` для каждого из отправляемых заголовков.

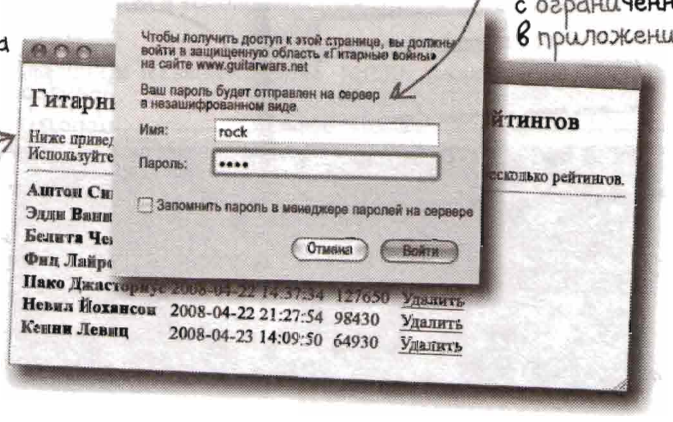
```
header('HTTP/1.1 401 Unauthorized');
header('WWW-Authenticate: Basic realm="Несоответствия"');
```

Для HTTP-аутентификации требуется отправить два заголовка.

Это защищенная область, которая распространяется на все приложение.

Пока пользователь не введет правильного имени и пароля, он не получит доступа к этой странице.

Имя пользователя и его пароль необходимы для того, чтобы открыть страницу с ограниченным доступом в приложении «Гитарные войны».

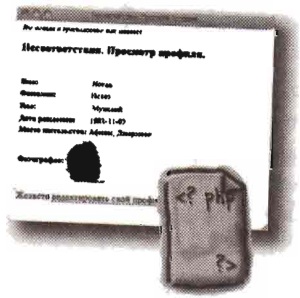


УПРАЖНЕНИЕ

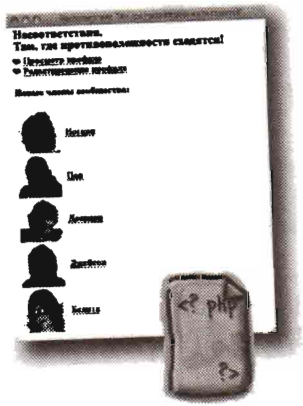
Определите части приложения «Несоответствия», на которые оказывает влияние сценарий «Вход в приложение» (`login.php`), и как они должны использовать HTTP-аутентификацию для контроля доступа. Прокомментируйте это.

Это сценарий «Вход в приложение».

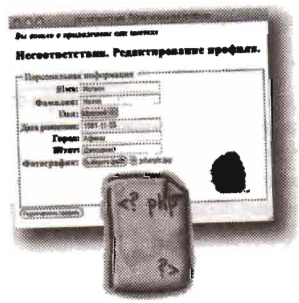
login.php



viewprofile.php

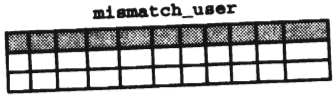


index.php



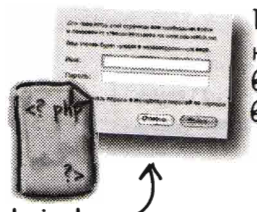
editprofile.php

mismatch_user



РЕШЕНИЕ К УПРАЖНЕНИЮ

Определите части приложения «Несоответствия», на которые оказывает влияние сценарий «Вход в приложение» (`login.php`), и как они должны использовать HTTP-аутентификацию для контроля доступа. Прокомментируйте это.

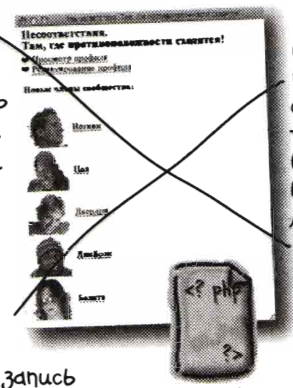


login.php

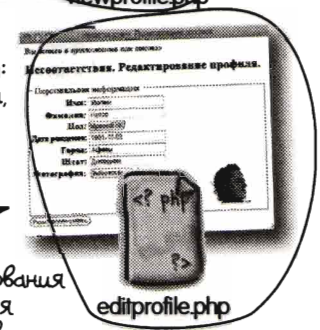
Главная страница не играет никакой роли во входе пользователей в приложение, потому что она должна всегда оставаться доступной для всех.

viewprofile.php

Страницы просмотра и редактирования профиля имеют ограниченный доступ: только пользователи, вошедшие в приложение, могут их открывать.

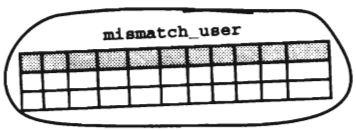


index.php



editprofile.php

Когда пользователь войдет в приложение, его имя и пароль проверяются на соответствие значениям, которые сохранены в базе данных.



Если не будет найдена запись с соответствующей парой «имя пользователя/его пароль», сценарий входа в приложение выводит сообщение об ошибке и отказывает в доступе.

Страница редактирования профиля полагается на сценарий входа в приложение не только из-за обеспечения ограниченного доступа. Ей также необходимо имя пользователя, чтобы знать, чей профиль будет редактироваться.

Не бывает глупых вопросов

В: Почему главная страница не включается в перечень страниц с ограниченным доступом, требующих входа пользователя в приложение?

О: Потому что главная страница — это самое первое место, куда пользователь попадает при открытии сайта. Важно дать пользователю возможность оглядеться, прежде чем требовать входа в приложение. Поэтому главная страница выступает одновременно и как зазывала, и как отправная точка: зазывала — для тех, кто впервые попал на сайт, отправная точка — для членов сообщества, которые должны войти в приложение, чтобы следовать далее.

В: Может ли пользователь, вошедший в приложение, просматривать чужой профиль?

О: Да. Основная идея здесь в том, что профили доступны для просмотра всем пользователям, вошедшим в приложение, но остаются недоступными гостям сайта. Иначе говоря, вы, будучи членом сообщества «Несоответствия» (имея в нем учетную запись), должны войти в него, чтобы просматривать профили всех остальных его членов.

В: Как шифрование паролей сказывается на HTTP-аутентификации?

О: Здесь имеются две отличные друг от друга проблемы: передача пароля и его хранение. Функция MySQL `SHA()` решает проблему безопасного сохранения пароля в базе данных в зашифрованном виде. База данных не имеет ни малейшего понятия о том, как вы перед этим передавали пароль, поэтому данная форма шифрования не оказывает никакого влияния на HTTP-аутентификацию.

Тем не менее очень важно обеспечить шифрование пароля и при передаче его от браузера к серверу вместе с остальными данными окна HTTP-аутентификации. Этот вид шифрования не рассматривается в данной главе и, вообще говоря, необходим только в приложениях, имеющих дело с весьма чувствительными данными.

Вход пользователей в приложение с использованием HTTP-аутентификации

Сценарий аутентификации (`login.php`) несет ответственность за запрос имени пользователя и его пароля с использованием заголовков HTTP-аутентификации, извлечение имени и пароля из суперглобального массива `$_SERVER` и проверку их соответствия данным, имеющимся в таблице `mismatch_user`, перед тем как предоставить доступ к защищенной странице.

```
<?php
   require_once('connectvars.php');

   if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW'])) {
      // Имя пользователя и/или его пароль не были введены,
      // поэтому отправляются заголовки аутентификации.
      header('HTTP/1.1 401 Unauthorized');
      header('WWW-Authenticate: Basic realm="Несоответствия"');
      exit('<h3>Несоответствия</h3>Простите, вы должны ввести ваше имя и пароль для того, ' .
           'чтобы войти в приложение и получить доступ к этой странице.');
   }

   // Соединение с базой данных
   $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

   // Получение введенных пользователем данных для аутентификации
   $user_username = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_USER']));
   $user_password = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_PW']));

   // Поиск имени пользователя и его пароля в базе данных
   $query = "SELECT user_id, username FROM mismatch_user " .
            "WHERE username = '$user_username' AND password = SHA('$user_password')";
   $data = mysqli_query($dbc, $query);

   if (mysqli_num_rows($data) == 1) {
      // Процедура входа прошла нормально, присваиваем переменным значения
      // идентификатора пользователя и его пароля
      $row = mysqli_fetch_array($data);
      $user_id = $row['user_id'];
      $username = $row['username'];
   }
   else {
      // Имя пользователя и/или его пароль введены неверно,
      // поэтому отправляются заголовки аутентификации.
      header('HTTP/1.1 401 Unauthorized');
      header('WWW-Authenticate: Basic realm="Несоответствия"');
      exit('<h3>Несоответствия</h3>Простите, вы должны ввести ваше имя и пароль для того, ' .
           'чтобы войти в приложение и получить доступ к этой странице.');
   }

   // Подтверждение успешного входа в приложение
   echo('<p class="login">Вы вошли в приложение как ' . $username . '.</p>');
?>
```

Если имя пользователя и/или пароль введены неправильно, отправляются заголовки аутентификации для повторного их запроса.

Получение имени пользователя и его пароля.

Выполнение запроса для получения записи пользователя с данным именем и паролем, если таковая имеется.

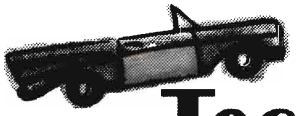
Если запись найдена, то вход в приложение прошел успешно и мы можем присваивать значения переменным `$user_id` и `$username`.

Если запись с таким именем пользователя и паролем отсутствует в базе данных, отправляются заголовки аутентификации для повторного их запроса.

К этому моменту все в порядке, поэтому выводится подтверждение успешного входа в приложение.

СДЕЛАНО

2 Создайте новый сценарий для запроса имени пользователя и его пароля.



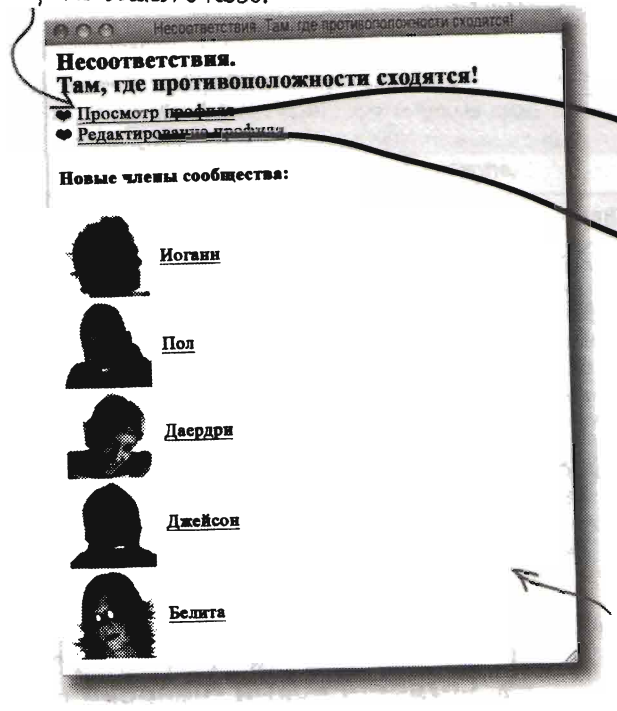
Тест-драйв

Создайте новый сценарий «Вход в приложение» и включите его в сценарии «Просмотр профиля» и «Редактирование профиля».

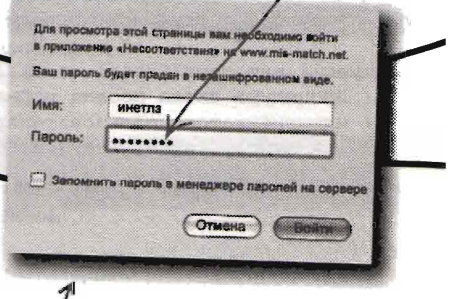
Создайте новый текстовый файл с именем `login.php` и введите в него код процедуры входа в приложение (или загрузите его с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Затем добавьте в начало сценариев `viewprofile.php` и `editprofile.php` PHP-код, включающий новый сценарий входа в приложение.

Загрузите все сценарии на ваш веб-сервер и откройте главную страницу приложения «Несоответствия» в браузере. Щелкните кнопкой мыши по гиперссылке «Просмотр профиля» или «Редактирование профиля», чтобы войти в приложение и получить доступ к персонализированным страницам. Конечно, это будет работать только при условии, что вы уже добавили запись с соответствующим именем и паролем в таблицу базы данных.

Эти две гиперссылки ведут на защищенные страницы, которые вызывают сценарий «Вход в приложение», если пользователь еще не вошел в него.



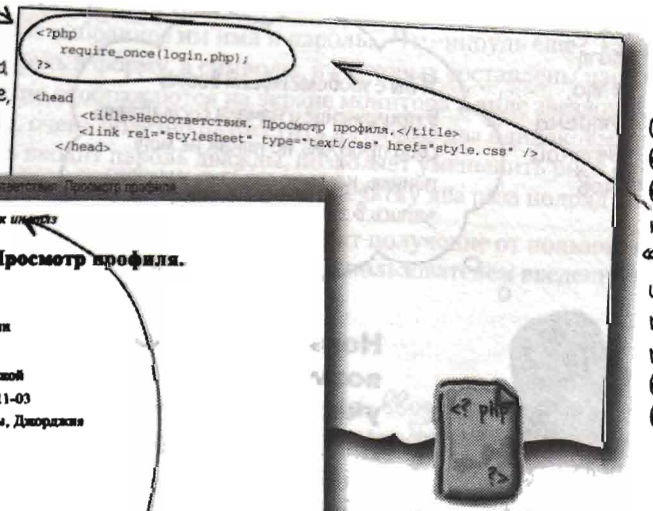
Этот пароль зашифровывается с помощью функции SHA() и сравнивается с зашифрованным паролем в базе данных, чтобы решить, разрешен ли допуск.



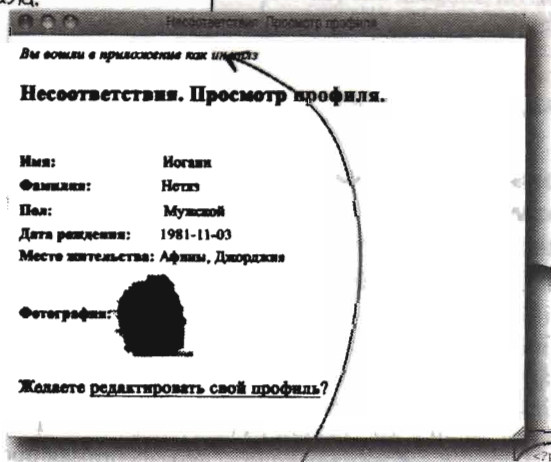
Сценарий «Вход в приложение» использует HTTP-аутентификацию для предотвращения несанкционированного доступа к страницам «Просмотр профиля» и «Редактирование профиля».

Главная страница не защищена сценарием «Вход в приложение» и служит отправной точкой для продвижения далее по приложению.

Любая страница приложения «Несоответствия», которой требуется поддержка процедуры входа пользователей в приложение, должна добавить сценарий login.php в самом начале своего файла.

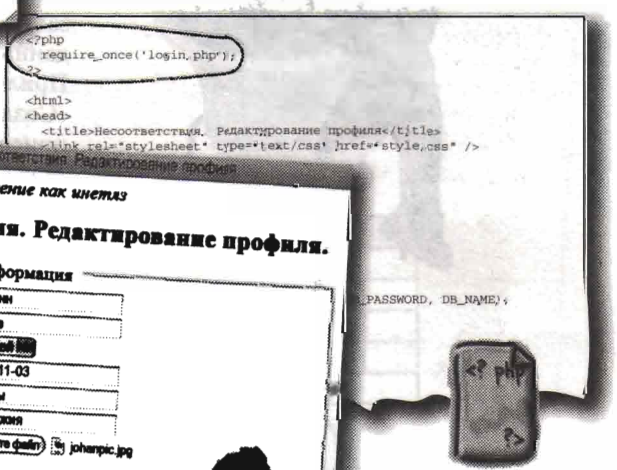


Сценарий «Вход в приложение» включен в самое начало сценариев «Просмотр профиля» и «Редактирование профиля», чтобы приводить в исполнение вход пользователей в приложение.



viewprofile.php

Обе страницы извещают пользователя об успешном входе в приложение, осуществленном сценарием «Вход в приложение».



editprofile.php

Если введены правильные имя и пароль, пользователь входит в приложение и загружается остальная часть страницы.

Каждый пользователь получает доступ к своим личным «несоответствующим» данным, которые он может редактировать.

СДЕЛАНО

3 **Согласуйте сценарии «Вход в приложение» с остальными частями приложения.**

Руби любит фильмы ужасов, кубик Рубика, острые блюда, но в настоящий момент ненавидит сайт «Несоответствия» за то, что в нем не предусмотрена процедура создания учетной записи для новых членов сообщества.

Я бы с удовольствием вошла в приложение и начала составлять свой профиль, но никак не могу понять, как мне создать учетную запись в этом приложении!

Новые пользователи должны иметь возможность создавать в приложении учетные записи для себя.

Новый сценарий «Вход в приложение» приложения «Несоответствия», используя HTTP-аутентификацию, выполняет полезную функцию по входу пользователей в приложение. Проблема заключается в том, что пользователи не имеют возможности создавать учетные записи. Если у вас еще нет имени и пароля в базе данных, войти в приложение — неразрешимая проблема. Приложению «Несоответствия» необходима форма «Создание учетной записи», которая позволяла бы присоединяться к сообществу, добавляя в базу данных новое имя и пароль.

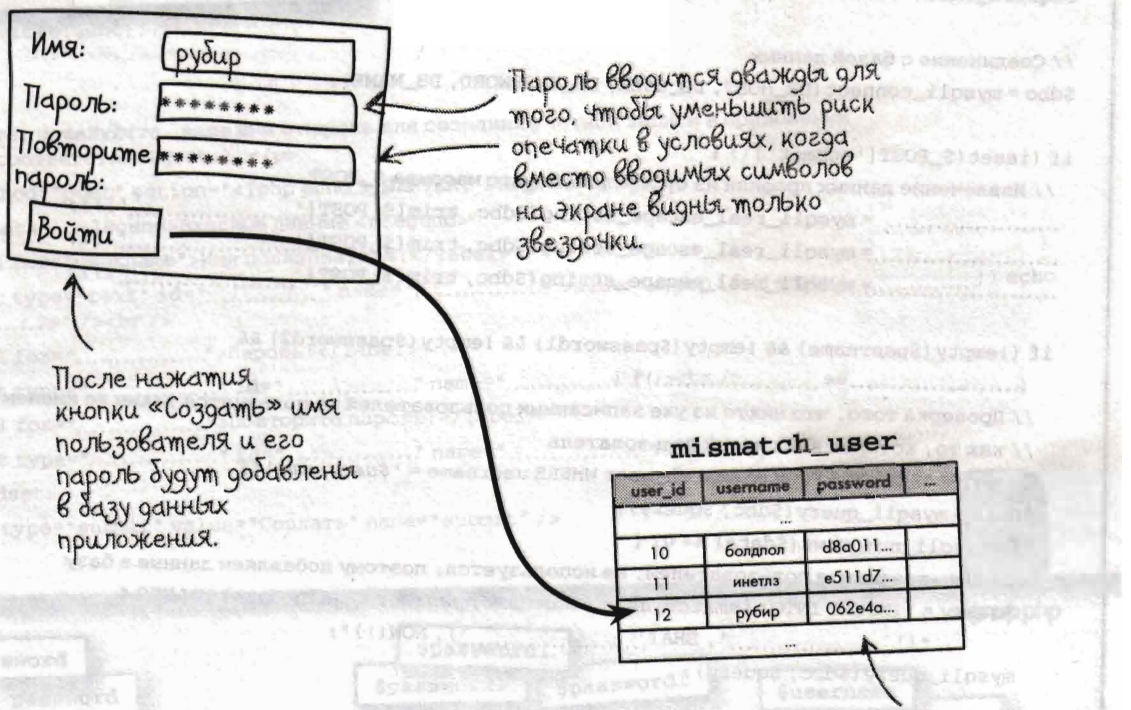
Имя пользователя?

Пароль?

Форма для создания учетных записей для новых пользователей

Как должна выглядеть форма «Создание учетной записи»? Мы знаем, что она должна позволить пользователю ввести выбранное им имя и пароль... Что-нибудь еще? Так как пользователь заносит свой пароль в форму, а символы, из которых составлены пароли, при занесении их форму обычно отображаются на экране монитора в виде звездочек из соображений безопасности, очень полезно разместить на форме два одинаковых поля ввода паролей. То, что пользователь вводит пароль дважды, позволяет уменьшить риск опечатки (маловероятно, что пользователь допустит одну и ту же опечатку два раза подряд).

Итак, в задачу страницы «Создание учетной записи» входят получение от пользователя его имени и пароля, проверка, не используется ли уже другим пользователем введенное имя, и добавление новой записи в таблицу `mismatch_user`.

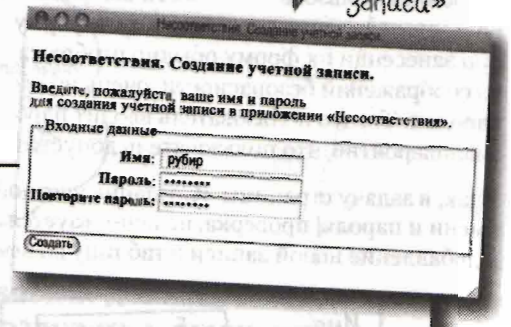


Существует потенциальная проблема, заключающаяся в том, что новый пользователь попытается создать учетную запись с именем уже существующего пользователя. Сценарий должен быть достаточно интеллектуальным, чтобы перехватить такую попытку и предложить пользователю выбрать другое имя. Таким образом, задачи, решаемые страницей «Создание учетной записи», сводятся к тому, чтобы получить от пользователя его имя и пароль, убедиться, что такое имя не используется кем-нибудь другим, и затем добавить запись с данными пользователя в таблицу `mismatch_user`.

Магниты PHP и MySQL

В сценарии «Создание учетной записи» приложения «Несоответствия» используется своя собственная форма для предоставления пользователю возможности ввести выбранное им имя и пароль. Проблема в том, что код сценария неполон. Используя магниты, расположенные ниже, завершите код сценария так, чтобы новый пользователь мог создать учетную запись и присоединиться к сообществу «Несоответствия».

Это форма «Создание учетной записи»

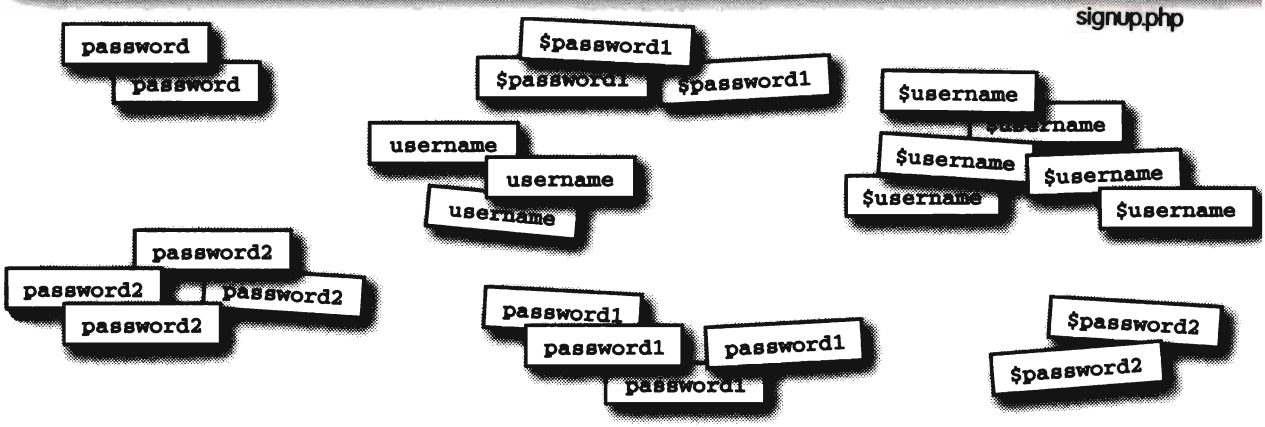



```
<?php
  require_once('appvars.php');
  require_once('connectvars.php');

  // Соединение с базой данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  if (isset($_POST['submit'])) {
    // Извлечение данных профиля из суперглобального массива $_POST
    ..... = mysqli_real_escape_string($dbc, trim($_POST['.....']));
    .....= mysqli_real_escape_string($dbc, trim($_POST['.....']));
    .....= mysqli_real_escape_string($dbc, trim($_POST['.....']));

    if (!empty($username) && !empty($password1) && !empty($password2) &&
      (.....==.....)) {
      // Проверка того, что никто из уже записанных пользователей не пользуется таким же именем,
      // как то, которое ввел новый пользователь
      $query = "SELECT * FROM mismatch_user WHERE username = '$username'";
      $data = mysqli_query($dbc, $query);
      if (mysqli_num_rows($data) == 0) {
        // Имя, введенное пользователем, не используется, поэтому добавляем данные в базу
        $query = "INSERT INTO mismatch_user (username, password, join_date) VALUES " .
            "('.....', SHA('.....'), NOW())";
        mysqli_query($dbc, $query);

        // Вывод подтверждения пользователю
        echo '<p>Ваша новая учетная запись успешно создана. Вы можете войти в приложение и ' .
            '<a href="editprofile.php">отредактировать свой профиль</a>.</p>';

        mysqli_close($dbc);
        exit();
      }
```

Не забывайте, что если в вашей строке, ограниченной одинарными кавычками, появится апостроф, вы должны замаскировать его с помощью наклонной черты (\').

```
else {
      // Учетная запись с таким именем уже существует в базе данных, поэтому выводится сообщение об
ошибке
      echo '<p class="error"> Учетная запись с таким именем уже существует. Введите, пожалуйста, другое
имя.</p>';
      .....= "";
    }
  }
  else {
    echo '<p class="error">Вы должны ввести все данные для создания учетной записи, в том числе пароль –
дважды.</p>';
  }
}

mysqli_close($dbc);
?>

<p>Введите, пожалуйста, ваше имя и пароль для создания учетной записи в приложении
&amp;quot;Несоответствия&amp;quot;\.</p>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
  <fieldset>    <legend>Входные данные </legend>
    <label for="username">Имя пользователя:</label>
    <input type="text" id="....." name="....." value="<?php if (!empty(.....)) echo
 .....;.?>" /><br />
    <label for=".....">Пароль:</label>
    <input type="....." id="....." name="....." /><br />
    <label for=".....">Повторите пароль:</label>
    <input type="....." id="....." name="....." /><br />
  </fieldset>
  <input type="submit" value="Создать" name="submit" />
</form>
```

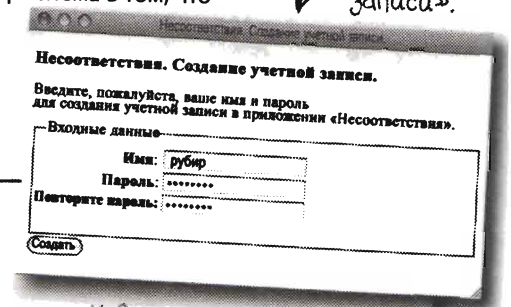
signup.php



Магниты PHP и MySQL

В сценарии «Создание учетной записи» приложения «Несоответствия» используется своя собственная форма для предоставления пользователю возможности ввести выбранное им имя и пароль. Проблема в том, что код сценария неполон. Используя магниты, расположенные ниже, завершите код сценария так, чтобы новый пользователь мог создать учетную запись и присоединиться к сообществу «Несоответствия».

Это форма «Создание учетной записи».




```
<?php
  require_once('appvars.php');
  require_once('connectvars.php');

  // Соединение с базой данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  if (isset($_POST['submit'])) {
    // Извлечение данных профиля из суперглобального массива $_POST
    $username = mysqli_real_escape_string($dbc, trim($_POST['username']));
    $password1 = mysqli_real_escape_string($dbc, trim($_POST['password1']));
    $password2 = mysqli_real_escape_string($dbc, trim($_POST['password2']));

    if (!empty($username) && !empty($password1) && !empty($password2) &&
        $password1 == $password2) {
      // Проверка того, что никто из уже записанных пользователей не пользуется таким же именем,
      // как то, которое ввел новый пользователь
      $query = "SELECT * FROM mismatch_user WHERE username = '$username'";
      $data = mysqli_query($dbc, $query);
      if (mysqli_num_rows($data) == 0) {
        // Имя, введенное пользователем, не используется, поэтому добавляем данные в базу
        $query = "INSERT INTO mismatch_user (username, password, join_date) VALUES " .
                 "('$username', SHA('$password1'), NOW())";
        mysqli_query($dbc, $query);

        // Вывод подтверждения пользователю
        echo '<p>Ваша новая учетная запись успешно создана. Вы можете войти в приложение и ' .
            '<a href="editprofile.php">отредактировать свой профиль</a>.</p>';

        mysqli_close($dbc);
        exit();
      }
```

Извлечение данных, введенных пользователем, с удалением лишних пробелов и экранированием проблемных символов.

Проверка, все ли данные введены и оба ли пароля одинаковые.

Выполнение запроса с целью проверки, нет ли в базе данных пользователя с таким именем.

Если такого пользователя нет, мы можем добавить запись, используя запрос INSERT.

Здесь может использоваться любой из двух введенных паролей, так как они уже прошли проверку на тождественность.

Вывод пользователю подтверждения об успешном создании для него учетной записи и завершение сценария.

```
else {
    // Учетная запись с таким именем уже существует в базе данных, поэтому выводится сообщение
об ошибке
    echo '<p class="error">Учетная запись с таким именем уже существует. Введите, пожалуйста,
другое</p>';
    $username = "";
  }
 }
 else {
  echo '<p class="error">Вы должны ввести все данные для создания учетной записи, в том числе пароль —
дважды.</p>';
 }
}

mysqli_close($dbc);
?>

<p>Введите, пожалуйста, ваше имя и пароль для создания учетной записи в приложении
&amp;quot;Несоответствия&amp;quot;.</p>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
 <fieldset>  <legend>Входные данные</legend>
  <label for="username">Имя пользователя:</label>
  <input type="text" id="username" name="username" value="<?php if (!empty($username)) echo
$username; ?>" /><br />
  <label for="password1">Пароль:</label>
  <input type="password" id="password1" name="password1" /><br />
  <label for="password2">Повторите пароль:</label>
  <input type="password" id="password2" name="password2" /><br />
 </fieldset>
 <input type="submit" value="Создать" name="submit" />
</form>
```

Обнуление переменной $username, для того чтобы поле «Имя пользователя» было пустым.

Имя пользователя используется, поэтому выводится сообщение об ошибке.

Одно или несколько полей пусты, поэтому выводится сообщение об ошибке.

signup.php

Не бывает глупых вопросов

В: Почему мы просто не можем использовать HTTP-аутентификацию для создания учетных записей?

О: Потому что задачей сценария «Создание учетной записи» является не ограничение доступа к веб-страницам, а предоставление пользователю возможности ввести уникальное имя и пароль и затем добавить эти данные в базу. Конечно, можно использовать окно диалога HTTP-аутентификации как интерфейс ввода имени пользователя и его пароля, но использовать для этих целей саму функцию аутентификации только лишь для создания учетной записи — это явный перегиб. Для такой цели лучше создать свою собственную форму, при этом вы получаете преимущество в том, что подстраховываете пользователя от опечаток при вводе пароля вслепую, предоставляя возможность вводить его два раза.

В: Значит, сценарий «Создание учетной записи» осуществляет вход пользователей в приложение после создания учетных записей?

О: Нет. Это нерезонно, в частности, и потому, что задачу входа пользователя в приложение уже решает сценарий «Вход в приложение» и нет никакой необходимости повторять тот же код в сценарии «Создание учетной записи». Вместо этого сценарий «Создание учетной записи» предоставляет пользователю гиперссылку на страницу «Редактирование профиля», на которую пользователь, предположительно, захочет зайти сразу же после создания учетной записи. А так как он еще не успел войти в приложение, ему будет выведено окно диалога «Вход в приложение» как часть попытки получить доступ к защищенной странице «Редактирование профиля». Таким образом, сценарий «Создание учетной записи» все равно приводит пользователя к окну диалога «Вход в приложение» через страницу «Редактирование профиля», а не напрямую.

Дайте пользователю возможность создать учетную запись

У нас есть сценарий «Создание учетной записи», но как пользователь откроет соответствующую страницу? Мы должны дать пользователю понять, как он может создать учетную запись. По одному из вариантов можно поместить гиперссылку «Создание учетной записи» на главную страницу приложения «Несоответствия». Неплохая идея, но тогда нам нужно включать или выключать ее в зависимости от того, вошел пользователь в приложение или нет. Другой вариант просто предусматривает размещение гиперссылки «Создание учетной записи» на странице «Вход в приложение».

Когда новый пользователь, например, щелкает кнопкой мыши на гиперссылках «Просмотр профиля» или «Редактирование профиля» на главной странице, сценарий «Вход в приложение» предлагает ему ввести свое имя и пароль. Так как у него еще нет учетной записи, а следовательно, имени и пароля, он, скорее всего, нажмет кнопку «Отмена», чтобы выпутаться из этой тупиковой ситуации. Здесь у нас появляется возможность вывести такому пользователю гиперссылку на страницу «Создание учетной записи». Сделать это можно, слегка изменив сообщение об ошибке, выводимое сценарием «Вход в приложение», чтобы оно включало гиперссылку на сценарий `signup.php`.

Ниже приведен оригинальный код сообщения об ошибке сценария «Вход в приложение»:

```
exit('<h3>Несоответствия</h3>Извините, вы должны ввести ваше имя и пароль, ' .
      'чтобы войти в приложение и получить доступ к этой странице.');
```

Этот код просто сообщает об ошибке без какого-либо упоминания о том, как можно создать учетную запись в приложении «Несоответствия».

Такой код практически появляется в двух различных частях сценария «Вход в приложение»: когда не введено имя пользователя и/или его пароль и когда они введены неправильно. Идея поместить гиперссылку на страницу «Создание учетной записи» в сообщение об ошибке в обоих случаях выглядит достаточно логично. Ниже показано, как бы мог выглядеть код в таком случае.

Этот код значительно полезнее, так как он генерирует гиперссылку на сценарий «Создание учетной записи», что дает пользователю возможность создать учетную запись.

```
exit('<h3>Несоответствия</h3>Извините, вы должны ввести ваше имя и пароль, ' .
      'чтобы войти в приложение и получить доступ к этой странице.' .
      'Если у вас нет учетной записи, вы можете ее <a href="signup.php">создать</a>.');
```

Ничего необычного, стандартная HTML-гиперссылка на сценарий signup.php.

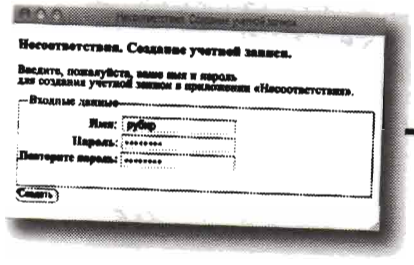
Тест-драйв

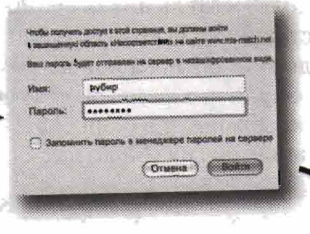
Добавьте функцию «Создание учетной записи» в приложение «Несоответствия».

Создайте новый текстовый файл с именем `signup.php` и введите в него код добавления учетной записи (или загрузите сценарий с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Затем для тех пользователей, которые не могут войти в приложение, добавьте в сценарий `login.php` гиперссылки на сценарий «Создание учетной записи».

Загрузите все сценарии на ваш веб-сервер и откройте страницу «Создание учетной записи» в браузере. Создайте новую учетную запись и войдите в приложение. Отредактируйте и просмотрите свой профиль, чтобы убедиться, что сценарии «Создание учетной записи» и «Вход в приложение» работают правильно. Теперь приложение обрело тот персонализированный оттенок, которого ему не хватало.

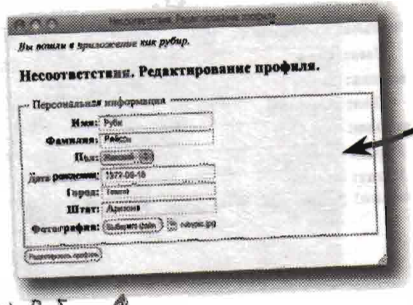
Щелкните кнопкой мыши по гиперссылке «Просмотр профиля» или «Редактирование профиля», чтобы войти в приложение и получить доступ к персонализированным страницам. Конечно, это будет работать только при условии, что вы уже добавили запись с соответствующим именем и паролем в таблицу базы данных.



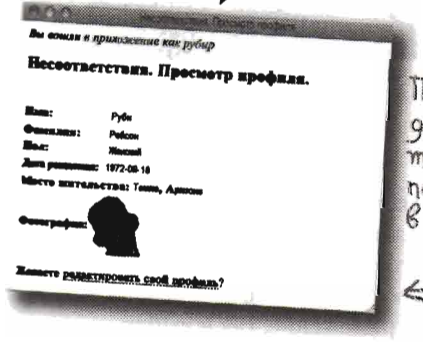


Для того чтобы Руби могла войти в приложение, используется HTTP-аутентификация, базирующаяся на информации ее учетной записи.

Учетные записи пользователей и процедура их входа в приложение превращают безликое приложение в сообщество заинтересованных людей.



Великолепно! Я могу войти в приложение «Несоответствия» и не только просматривать профили всех пользователей, но и редактировать мой собственный.

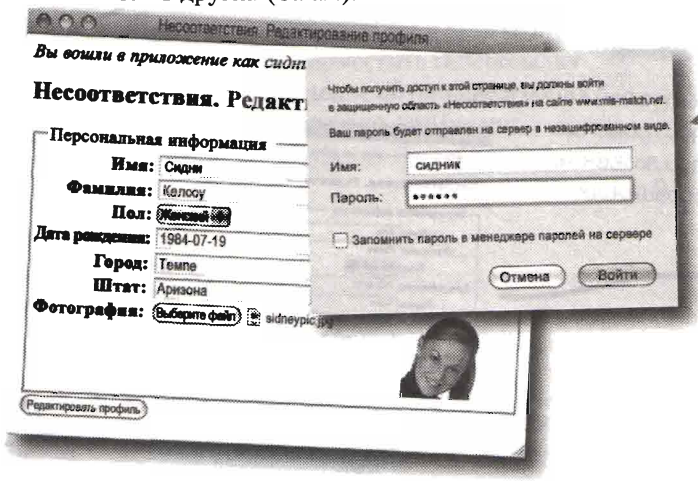


Профиль Руби доступен только после входа в приложение.

Я пользуюсь компьютером вместе с двумя своими соседями по комнате. Мне бы не хотелось, чтобы у них был полный доступ к моему профилю в приложении «Несоответствия». Мне необходима возможность выхода из системы!

Сайты интернет-сообществ должны предоставлять пользователям возможность выхода из приложения, чтобы в случае совместного использования одного компьютера одни его пользователи не имели полного доступа к персональным данным других.

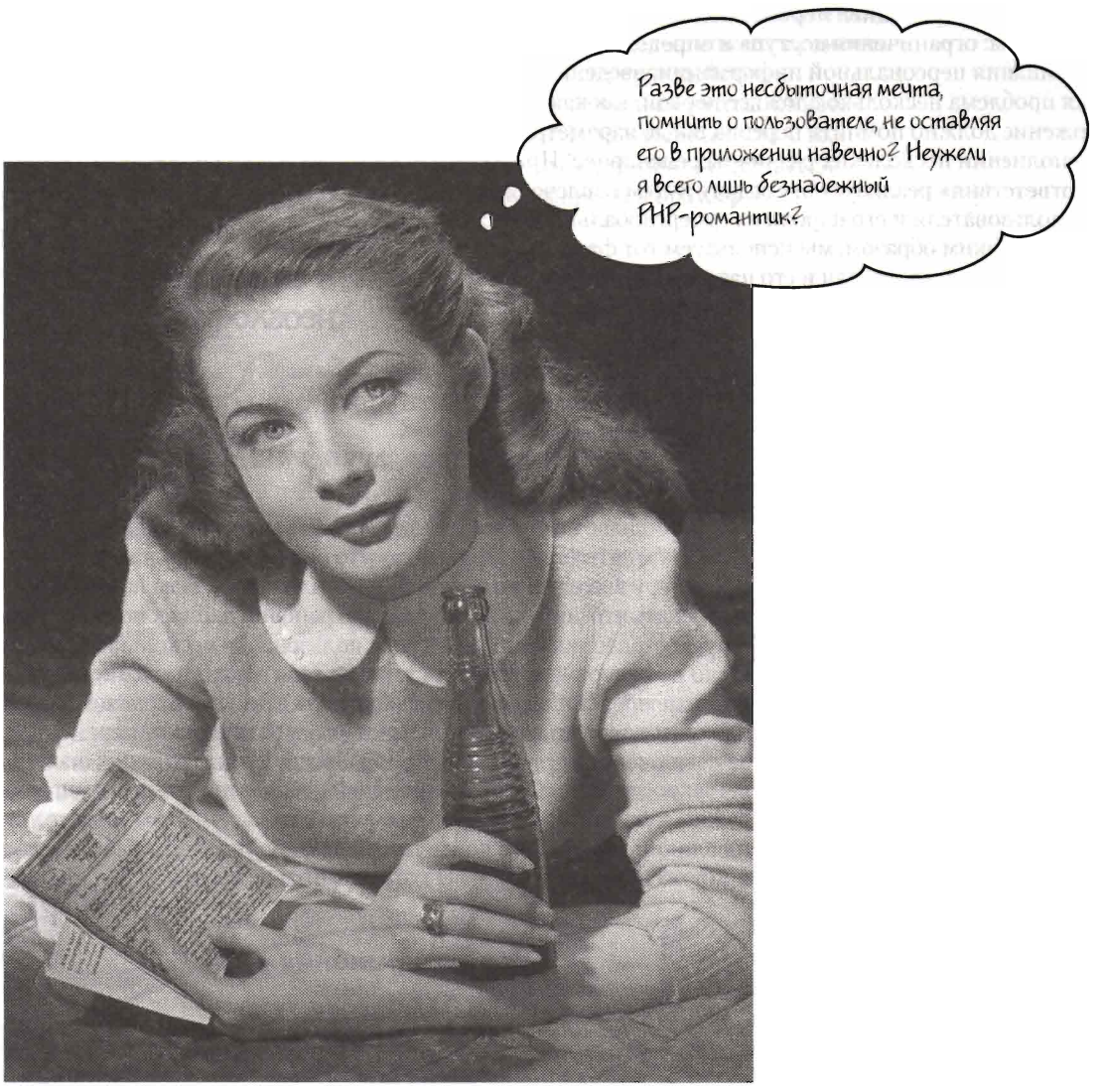
Предоставление пользователям возможности выхода из системы может показаться простой задачей, но она оказывается достаточно сложной, что связано с HTTP-аутентификацией. Проблема заключается в том, что HTTP-аутентификация производится только один раз для данной страницы или перечня страниц, после чего все ее параметры сохраняются и сбрасываются только после закрытия браузера. Иначе говоря, пользователь никак не может «полностью выйти» из страницы или группы страниц, для которых он прошел процедуру HTTP-аутентификации, до тех пор, пока или не закроется браузер, или он не сможет как-то обнулить параметры текущей сессии HTTP-аутентификации. Последнюю возможность в одних браузерах (например, Firefox) проще реализовать, чем в других (Safari).



Как только вы вошли в приложение, вы «остаетесь» в нем до тех пор, пока не закроется браузер.

Процедура «Выход из приложения» позволит Сидни контролировать доступ к данным ее персонального профиля.

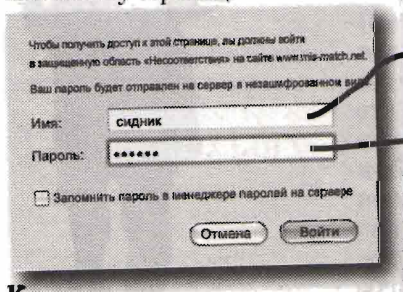
Хотя HTTP-аутентификация предоставляет простой и удобный способ для поддержки входа пользователя в приложение «Несоответствия», она не дает возможности управлять процессом выхода пользователя из системы. Нам необходим способ, позволяющий пользователю как войти в приложение, так и выйти, как только у него возникнет такое желание.


Разве это несбыточная мечта, помнить о пользователе, не оставляя его в приложении навечно? Неужели я всего лишь безнадежный PHP-романтик?

Иногда все, что вам нужно, — это куки

С HTTP-аутентификацией первоначально было связано решение двух проблем: ограничения доступа к определенным страницам и запоминания персональной информации, введенной пользователем. Вторая проблема несколько сложнее первой, так как для ее решения приложение должно помнить персональные параметры пользователя при исполнении нескольких различных сценариев. Приложение «Несоответствия» реализует эту задачу путем извлечения имени пользователя и его пароля из суперглобального массива `$_SERVER`. Таким образом, мы используем тот факт, что PHP сохраняет имя пользователя и его пароль, полученные в результате HTTP-аутентификации, в суперглобальной переменной, доступной множеству страниц.

В результате HTTP-аутентификации данные пользователя сохраняются на постоянной основе, но не могут быть удалены, когда в этом возникнет необходимость.



`$_SERVER['PHP_AUTH_USER']`

`$_SERVER['PHP_AUTH_PW']`

Имя пользователя и пароль сохраняются в суперглобальном массиве `$_SERVER` на постоянной основе.

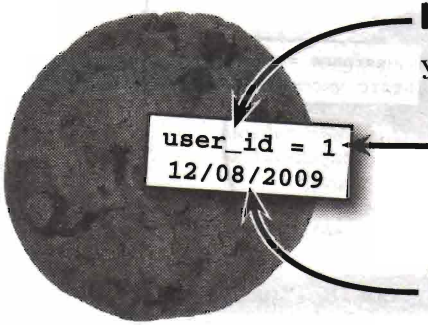
Но нас больше не устраивает замечательный сервис, предоставляемый HTTP-аутентификацией, потому что она не поддерживает процедуру выхода пользователя из приложения. Нам необходимо подыскать что-нибудь другое для сохранения данных пользователя и предоставления их в совместное пользование нескольким страницам. Одно из возможных решений заключается в использовании куки, которые представляют собой фрагменты данных, сохраняемых браузером на компьютере пользователя. Куки очень напоминают переменные PHP за исключением того, что они продолжают существовать и после того, как вы закроете браузер, выключите свой компьютер и т. п. Что еще более важно, куки могут быть удалены, а это означает, что вы можете устранить их влияние на процесс после того, как закончите свою работу и дадите приложению понять, что вы хотите выйти из него.

Куки позволяют вам сохранять данные на клиентском компьютере на постоянной основе, так что они могут пережить любой сценарий... и быть удалены по вашему желанию!

Куки сохраняются на компьютере пользователя его браузером. У вас есть возможность получить к ним доступ из PHP-кода, и они могут быть совместно использованы не только на нескольких страницах одного приложения, но и в нескольких сессиях. Поэтому когда пользователь закрывает браузер, он фактически не выходит из приложения «Несоответствия» автоматически. Но это не является проблемой, потому что мы можем удалить куки в любой момент выполнения сценария, предоставив пользователю процедуру выхода из приложения. При этом пользователь совершенно самостоятельно и независимо решает, когда ему необходимо выйти из приложения.

Что такое куки?

Куки сохраняет **одно значение данных** под уникальным именем подобно переменной PHP. Но в отличие от переменной для куки может быть определен момент времени, когда он прекращает свое существование, или, говоря иначе, срок действия. Когда истекает срок действия, куки удаляется. Поэтому их нельзя рассматривать как бессмертные объекты, просто они живут дольше PHP-переменных. Вы можете создать куки без указания срока действия, тогда они не будут ничем отличаться от обычных переменных и будут удалены, как только закроется браузер.



Имя

Уникальное имя куки.

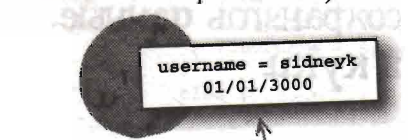
Значение

Значение, сохраненное в куки.

Срок действия

Срок действия куки — момент времени, когда он будет уничтожен.

Куки позволяют вам сохранить строку текста под определенным именем: примерно так же, как и текстовая переменная PHP. Тот факт, что куки переживают обычные переменные сценариев, делает их исключительно удобным средством, особенно в ситуациях, когда приложение состоит из нескольких страниц, которым необходимо помнить некоторые данные (например, информацию о входе пользователя в приложение).



Момент истечения срока действия куки установлен далеко в будущем, что делает его более постоянным.

Момент истечения срока действия куки не установлен. Куки будет удалено при закрытии браузера.

Итак, приложение «Несоответствия» могло бы сымитировать постоянство существования переменных, предоставляемое суперглобальным массивом $_SERVER путем создания пары куки: одного — для имени пользователя, другого — для его пароля. Так как в действительности нам нет необходимости постоянно помнить пароль, было бы более полезно вместо него запомнить идентификатор пользователя.

Не бывает глупых вопросов

В: Откуда столько шума по поводу способности куки надолго сохранять данные? Разве данные, сохраненные в базе данных, не могут содержаться там как угодно долго?

О: Да, базы данных действительно могут хранить данные как угодно долго, и в этом они превосходят куки, потому что в них нет такого понятия, как момент истечения срока действия. Если вы поместили данные в базу, они будут храниться там до тех пор, пока вы не удалите их явно. Действительное преимущество куки в вопросах сохранения данных — это удобство. Нам нет никакой необходимости вечно помнить текущее имя пользователя или его идентификатор только лишь для того, чтобы он смог получить доступ к редактированию своего профиля. Все, что нам нужно в данном случае, — это знать, что это за пользователь. В действительности нам необходимо временное постоянство, что, возможно, прозвучит для вас как оксюморон, если вы не примете во внимание, что нам нужно, чтобы данные были доступны дольше, чем существует веб-страница, но не вечно.

Использование куки в PHP

PHP предоставляет доступ к куки через функцию `setcookie()` и суперглобальную переменную `$_COOKIE`. Функция `setcookie()` используется для установки значения и (выборочно) срока действия. А суперглобальная переменная `$_COOKIE` применяется для получения данных куки.

```
setcookie('username', 'сидник');
```

Первый аргумент, передаваемый функции setcookie(), — это имя куки.

Значение, которое должно быть сохранено в куки, передается функции setcookie() вторым аргументом.


```
echo('<p class="login">Вы вошли в приложение как ' . $_COOKIE['username'] . ',</p>');
```

Имя куки используется как ссылка на его значение в суперглобальном массиве $_COOKIE.

Удобство куки заключается в том, что его данные доступны множеству сценариев приложения, поэтому вы можете запомнить имя пользователя, и он будет избавлен от необходимости входить в приложение каждый раз, когда у него возникнет необходимость перейти с одной страницы на другую внутри приложения. Но не забывайте, что мы также должны запомнить в куки идентификатор пользователя, так как он выполняет функцию первичного ключа и поэтому необходим для запросов к базе данных.

Функция PHP setcookie() позволяет вам сохранять данные в куки.

```
setcookie('user_id', '1');
```

Значения сохраняются в куки всегда в виде текста, поэтому, хотя идентификатор пользователя является целым числом, мы сохраняем его как текст '1'.

Функция `setcookie()` принимает также третий, необязательный аргумент, который устанавливает срок действия куки. При достижении этой даты куки автоматически удаляется. Если дата истечения срока действия куки не устанавливается, как в этом примере, куки автоматически удаляется при закрытии браузера.

Время поработать

Перевод приложения «Несоответствия» на использование куки требует больших усилий, чем просто написание сценария выхода из него. Первое, что мы должны сделать, — это пересмотреть сценарий «Вход в приложение» и внести в него изменения, позволяющие использовать куки вместо HTTP-аутентификации. Просмотрите и прокомментируйте те фрагменты кода, которые, по вашему мнению, требуют изменений, обеспечивающих использование куки.

```
<?php
   require_once('connectvars.php');

   if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW'])) {
      // Имя пользователя и/или его пароль не были введены,
      // поэтому отправляются заголовки аутентификации.
      header('HTTP/1.1 401 Unauthorized');
      header('WWW-Authenticate: Basic realm="Несоответствия"');
      exit('<h3>Несоответствия</h3>Простите, вы должны ввести ваше имя и пароль для того, ' .
           'чтобы войти в приложение и получить доступ к этой странице.';
           'Если у вас нет учетной записи, вы можете ее <a href="signup.
php">создать</a>.';
   }

   // Соединение с базой данных
   $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

   // Получение введенных пользователем данных для аутентификации
   $user_username = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_USER']));
   $user_password = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_PW']));

   // Поиск имени пользователя и его пароля в базе данных
   $query = "SELECT user_id, username FROM mismatch_user " .
   $data = mysqli_query($dbc, $query);

   if (mysqli_num_rows($data) == 1) {
      // Процедура входа прошла нормально, присваиваем переменным значения
      // идентификатора пользователя и его пароля
      $row = mysqli_fetch_array($data);
      $user_id = $row['user_id'];
      $username = $row['username'];
   }
   else {
      // Имя пользователя и/или его пароль введены неверно,
      // поэтому отправляются заголовки аутентификации.
      header('HTTP/1.1 401 Unauthorized');
      header('WWW-Authenticate: Basic realm="Несоответствия"');
      exit('<h3>Несоответствия</h3>Простите, вы должны ввести ваше имя и пароль для того, ' .
           'чтобы войти в приложение и получить доступ к этой странице.';
           'Если у вас нет учетной записи, вы можете ее <a href="signup.
php">создать</a>.';
   }

   // Подтверждение успешного входа в приложение
   echo('<p class="login">Вы вошли в приложение как ' . $username . '.</p>')
?>
```

login.php

Решение задачи

Перевод приложения «Несоответствия» на использование куки требует больших усилий, чем просто написание сценария выхода из него. Первое, что мы должны сделать, — это пересмотреть сценарий «Вход в приложение» и внести в него изменения, позволяющие использовать куки вместо HTTP-аутентификации. Просмотрите и прокомментируйте те фрагменты кода, которые, по вашему мнению, требуют изменений, обеспечивающих использование куки.

Нам необходимо проверить существование куки, чтобы узнать, вошел пользователь в приложение или нет.

Вместо того чтобы извлекать имя пользователя и его пароль из данных окна аутентификации, нам необходимо использовать обычную форму и получить данные через суперглобальный массив $_POST.

```
<?php
   require_once('connectvars.php');

   if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW'])) {
      // Имя пользователя и/или его пароль не были введены,
      // поэтому отправляются заголовки аутентификации.
      header('HTTP/1.1 401 Unauthorized');
      header('WWW-Authenticate: Basic realm="Несоответствия"');
      exit('<h3>Несоответствия</h3>Простите, вы должны ввести ваше имя и пароль для того, ' .
           'чтобы войти в приложение и получить доступ к этой странице.' .
           'Если у вас нет учетной записи, вы можете ее <a href="signup.
php">создать</a>.');
   }

   // Соединение с базой данных
   $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

   // Получение введенных пользователем данных для аутентификации
   $user_username = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_USER']));
   $user_password = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_PW']));

   // Поиск имени пользователя и его пароля в базе данных
   $query = "SELECT user_id, username FROM mismatch_user " .
   $data = mysqli_query($dbc, $query);

   if (mysqli_num_rows($data) == 1) {
      // Процедура входа прошла нормально, присваиваем переменным значения
      // идентификатора пользователя и его пароля
      $row = mysqli_fetch_array($data);
      $user_id = $row['user_id'];
      $username = $row['username'];
   }
   else {
      // Имя пользователя и/или его пароль введены неверно,
      // поэтому отправляются заголовки аутентификации.
      header('HTTP/1.1 401 Unauthorized');
      header('WWW-Authenticate: Basic realm="Несоответствия"');
      exit('<h3>Несоответствия</h3>Простите, вы должны ввести ваше имя и пароль для того, ' .
           'чтобы войти в приложение и получить доступ к этой странице.' .
           'Если у вас нет учетной записи, вы можете ее <a href="signup.
php">создать</a>.');
   }

   // Подтверждение успешного входа в приложение
   echo('<p class="login">Вы вошли в приложение как ' . $username . '.</p>');
?>
```

Нам больше не нужно отправлять HTTP-заголовок.

Нет никакой необходимости изменять запрос!

Здесь вместо присвоения данных переменным мы должны создать пару куки.

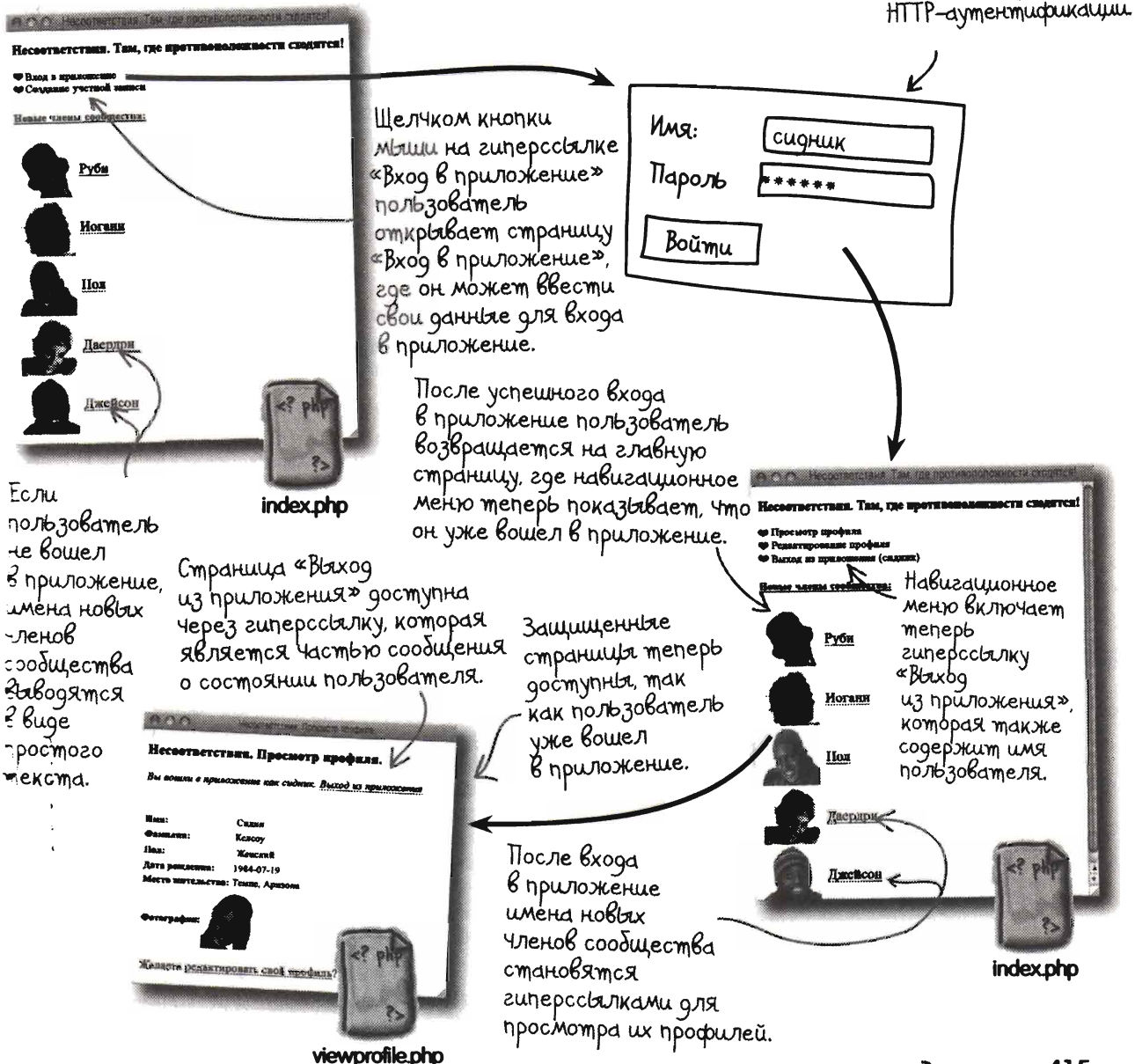
login.php

Так как мы больше не полагаемся на окно диалога HTTP-аутентификации для запроса имени пользователя и его пароля, нам необходимо создать HTML-форму для ввода входных данных.

Переосмысление процедуры входа в приложение

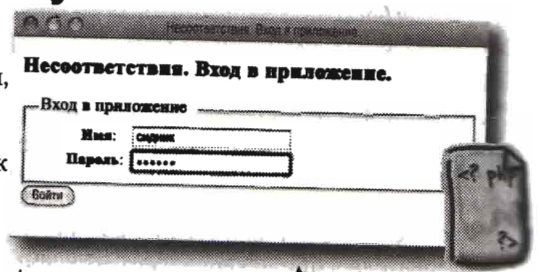
Использование куки вместо HTTP-аутентификации для входа пользователя в приложение «Несоответствия» влечет за собой больше, чем просто замену метода сохранения данных. Что, например, можно сказать по поводу интерфейса входа пользователей в приложение? Вход в приложение с использованием куки должен применять свою собственную форму, так как он не может полагаться на окно HTTP-аутентификации, чтобы предоставлять пользователю возможность ввода имени и пароля. Мы не только должны создать эту форму, но также продумать, как это повлияет на процесс входа пользователя в приложение и его доступ к необходимым страницам.

Новая форма для ввода имени пользователя и его пароля занимает место окна диалога HTTP-аутентификации.



Вход в приложение с использованием куки

Новая версия сценария «Вход в приложение», полагающаяся на куки в вопросах запоминания входных данных пользователя, немного сложнее предшествующей, так как она должна предоставлять свою собственную форму для ввода имени пользователя и его пароля. Но она значительно удобнее, так как дает возможность выхода из приложения.




```
<?php
    require_once('connectvars.php');

    // Обнуление сообщения об ошибке
    $error_msg = "";
    // Если пользователь еще не вошел в приложение, попытка войти
    if (!isset($_COOKIE['user_id'])) {

      if (isset($_POST['submit'])) {
          // Соединение с базой данных
          $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
          // Получение введенных пользователем данных для аутентификации
          $user_username = mysqli_real_escape_string($dbc, trim($_POST['username']));
          $user_password = mysqli_real_escape_string($dbc, trim($_POST['password']));
          if (!empty($user_username) && !empty($user_password)) {
              // Поиск имени пользователя и его пароля в базе данных
              $query = "SELECT user_id, username FROM mismatch_user " .
                       "WHERE username = '$user_username' AND password = SHA('$user_password')";
              $data = mysqli_query($dbc, $query);
              if (mysqli_num_rows($data) == 1) {
                  // Вход в приложение прошел успешно, сохранение в куки имени пользователя и его
идентификатора.
                  // Переход на главную страницу
                  $row = mysqli_fetch_array($data);
                  setcookie('user_id', $row['user_id'], time() + (60 * 60 * 24 * 30));    // срок действия 30 дней
                  setcookie('username', $row['username'], time() + (60 * 60 * 24 * 30));  // срок действия 30 дней
                  $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
                  header('Location: ' . $home_url);
              }
              else {
                  // Имя пользователя/его пароль введены неверно, создание сообщения об ошибке
                  $error_msg = 'Извините, для того чтобы войти в приложение, вы должны ввести правильное
имя и пароль.';
              }
          }
          else {
              // Имя пользователя/его пароль не введены, создание сообщения об ошибке
              $error_msg = 'Извините, для того чтобы войти в приложение, вы должны ввести имя и пароль.';
          }
      }
    }
?>
<html>
    <head>
        <title>Несоответствия. Вход в приложение.</title>
        <link rel="stylesheet" type="text/css" href="style.css" />
    </head>
    <body>
        <h3>Несоответствия. Вход в приложение.</h3>
```

Сообщение об ошибке теперь сохраняется в переменной и выводится, когда это требуется, по ходу выполнения сценария.

Это новая форма «Вход в приложение».

Проверка значения куки user_id с целью определения, вошел пользователь в приложение или нет.

Если пользователь не вошел в приложение, проводится проверка: не передаются ли его входные данные с формой.

Входные данные пользователя теперь поступили как данные формы, а не окна HTTP-аутентификации.

Вход пользователя в приложение путем сохранения в куки его имени и идентификатора.

Переадресация на главную страницу после успешного входа в приложение.

Если с входом в приложение что-то не ладится, составляется сообщение об ошибке.

В результате выполнения сценария «Вход в приложение» будет генерироваться стандартная HTML-страница, поэтому в нем необходимо поместить все необходимые HTML-теги.

продолжение на следующей странице...

```
<?php
   // Если куки не содержит данных,  выводятся сообщение об ошибке
   // и форма входа в приложение; в противном случае – подтверждение входа
   if (empty($_COOKIE['user_id'])) {
     echo '<p class="error">' . $error_msg . '</p>';
?>

   <form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
      <fieldset>
         <legend>Вход в приложение</legend>
         <label for="username">Имя пользователя:</label>
         <input type="text" name="username"
           value="<?php if (!empty($user_username)) echo $user_username; ?>" /><br />
         <label for="password">Пароль:</label>
         <input type="password" name="password" />
      </fieldset>
      <input type="submit" value="Войти" name="submit" />
   </form>
<?php
   }
   else {
      // Подтверждение успешного входа в приложение
      echo('<p class="login">Вы вошли в приложение как ' . $_COOKIE['username'] . '.</p>');
   }
?>

   </body>
</html>
```

Если пользователь к этому моменту все еще не вошел в приложение, выводится сообщение об ошибке.

Эти два поля формы используются для ввода имени пользователя и его пароля, чтобы пройти процедуру входа в приложение.

Все, что записано перед этой фигурной скобкой, является частью блока кода управляющей конструкции if.

Если пользователь к этому моменту прошел процедуру входа в приложение, ему выводится сообщение об этом.

HTML-теги, закрывающие веб-страницу.

Не бывает глупых вопросов

В: Какая необходимость в сохранении в куки и имени пользователя, и его идентификатора?

О: Так как и имя пользователя, и его идентификатор уникально определяют пользователя в базе данных `mismatch`, вы можете использовать любую из этих переменных для определения конкретного пользователя. Тем не менее `user_id` является более предпочтительной (более эффективной) ссылкой на пользователя для базы данных, потому что в ней эта колонка таблицы `mismatch_user` является первичным ключом. С другой стороны, `user_id` недостаточно понятна и мало что говорит обыкновенному человеку о том пользователе, которого она определяет; для него значительно содержательней имя пользователя, которое он видит на главной странице после успешного входа в приложение. Так как иногда несколько человек могут использовать один и тот же компьютер, очень важно сообщить им не только то, что они успешно вошли в приложение, но также и то, кто конкретно вошел.

В: Тогда почему бы не сохранить в куки и пароль как часть входных данных?

О: Пароль важен только для первоначальной проверки того, что пользователь, назвавшийся каким-либо именем, действительно им и является. Как только проверка окончена, какая-либо необходимость в пароле отпадает. Кроме того, пароль относится к чувствительным с точки зрения безопасности данным, поэтому, по возможности, лучше воздерживаться от сохранения его во временных хранилищах данных.

В: Похоже, что форма в сценарии «Вход в приложение» помещена внутри управляющей конструкции if. Это допустимо?

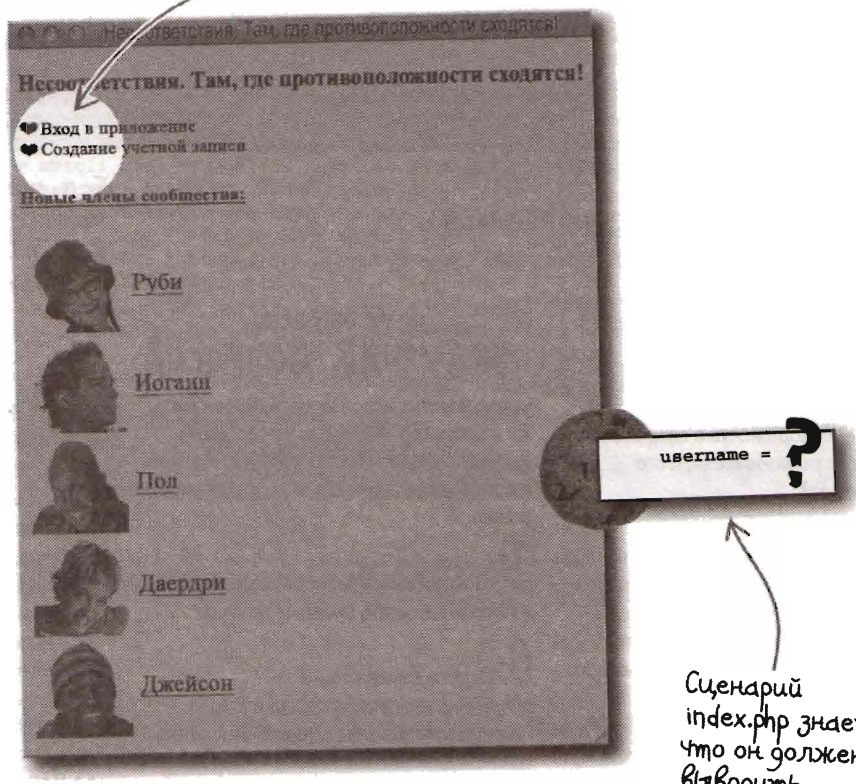
О: Да. Фактически это обычная практика, когда фрагменты HTML-кода «разрывают» PHP-код, как в случае с кодом сценария «Вход в приложение». Тот факт, что вы закрыли фрагмент PHP-кода тегом `?>`, вовсе не означает, что вы завершили развитие логики кода. Когда вы откроете следующий фрагмент PHP-кода тегом `<?php`, логика продолжит развиваться, начиная с того момента, на котором вы ранее прервали ее. В сценарии «Вход в приложение» HTML-форма расположена внутри блока кода первой управляющей конструкции `if`, в то время как ветвь `else` управляющей конструкции `if` размещена за кодом формы. Расположение PHP-кода таким способом избавляет вас от необходимости генерировать код формы с помощью группы не слишком удобочитаемых выражений `echo`.

Вход в приложение

Новая версия сценария «Вход в приложение» меняет процесс обработки данных в приложении «Несоответствия», требуя размещения простого навигационного меню на главной странице (`index.php`). Это меню играет значительную роль в приложении, так как предоставляет доступ к различным его частям, в данном случае — к страницам «Просмотр профиля» и «Редактирование профиля». Кроме того, через это меню пользователь может создать учетную запись, войти в приложение или выйти из него. Тот факт, что состав меню гибко изменяется в зависимости от того, вошел пользователь в приложение или еще нет, очень важен, потому что это предоставляет пользователю дополнительные удобства в работе.

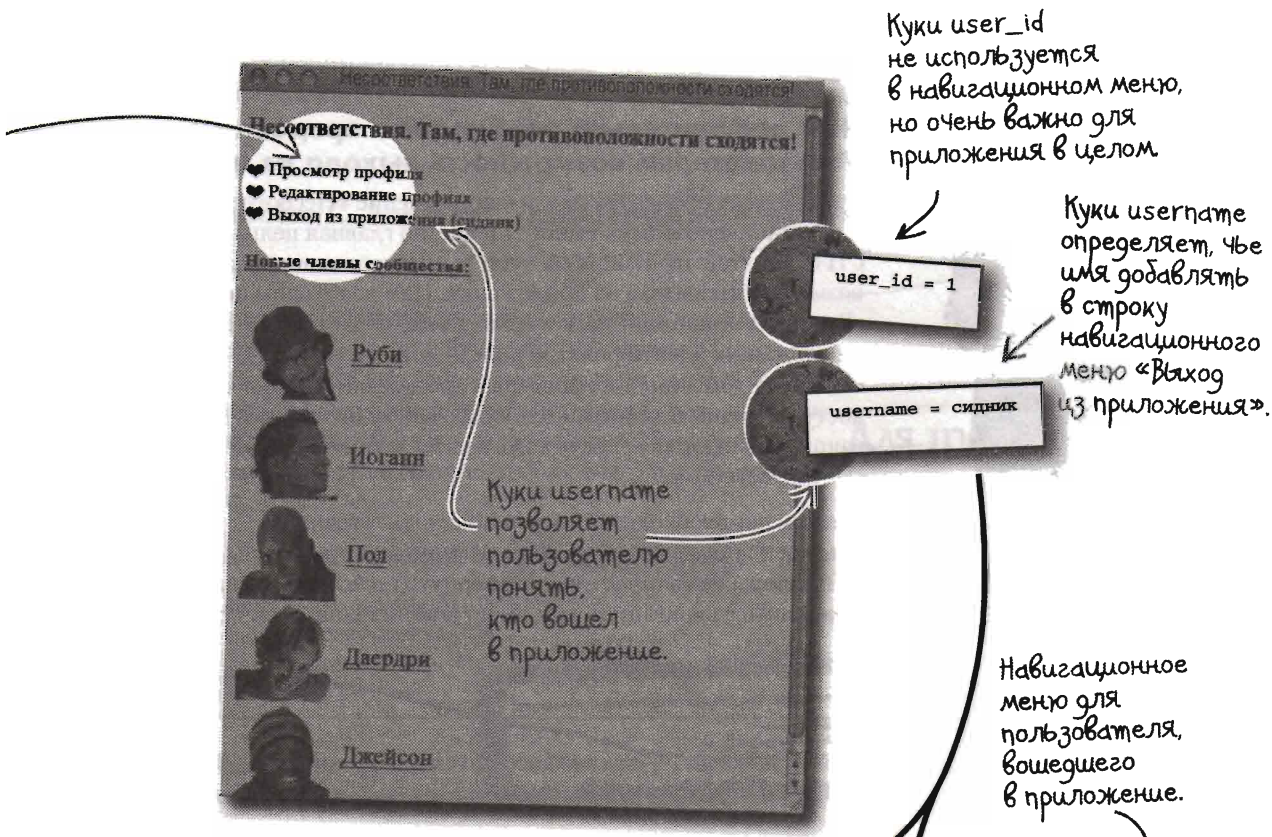
Если существует куки username, выводится навигационное меню другого состава.

Это меню появляется, когда пользователь не вошел в приложение, давая ему возможность либо войти, либо создать учетную запись.



Сценарий index.php знает, что он должен выводить ограниченное навигационное меню, когда он не находит куки username.

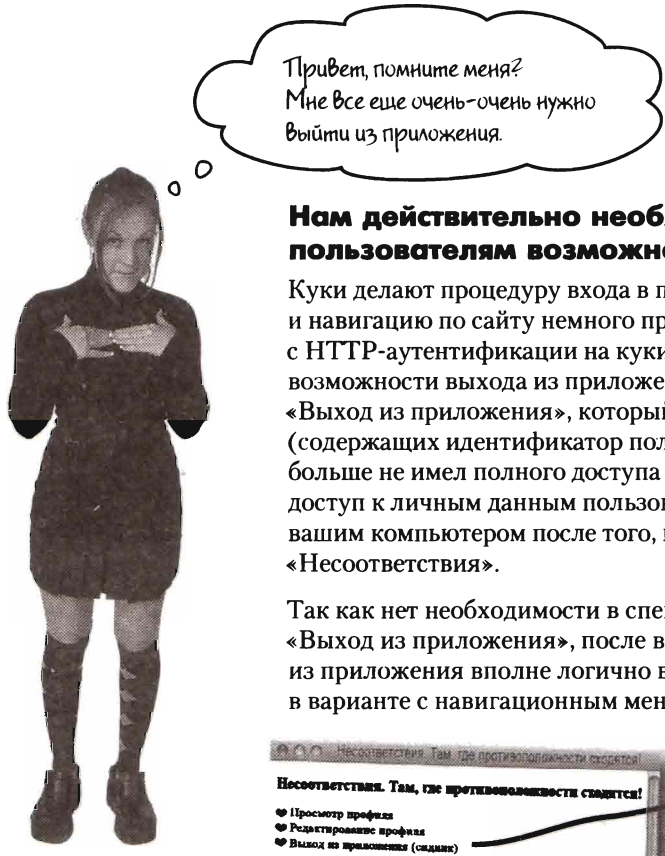
Навигационное меню создается в результате интерпретации PHP-кода в сценарии `index.php`. Этот код проверяет суперглобальный массив `$_COOKIE` на наличие куки `username` и таким образом определяет, прошел ли пользователь процедуру входа в приложение. Можно точно также использовать куки `user_id`, но так как имя пользователя необходимо на этом этапе для включения в строку меню `Выход из приложения` (имя пользователя), то логичнее проверять куки `username`.




```
// Создание меню навигации
if (isset($_COOKIE['username'])) {
   echo '&#10084; <a href="viewprofile.php">Просмотр профиля </a><br />';
   echo '&#10084; <a href="editprofile.php">Редактирование профиля </a><br />';
   echo '&#10084; <a href="logout.php">Выход из приложения (' . $_SESSION['username'] . ')</a>';
}
else {
   echo '&#10084; <a href="login.php">Вход в приложение</a><br />';
   echo '&#10084; <a href="signup.php">Создание учетной записи</a>';
}
```

Символ в виде сердечка перед каждой строкой меню можно вывести, используя этот HTML-код. Его поддерживает большинство браузеров.

Навигационное меню для посетителей (пользователей, не прошедших процедуру входа в приложение).



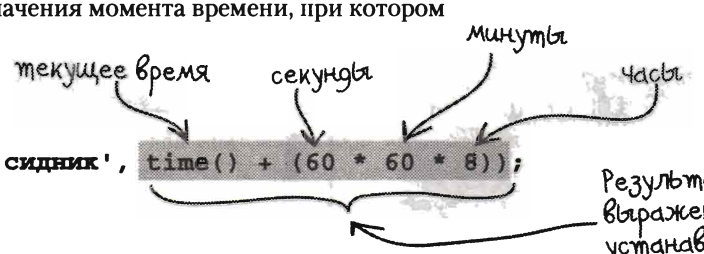
Нам действительно необходимо предоставить пользователям возможность выхода из приложения.

Куки делают процедуру входа в приложение «Несоответствия» и навигацию по сайту немного проще, но главная цель перехода с HTTP-аутентификации на куки — предоставление пользователям возможности выхода из приложения. Нам необходим новый сценарий «Выход из приложения», который должен будет удалять два куки (содержащих идентификатор пользователя и его имя), чтобы пользователь больше не имел полного доступа к приложению. Это должно закрыть доступ к личным данным пользователей, если кто-нибудь воспользуется вашим компьютером после того, как вы закончили работу с приложением «Несоответствия».

Так как нет необходимости в специальной странице «Выход из приложения», после выполнения процедуры выхода из приложения вполне логично вернуть пользователя на главную страницу в варианте с навигационным меню ограниченного доступа.

Выход из приложения означает удаление куки

Выход из приложения включает удаление двух куки (содержащих идентификатор пользователя и его имя). Это происходит с вызовом функции `setcookie()` и передачей ей в качестве аргумента такого значения момента времени, при котором куки должен быть уже удален.



Этот код устанавливает момент истечения срока действия куки на 8 часов вперед. Это означает, что куки будет автоматически удален через 8 часов. Но мы хотим удалить куки немедленно, что может быть осуществлено установкой в прошлое момента истечения срока действия куки. На какое время переноситься в прошлое, значения не имеет, выберите любой сдвиг во времени (например, один час) и отнимите его от текущего времени.

```
setcookie('username', 'сидник', time() - 3600);
```

60 секунд умножить на 60 минут равняется 3600 секундам, что составляет 1 час, а знак минус говорит о том, что это момент времени в прошлом.

Для того чтобы удалить куки, просто установите в прошлое момент истечения срока его действия.

В сценарии «Выход из приложения» отсутствует часть кода. Напишите недостающий код, обеспечивающий удаление куки перед переадресацией браузера на главную страницу.

```
<?php
   // Если пользователь вошел в приложение, удаление куки, приводящее к выходу его из приложения
   if ( ..................................... ) {
      // Установка момента истечения срока действия для куки,
      // содержащих идентификатор пользователя и его имя.
      // В результате эти куки удаляются/
      ..........................................................

      ..........................................................
   }

   // Переадресация на главную страницу
   $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '.....................';
   header('Location: ' . $home_url);
?>
```

В сценарии «Выход из приложения» отсутствует часть кода. Напишите недостающий код, обеспечивающий удаление куки перед переадресацией браузера на главную страницу.

```
<?php
   // Если пользователь вошел в приложение, удаление куки, приводящее к выходу его из приложения
   if ( isset($_COOKIE['user_id']) ) {
      // Установка момента истечения срока действия для куки,
      // содержащих идентификатор пользователя и его имя.
      // В результате эти куки удаляются/
      setcookie('user_id', '', time() - 3600);
      setcookie('username', '', time() - 3600);
   }

   // Переадресация на главную страницу
   $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
   header('Location: ' . $home_url);
?>
```

Только тот пользователь, который вошел в приложение накануне, может выйти из него.

Установка момента истечения срока действия куки на час назад, чтобы они были удалены системой.

Переадресация на главную страницу «Несоответствия» с использованием ее абсолютного URL.

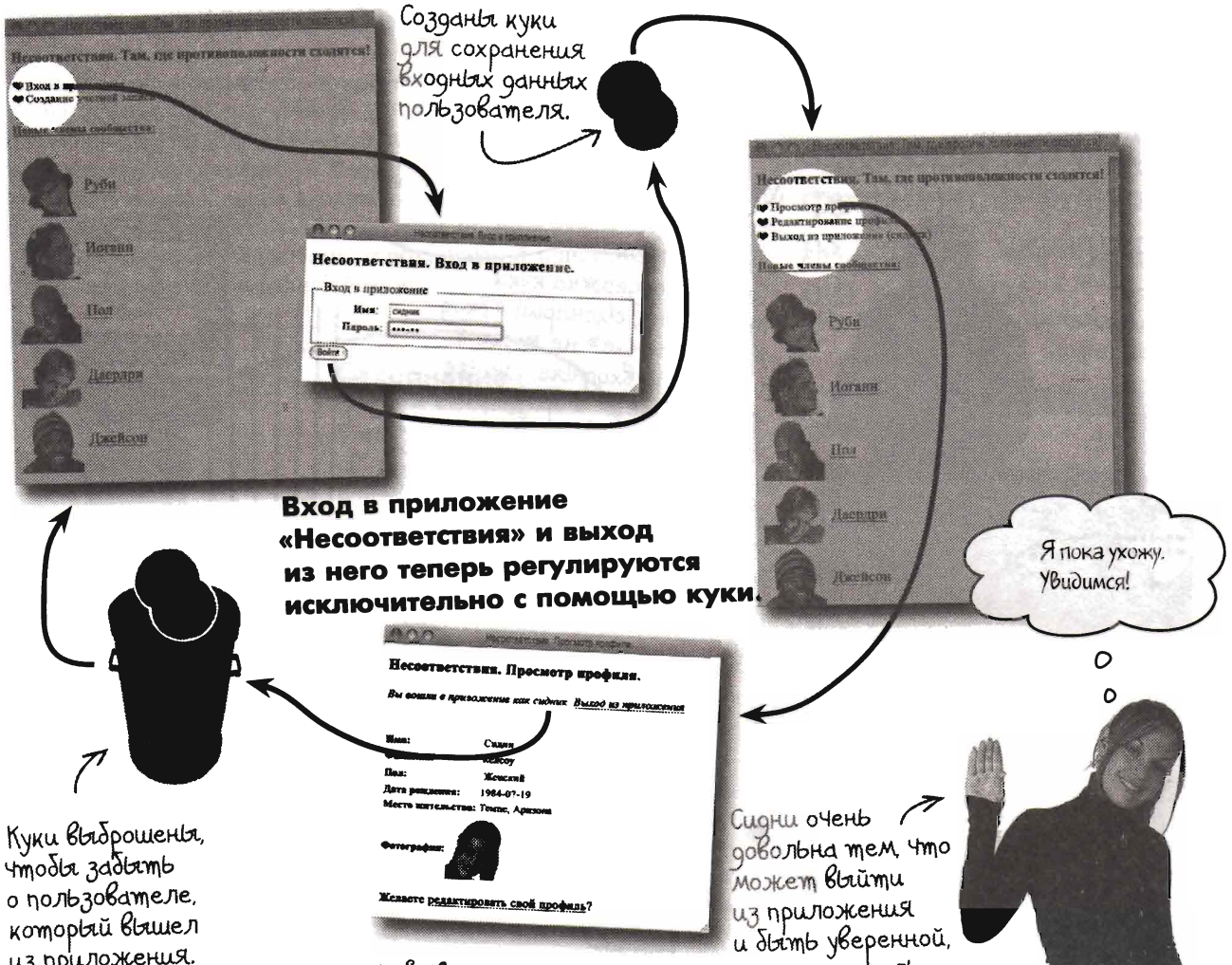
Используя HTTP-заголовок места расположения (location header), сценарий переадресует браузер на главную страницу.

Тест-драйв

Используйте куки для добавления в приложение «Несоответствия» возможности выхода.

Измените сценарии приложения «Несоответствия» так, чтобы в них использовались куки для процедуры входа пользователей в приложение и выхода из него (или загрузите их с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Переход на использование куки требует внесения изменений в сценарии `index.php`, `login.php`, `logout.php`, `editprofile.php` и `viewprofile.php`. Изменения в последних двух сценариях незначительны и заключаются только в том, что значения переменных `$user_id` и `$username` будут извлекаться из суперглобального массива `$_COOKIE`.

Загрузите все сценарии на ваш веб-сервер и откройте главную страницу приложения «Несоответствия» (`index.php`) в браузере. Обратите внимание на то, как выглядит навигационное меню, а затем щелкните кнопкой мыши по гиперссылке «Вход в приложение». Заметьте, что теперь навигационное меню изменилось и отражает ваше состояние. Щелкните кнопкой мыши по гиперссылке «Выход из приложения», чтобы удалить куки и выйти из приложения.



Не бывает глупых вопросов

В: Значит, чтобы выйти из приложения, достаточно просто удалить куки?

О: Да. Куки ответственны за сохранение всей входной информации в приложении «Несоответствия» (идентификатор пользователя и его имя), поэтому удаление этой информации приводит к выходу пользователя из приложения.

В: Почему для удаления куки время истечения срока их действия устанавливается на час назад? Это обязательно должен быть один час?

О: Нет. Куки автоматически удаляются браузером, как только истечет срок их действия. Поэтому куки будут удалены при установке любого момента времени в прошлом. Один час (3600 секунд) — это просто произвольный отрезок времени, выбранный для того, чтобы указать, что мы удаляем куки.

У пользователя приложения «Несоответствия» Джейсона, любителя путешествий, пирсинга и Говарда Стерна, в браузере отключена поддержка куки, в результате чего у него большие проблемы с входом в приложение.

Ну вот! В моем браузере отключена поддержка куки, и я не могу войти в приложение. Что же мне теперь делать?

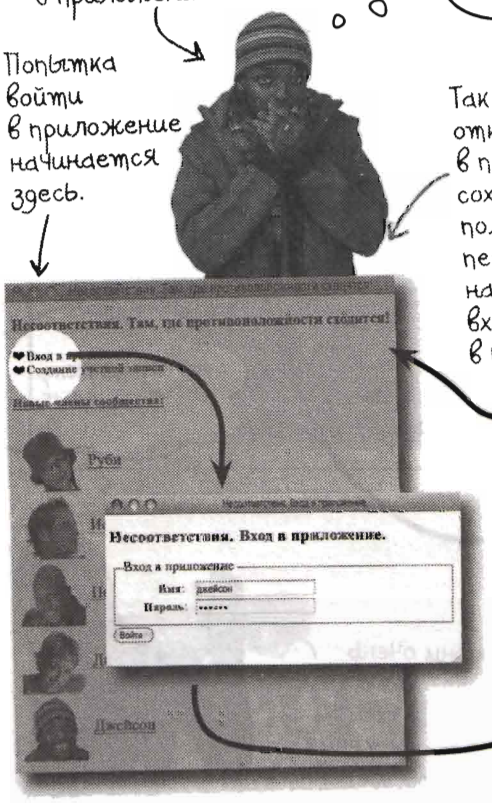
Попытка войти в приложение начинается здесь.

Так как поддержка куки отключена, сценарий «Вход в приложение» не может сохранить входные данные пользователя и браузер переадресовывается назад на главную страницу без входа пользователя в приложение.

Браузер

Браузер отвергает куки, не давая сценарию «Вход в приложение» сохранить входные данные пользователя.

Сервер пытается сохранить идентификатор пользователя и его имя в куки на компьютере пользователя.



Веб-сервер

Кого заботят проблемы Джейсона? Разве поддержка куки не включена у большинства?

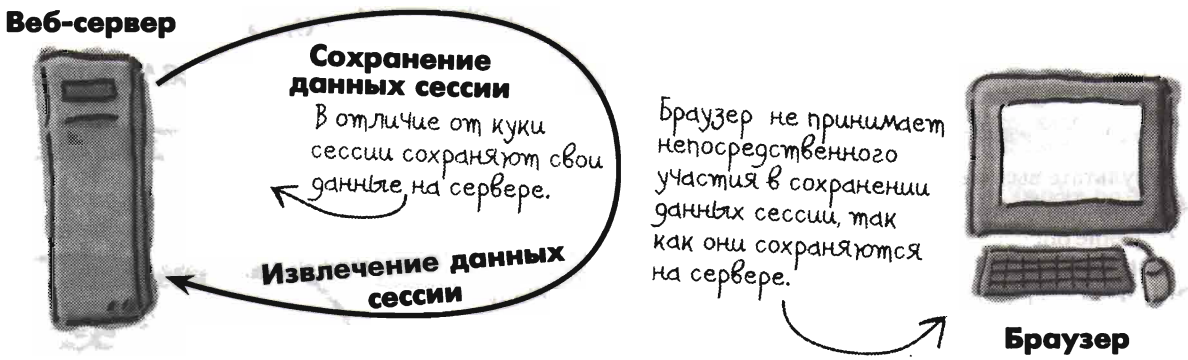
Это так, но веб-приложение должно быть доступно максимально возможному количеству пользователей.

Некоторые люди не хотят включать поддержку куки, потому что предпочитают более высокий уровень безопасности. Имея это в виду, есть смысл учесть нужды и тех людей, которые не полагаются на поддержку куки при входе в приложение. Более того, оказывается, существует возможность сохранять входные данные на сервере, а не на клиенте. А так как наш сценарий выполняется на сервере, целесообразно сохранять входные данные пользователей также на сервере.

Сессия не зависит от клиента

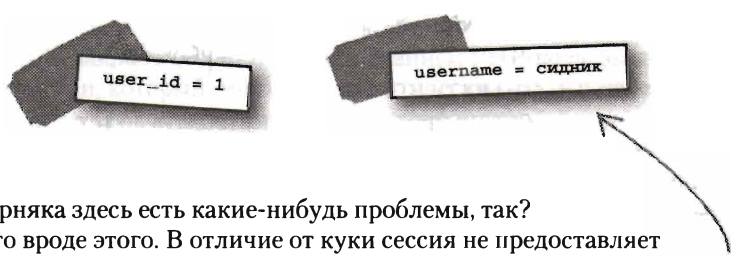
Куки, конечно, крутые ребята, но у них есть свои ограничения, которые находятся вне нашего контроля. Но что, если вы не будете зависеть от конфигурации браузера? Что, если бы мы могли сохранять данные непосредственно на сервере? **Сессии** обеспечивают как раз эту возможность, позволяя вам сохранять отдельные данные, так же как это делается и в случае использования куки, но не на клиенте, а на сервере. Это делает данные свободными от ограничений, присущих куки и связанных с конфигурацией браузера.

Сессии позволяют вам сохранять небольшие фрагменты данных на сервере независимо от клиента.



Сессии сохраняют данные в переменных, которые логически эквивалентны куки на сервере. Когда вы, используя код PHP, помещаете данные в переменную сессии, они сохраняются на сервере. Вы можете затем получить доступ к этим данным, и они сохраняют свои значения независимо от того, в каком сценарии приложения вы их используете. Так же, как и в случае с куки, вы можете удалить переменную сессии в любой момент, таким образом предоставляя пользователям процедуру выхода из приложения.

Раз данные сессии сохраняются на сервере, они более защищены и более надежны, чем данные, сохраненные в куки.



Пользователь не может вручную удалить данные сессии, используя свой браузер, что может оказаться проблемой в случае использования куки.

Наверняка здесь есть какие-нибудь проблемы, так? Что-то вроде этого. В отличие от куки сессия не предоставляет такого гибкого контроля над временем жизни ее переменной. Переменная сессии автоматически удаляется, как только сессия закончится, что обычно совпадает с моментом, когда пользователь закрывает браузер. Поэтому, хотя переменные сессии и не сохраняются на компьютере клиента, они косвенно зависят от браузера, так как удаляются при его закрытии.

Для переменных сессии не существует момента истечения срока действия, так как они автоматически удаляются при окончании сессии.

Сессия

Сессии названы так неслучайно. У них имеется вполне определенное время начала и окончания. Данные, ассоциируемые с сессией, появляются и прекращают свое существование в соответствии с открытием и закрытием сессии, причем все эти события контролируются PHP-кодом вашего приложения. Единственной ситуацией, которую вы контролировать не в состоянии, является закрытие браузера пользователем, в результате чего сессия также закрывается, хотите вы этого или нет.

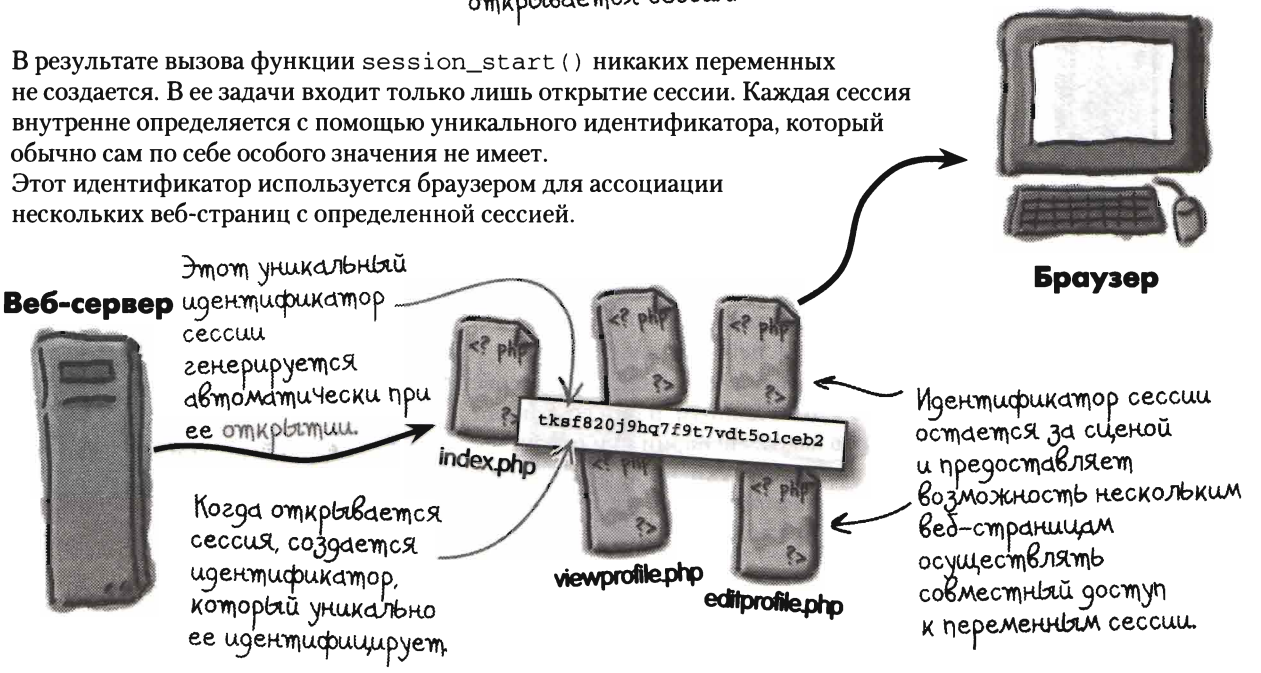
PHP-функция session_start() открывает сессию и дает вам возможность сохранять данные в ее переменных.

Вы можете открыть сессию, как только готовы к этому, вызовом PHP-функции `session_start()`.

```
session_start();
```

Вызовом этой PHP-функции открывается сессия.

В результате вызова функции `session_start()` никаких переменных не создается. В ее задачи входит только лишь открытие сессии. Каждая сессия внутренне определяется с помощью уникального идентификатора, который обычно сам по себе особого значения не имеет.
Этот идентификатор используется браузером для ассоциации нескольких веб-страниц с определенной сессией.



Идентификатор сессии продолжает существовать до тех пор, пока сессия не будет закрыта, что произойдет либо когда будет закрыт браузер, либо когда будет вызвана функция `session_destroy()`.

```
session_destroy();
```

Вызовом этой PHP-функции сессия закрывается.

PHP-функция session_destroy() закрывает сессию.

Закрытие сессии вызовом этой функции не приводит к автоматическому удалению созданных переменных этой сессии. Чтобы понять, почему так происходит, давайте рассмотрим подробнее, как сохраняются данные в сессии.

Использование данных сессии

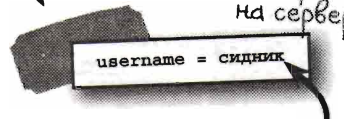
Использование сессий очень удобно тем, что они во многом похожи на куки. После того как вы открыли сессию вызовом функции `session_start()`, вы можете сохранять данные, например о входе в приложение «Несоответствия» (входные данные), в суперглобальном массиве `$_SESSION`.

```
$_SESSION['username'] = 'сидник';
```

Имя переменной сессии используется как индекс в суперглобальном массиве $_SESSION.

Значение переменной сессии, сохраненное в суперглобальном массиве $_SESSION.

Переменные сессии создаются и сохраняются на сервере.




```
echo('<p class="login">Вы вошли в приложение как ' . $_SESSION['username'] . '.</p>');
```

Для того чтобы получить доступ к переменной сессии, используйте элемент суперглобального массива $_SESSION с индексом имени переменной.

В отличие от куки для сохранения данных в переменных сессии не требуется вызова никакой специальной функции. Достаточно присвоить необходимое значение элементу суперглобального массива `$_SESSION`, выбрав для него в качестве индекса имя соответствующей переменной.

А как удаляются переменные сессии? При ее закрытии вызовом функции `session_destroy()` никакие переменные не удаляются, поэтому вы должны вручную удалить переменные вашей сессии, если надобность в них отпала, до того как пользователь закроет браузер (выход из приложения!). Быстрый и эффективный способ удалить все переменные сессии — это присвоить суперглобальному массиву `$_SESSION` значение пустого массива.

```
$_SESSION = array();
```

В результате выполнения этого кода все переменные текущей сессии удаляются.

Переменные сессии не удаляются автоматически при ее закрытии.

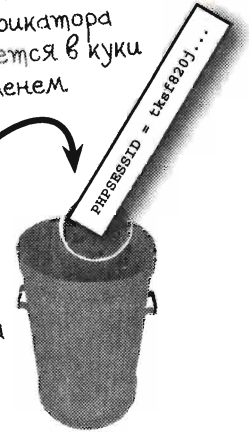
Но и это еще не все. Сессия может за сценой использовать куки. Если в браузере включена поддержка куки, сессия может создать куки, в котором будет сохранено значение идентификатора сессии. Поэтому, чтобы полностью закрыть сессию с использованием PHP-кода, вы должны также удалить куки, который мог быть автоматически создан сессией на клиентском компьютере для сохранения своего идентификатора. Как любой другой куки, вы удаляете его вызовом функции `setcookie()` с временем истечения срока действия в прошлом. Все, что вам необходимо знать для этого, — это имя куки, которое может быть определено вызовом функции `session_name()`.

Если сессия использует куки для сохранения ее идентификатора, то значение этого идентификатора сохраняется в куки с его именем.

```
if (isset($_COOKIE[session_name()])) {
  setcookie(session_name(), '', time() - 3600);
}
```

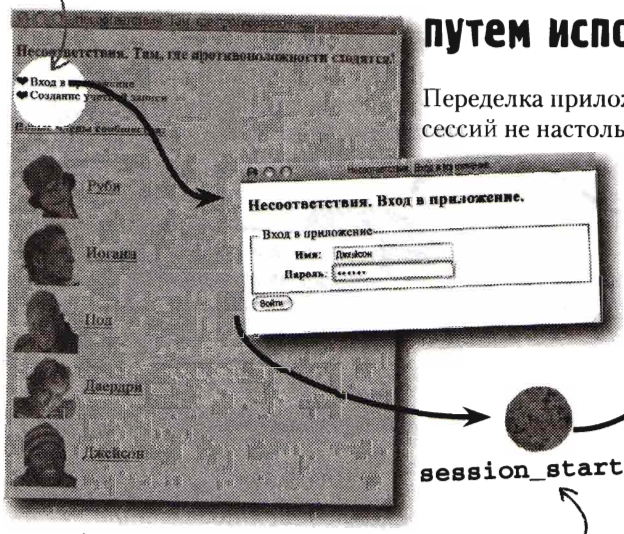
Вначале необходимо проверить, существует ли куки с именем идентификатора сессии.

Куки с именем идентификатора сессии удаляется установкой времени истечения срока действия в прошлом.



Обновление приложения «Несоответствия» путем использования сессий

Переделка приложения «Несоответствия» путем использования сессий не настолько драматична, как это может показаться на первый взгляд. Фактически общий ход процесса остается прежним, для вас добавляется только одна забота: не забыть вовремя открыть и закрыть сессию и удалить все данные, с ней связанные.



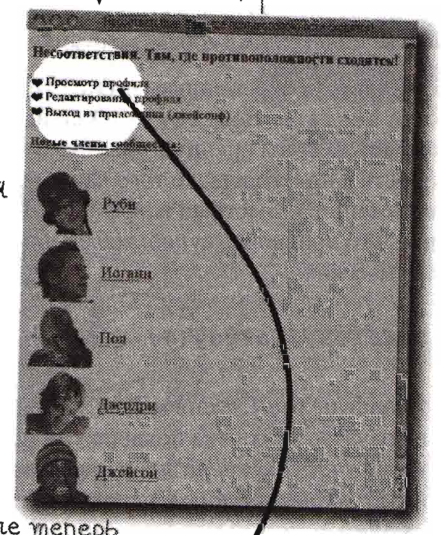
Отправная точка здесь!

session_start();

Функция session_start() начинает действие, открывая сессию.

Для сохранения идентификатора пользователя и его имени, необходимых для входа в приложение, было создано две переменных сессии.

Если в браузере включена поддержка куки, сервер создает одно для сохранения идентификатора сессии. В противном случае идентификатор сессии будет передаваться каждой странице в составе URL.

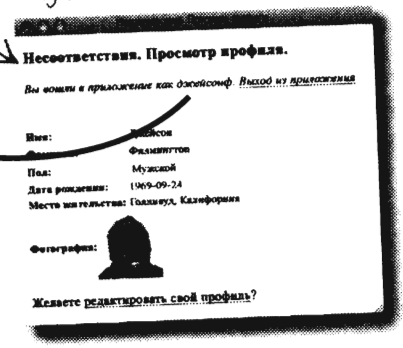


session_destroy();

В результате вызова функции session_destroy() сессия закрывается, предотвращая таким образом использование другими страницами данных, которые были в ней сохранены.

Если для хранения идентификатора сессии было создано куки, оно удаляется.

Входные данные теперь сохранены благодаря сессии, а не куки.



Данные сессии удаляются обнулением суперглобального массива $_SESSION.

Выход из приложения при использовании в нем сессии

Выход из приложения «Несоответствия» при использовании в нем сессии требует большего количества операций по сравнению с предыдущей версией, в которой использовались куки. Ниже перечислены этапы, которые необходимо пройти для того, чтобы пользователь успешно вышел из приложения «Несоответствия», в котором применяются сессии.

Вы не можете точно знать, использовалось ли куки для сохранения идентификатора сессии без такой проверки.

1. **Удаление переменных сессии.**
2. **Проверка существования куки с идентификатором сессии. Если такое куки существует — его удаление.**
3. **Закрытие сессии.**
4. **Переадресация браузера на главную страницу.**

Последний этап можно рассматривать как дополнительный, который, вообще говоря, является необязательным при выходе пользователя из приложения, но весьма полезным.

Время поработать

Сценарий «Выход из приложения» приложения «Несоответствия» подвергся реконструкции для того, чтобы в нем использовалась сессия вместо куки для сохранения входных данных. Напишите код, необходимый для «сессионизации» сценария «Выход из приложения», и прокомментируйте каждый его этап, имеющий к этому отношение.

```
<?php
  // Если пользователь вошел в приложение, удаление переменных сессии,
  // приводящее к выходу его из приложения
  session_start();
  if (.....)...
    // Удаление переменных сессии путем обнуления суперглобального массива $_SESSION
                ....

    // Удаление куки, содержащего идентификатор сессии,
    // установкой момента истечения срока действия
    // на час (3600 секунд) ранее текущего времени
    if (isset($_COOKIE[session_name()])) { ..... }

    // Закрытие сессии

  }

  // Переадресация на главную страницу
  $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
  header('Location: ' . $home_url);
?>
```

Решение задачи

Сценарий «Выход из приложения» приложения «Несоответствия» подвергся реконструкции для того, чтобы в нем использовалась сессия вместо куки для сохранения входных данных. Напишите код, необходимый для «сессионизации» сценария «Выход из приложения», и прокомментируйте каждый его этап, имеющий к этому отношение.

1. Удаление переменных сессии.
2. Проверка существования куки с идентификатором сессии. Если такое куки существует — его удаление.
3. Закрытие сессии.
4. Переадресация браузера на главную страницу.

Даже при выходе из приложения вы должны открыть сессию для того, чтобы получить доступ к ее переменным.

```
<?php
  // Если пользователь вошел в приложение, удаление переменных сессии,
  // приводящее к выходу его из приложения
  session_start();
  if (isset($_SESSION['user_id'])) {
    // Удаление переменных сессии путем обнуления суперглобального массива $_SESSION
    $_SESSION = array();                                                          (1)

    // Удаление куки, содержащего идентификатор сессии,
    // установкой момента истечения срока действия
    // на час (3600 секунд) ранее текущего времени
    if (isset($_COOKIE[session_name()])) { setcookie(session_name(), '', time() - 3600); }  (2)

    // Закрытие сессии
    session_destroy();                                                            (3)
  }
  // Переадресация на главную страницу
  $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
  header('Location: ' . $home_url);                                               (4)
?>
```

Теперь мы можем использовать переменную сессии вместо куки для проверки состояния пользователя.

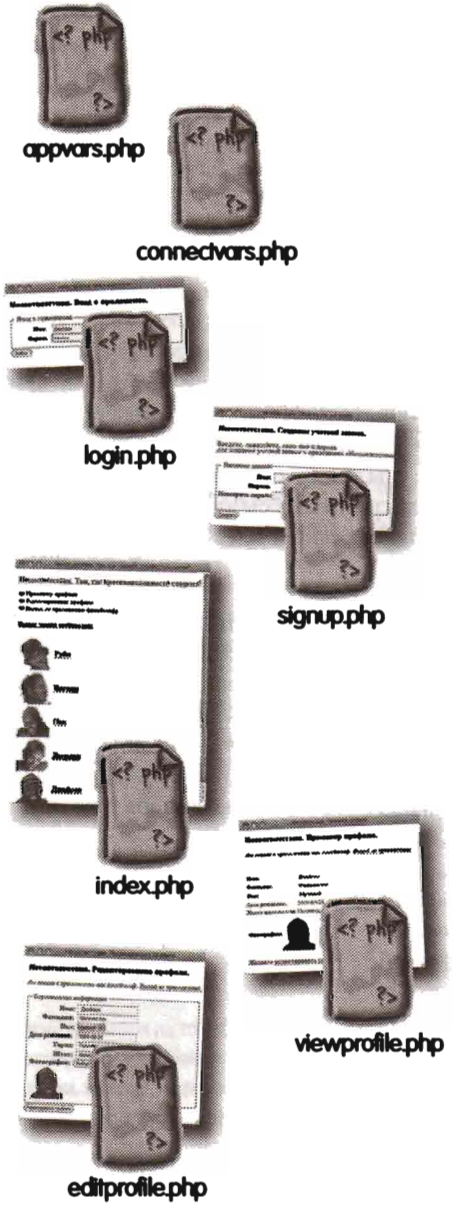
Для обнуления переменных сессии суперглобальному массиву $_SESSION присваивается значение пустого массива.

Закрытие сессии вызовом встроенной функции session_destroy().

Если куки, содержащий идентификатор сессии, существует, удаляем его, устанавливая момент истечения срока его действия на час ранее текущего времени.

Что и как изменять?

Переход от куки к сессиям затрагивает не только сценарий «Выход из приложения». Измените другие части приложения «Несоответствия» для поддержки сессий.



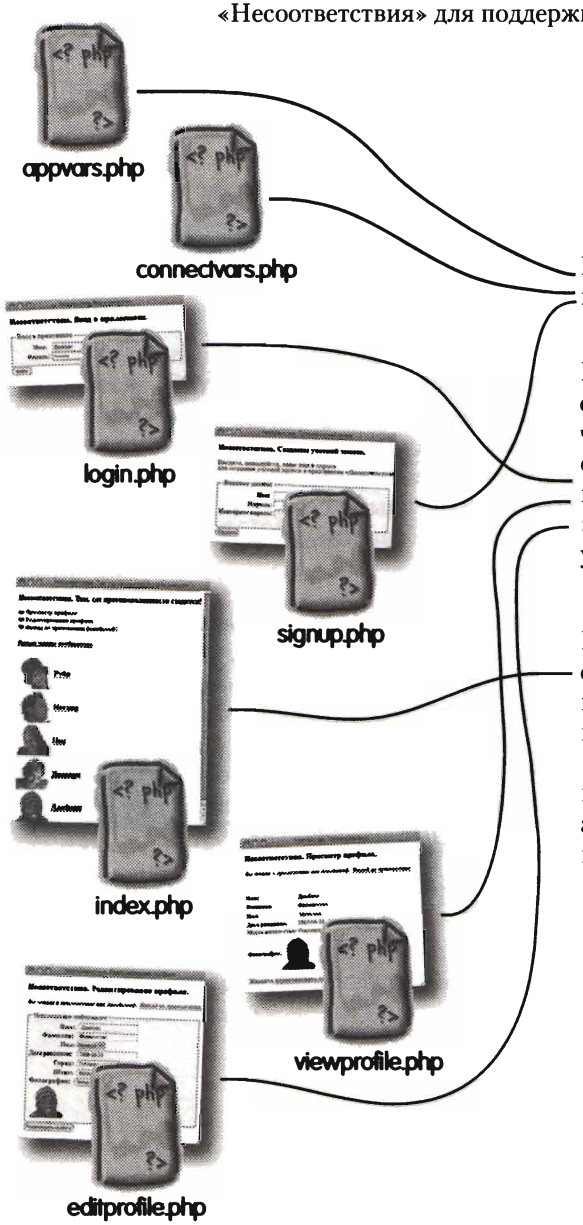
Никаких изменений, так как сценарий никак не зависит от входных данных.

Необходимы входные данные пользователя, сохраненные в переменных сессии, чтобы знать, что это за пользователь. Нужно добавить вызов функции `session_start()` для открытия сессии и заменить все ссылки на суперглобальный массив `$_COOKIE` аналогичными ссылками, только теперь уже — на суперглобальный массив `$_SESSIONS`.

Необходимы входные данные пользователя, сохраненные в переменных сессии, чтобы знать, навигационное меню какого состава выводить в браузере. Нужно добавить вызов функции `session_start()` для открытия сессии и заменить все ссылки на суперглобальный массив `$_COOKIE` аналогичными ссылками, только теперь уже — на суперглобальный массив `$_SESSIONS`.

Что и как изменять? Решение

Переход от куки к сессиям затрагивает не только сценарий «Выход из приложения». Измените другие части приложения «Несоответствия» для поддержки сессий.



Никаких изменений, так как сценарий никак не зависит от входных данных.

Необходимы входные данные пользователя, сохраненные в переменных сессии, чтобы знать, что это за пользователь. Нужно добавить вызов функции `session_start()` для открытия сессии и заменить все ссылки на суперглобальный массив `$_COOKIE` аналогичными ссылками, только теперь уже — на суперглобальный массив `$_SESSIONS`.

Необходимы входные данные пользователя, сохраненные в переменных сессии, чтобы знать, навигационное меню какого состава выводить в браузере. Нужно добавить вызов функции `session_start()` для открытия сессии и заменить все ссылки на суперглобальный массив `$_COOKIE` аналогичными ссылками, только теперь уже — на суперглобальный массив `$_SESSIONS`.

КЛЮЧЕВЫЕ МОМЕНТЫ

- HTTP-аутентификацию удобно использовать для ограничения доступа к отдельным страницам, но она не предоставляет хорошего способа для выхода из приложения.
- Куки позволяют вам сохранять некоторую информацию на клиентском компьютере (на котором запущен браузер), например входные данные пользователя.
- Все куки имеют срок действия, который может как простираться в далекое будущее, так и заканчиваться вместе с закрытием браузера.
- Для удаления куки вы просто устанавливаете срок их действия на любой момент времени в прошлом.
- Сессии предлагают аналогичные с куки возможности для сохранения данных, но находятся на сервере, и поэтому на них не влияют ограничения браузера, например отключение поддержки куки.
- У переменных сессий ограниченный срок действия, и они всегда удаляются, как только сессия закрывается, например когда закрывается браузер.

Не бывает глупых вопросов

В: Функция session_start() вызывается во многих различных местах, даже после того как сессия уже была открыта. Будет ли в результате таких многократных вызовов создаваться множество сессий?

О: Нет. В результате вызова функции `session_start()` не создается новая сессия, а происходит подключение к текущей. Поэтому, когда сценарий вызывает функцию `session_start()`, она вначале проверяет, существует ли идентификатор сессии, пытаясь узнать таким образом, имеется ли уже у приложения активная сессия. И только если такой сессии нет, она генерирует новый идентификатор и открывает новую сессию. Все последующие вызовы функции `session_start()` из этого же приложения определяют и используют текущую сессию вместо открытия новой.

В: Тогда как же сохраняется идентификатор сессии? Для этого сессии иногда используют куки?

О: Да. Даже несмотря на то что данные сессии сохраняются на сервере и находятся, таким образом, в более защищенном и не подверженном контролю со стороны браузера месте, должен существовать механизм, позволяющий сценарию получать доступ к текущей сессии.
Вот зачем нужен идентификатор сессии, который однозначно определяет ее и все данные, с ней связанные. Он должен существовать на клиенте так, чтобы множество страниц могли рассматривать себя как части одной и той же сессии. Один из способов выполнения этого требования предоставляют куки, сохраняя значение идентификатора, который затем используется для ассоциации любого сценария приложения с данной сессией.

В: Если сессии все равно зависят от куки, какие преимущества мы получаем, используя их вместо куки?

О: Сессии не зависят полностью от куки. Важно понимать, что использование куки служит для оптимизации процесса сохранения сессии и доступа к ее данным множества сценариев и не является необходимостью. Если поддержка куки отключена, идентификатор сессии передается от сценария к сценарию в составе URL подобно тому, как передаются данные при уже рассмотренном ранее запросе GET. Поэтому сессии могут вполне успешно выполнять свои задачи и без использования куки. Особенности реакции сессий при отсутствии поддержки куки контролируются в конфигурационном файле `php.ini` на сервере через установку значений переменных `session.use_cookies`, `session.use_only_cookies` и `session.use_trans_sid`.

В: Все равно, выглядит довольно странным то, сессии могут использовать куки, в то время как утверждается, что они выполняют свою задачу лучше, чем куки. Все ли здесь в порядке?

О: Хотя сессии действительно предоставляют некоторые ощутимые преимущества по сравнению с куки, было бы не правильно противопоставлять их друг другу в такой категорической форме, как или — или. Тот факт, что сессии сохраняются на сервере, а не на клиенте, является их большим преимуществом, так как это обеспечивает их большую защищенность и надежность. Поэтому, если у вас возникнет необходимость в постоянном существовании чувствительных данных, сессии будут лучшим выбором по сравнению с куки, так как предоставляют более высокий уровень защиты этих данных. Сессии также способны сохранять большие объемы данных по сравнению с куки. Это все — неоспоримые преимущества сессий перед куки независимо от того, включена поддержка куки или нет.
В приложении «Несоответствия» сессии предлагают удобное решение для сохранения входных данных на сервере. Для тех пользователей, у которых включена поддержка куки, сессии обеспечивают повышенный уровень защиты и надежности данных, используя куки как средство оптимизации процесса. А для тех пользователей, у которых поддержка куки отключена, сессии продолжают выполнять свои задачи, предавая при этом идентификатор в составе URL, полностью отказавшись от услуг куки.

Завершение перехода к сессиям

Хотя различные части приложения «Несоответствия», зависящие от сессии, используют ее для выполнения различных задач, все сценарии требуют внесения похожих изменений для перехода от использования куки к использованию сессии. Во-первых, всем им необходим вызов функции `session_start()` для того, чтобы активировать сессию. Кроме того, все эти преобразования требуют перехода от использования суперглобального массива `$_COOKIE` к использованию суперглобального массива `$_SESSION`, который отвечает за сохранение переменных сессии.

```
<?php
  session_start();
?>
```

Все сценарии, использующие сессии, начинаются с вызова функции session_start(), чтобы активировать сессию.

```
// Если пользователь еще не зашел в приложение, попытка войти
if (!isset($_SESSION['user_id'])) {
   if (isset($_POST['submit'])) {
      // Соединение с базой данных
      $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

      // Получение введенных пользователем данных для аутентификации
      $user_username = mysqli_real_escape_string($dbc, trim($_POST['username']));
      $user_password = mysqli_real_escape_string($dbc, trim($_POST['password']));

      if (!empty($user_username) && !empty($user_password)) {
         // Поиск имени пользователя и его пароля в базе данных
         $query = "SELECT user_id, username FROM mismatch_user WHERE username = '$user_username' AND
password = SHA('$user_password')";
         $data = mysqli_query($dbc, $query);

         if (mysqli_num_rows($data) == 1) {
            // Вход в приложение прошел успешно, присваиваем значения идентификатора пользователя и его
имени
            // переменным сессии (и куки) и переадресуем браузер на главную страницу
            $row = mysqli_fetch_array($data);
            $_SESSION['user_id'] = $row['user_id'];
            $_SESSION['username'] = $row['username'];
            setcookie('user_id', $row['user_id'], time() + (60 * 60 * 24 * 30));    // срок действия
истекает через 30 дней
            setcookie('username', $row['username'], time() + (60 * 60 * 24 * 30));  // срок действия
истекает через 30 дней
            $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
            header('Location: ' . $home_url);
         }
         else {
            // Имя пользователя/его пароль введены неверно, создание сообщения об ошибке
            $error_msg = 'Извините, для того, чтобы войти в приложение, вы должны ввести правильное имя
и пароль.';
         }
      }
```

login.php

Сценарий «Вход в приложение» использует сессию, чтобы запомнить идентификатор пользователя и его имя для обеспечения постоянного существования входных данных, и он решает эту задачу, полагаясь на использование суперглобального массива $_SESSION вместо суперглобального массива $_COOKIE.

```
// Создание меню навигации
if (isset($_SESSION['username'])) {
  echo '&#10084; <a href="viewprofile.php">Просмотр профиля </a><br />';
  echo '&#10084; <a href="editprofile.php">Редактирование профиля </a><br />';
  echo '&#10084; <a href="logout.php">Выход из приложения (' . $_
SESSION['username'] . ')</a>';
}
else {
  echo '&#10084; <a href="login.php">Вход в приложение</a><br />';
  echo '&#10084; <a href="signup.php">Создание учетной записи</a>';
}

...

// Прохождение в цикле массива данных пользователей,
// форматирование данных в виде HTML
echo '<h4>Новые члены сообщества:</h4>';
echo '<table>';
while ($row = mysqli_fetch_array($data)) {
  ...
  if (isset($_SESSION['user_id'])) {
    echo '<td><a href="viewprofile.php?user_id=' . $row['user_id'] . '">' .
$row['first_name'] . '</a></td></tr>';
  }
  else {
    echo '<td>' . $row['first_name'] . '</td></tr>';
  }
}
echo '</table>';
```

index.php

Главная страница «Несоответствия» для доступа к входным данным пользователя в процессе создания навигационного меню использует вместо суперглобального массива $_COOKIE суперглобальный массив $_SESSION и на базе этих данных решает, добавлять в меню гиперссылку на профили последних пользователей или нет.

Так же как сценарии главной страницы и «Вход в приложение», сценарий «Редактирование профиля» для доступа к входным данным пользователя теперь использует суперглобальный массив $_SESSION вместо суперглобального массива $_COOKIE.

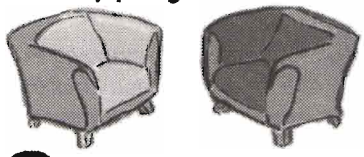
Хотя здесь это не показано, сценарий «Просмотр профиля» использует сессии, так же как и сценарий «Редактирование профиля».

```
// Проверка, вошел ли пользователь в приложение, прежде чем двигаться дальше
if (!isset($_SESSION['user_id'])) {
  echo '<p class="login">Пожалуйста, <a href="login.php">войдите в приложение </a>' .
    'для получения доступа к этой странице.</p>';
  exit();
}
else {
  echo('<p class="login">Вы вошли в приложение как ' . $_SESSION['username'] .
    '. <a href="logout.php">Выход из приложения</a>.</p>');
}

...

if (!empty($first_name) && !empty($last_name) && !empty($gender) &&
    !empty($birthdate) && !empty($city) && !empty($state)) {
  // Изображение выводится только в том случае, если имеется его файл
  if (!empty($new_picture)) {
    $query = "UPDATE mismatch_user SET first_name = '$first_name', last_name = '$last_name',
" .
      "gender = '$gender', birthdate = '$birthdate', city = '$city', state =
'$state', " .
      "picture = '$new_picture' WHERE user_id = '" . $_SESSION['user_id'] . "'";
  }
  else {
    $query = "UPDATE mismatch_user SET first_name = '$first_name', last_name = '$last_name',
" .
      "gender = '$gender', birthdate = '$birthdate', city = '$city', state =
'$state' " .
      "WHERE user_id = '" . $_SESSION['user_id'] . "'";
  }
  mysqli_query($dbc, $query);
```

viewprofile.php

editprofile.php

Беседы у камина



Сегодня вечером: **Куки и переменная сессии: откровенно о том, у кого память лучше**

Куки:

В последнее время среди нас, куки, ходят слухи по поводу того, что происходит на сервере. Говорят, вы хотите перебраться на нашу территорию и захватить нашу работу по сохранению данных. Что происходит?

Не вижу здесь никакого смысла. Клиент — отличное место для хранения данных, а я всего лишь парень, который выполняет эту работу.

Но это совсем другая ситуация! Если пользователь отключит мою поддержку, то совершенно очевидно, что нет никакой необходимости в сохранении данных.

Итак, насколько я понимаю, ты предлагаешь сохранять данные на сервере? Как удобно.

Ладно, Эйнштейн. Раз ты так великолепно все это рассчитал, почему ты все-таки пользуешься моими услугами по сохранению своего маленького драгоценного идентификатора на клиенте?

Переменная сессии:

Погоди, «захватить» — слишком громко сказано. Иногда просто имеет смысл сохранять данные на сервере.

А что ты будешь делать, если пользователь отключит твою поддержку?

Вот уж нет. Пользователь часто даже и не подозревает, что веб-приложению требуется сохранять данные, просто потому что в большинстве случаев речь идет о данных, действующих за сценой, например об идентификаторе пользователя. И вот тогда, если твоя поддержка отключена, пользователь остается ни с чем.

Вот именно. И здесь очень важно то, что у пользователя нет никакой возможности изменить конфигурацию сервера, поэтому тебе нечего беспокоиться по поводу того, могут данные в данный момент сохраняться или нет.

Э-э-э... ну, на самом деле большинство об этом ничего не знает, поэтому нет особой нужды развивать эту тему дальше. Мы можем поговорить об этом в частном порядке. Самое главное, что я все время здесь, в полной готовности сохранять данные на сервере.

Куки:

Вот уж нет, давай-ка, скажи, насколько ты нуждаешься в моей помощи!

О, я знаю, ты можешь, но правда заключается в том, что ты предпочитаешь не делать этого. И, возможно, где-то в глубине души ты чувствуешь ко мне что-то вроде расположения.

Ах вот как! Значит, ты защищаешься тем, что придираешься к размерам маленького парнишки. Пусть так. Мне, может, и не под силу сохранять такой объем информации, который можешь сохранять ты, и я признаю, что существование на клиентской территории делает меня менее защищенным. Но это только мобилизует! Кроме того, у меня есть кое-что, о чем ты можешь только мечтать.

Так вот. Все эти огромные объемы для хранения данных и защищенность, которыми ты так гордишься, достигнуты ценой... короткой жизни. Я не хотел быть тем, кто должен сказать тебе это, но все твое существование полностью зависит от длительности единственной сессии браузера. Мне кажется, отсюда происходит и твое имя.

Очень просто. Я не умираю вместе сессией, а прекращаю свое существование исключительно в заданный срок. Поэтому мне может быть назначен достаточно долгий срок жизни, значительно превосходящий каприз какого-нибудь веб-серфера, приходящего в восторг от каждого щелчка мышью, думающего, что он тут самый крутой, и открывающего и закрывающего браузер при первой же возможности.

Тут, правда, есть одна проблема. Тот же самый писатель часто устанавливает срок моей жизни на такой короткий период, что я не в состоянии прожить ту долгую жизнь, которую я заслуживаю. Я имею в виду то, что я...

Переменная сессии:

Ладно, я признаю: время от времени я действительно соглашаюсь с тем, чтобы принять небольшую помощь от тебя в вопросах сохранения данных, совместно используемых несколькими страницами. Но при необходимости я могу обойтись и без твоей помощи.

Послушай, я ничего не имею против тебя. Мне бы просто хотелось, чтобы ты был немного более защищенным. Кроме того, у тебя такие серьезные ограничения в размерах... Знаешь, не все постоянно живущие данные имеют размер в один байт.

И что же это? Ну-ка, скажи.

Ты хочешь сказать, что можешь существовать во времени за пределами одной сессии? Как такое возможно?

Ого! Что за чувство — ощущать бессмертие? У меня остается единственная надежда на то, что какой-нибудь небрежный писатель сценариев случайно забудет удалить меня при закрытии сессии... да вот только браузер все равно удалит меня при своем закрытии.

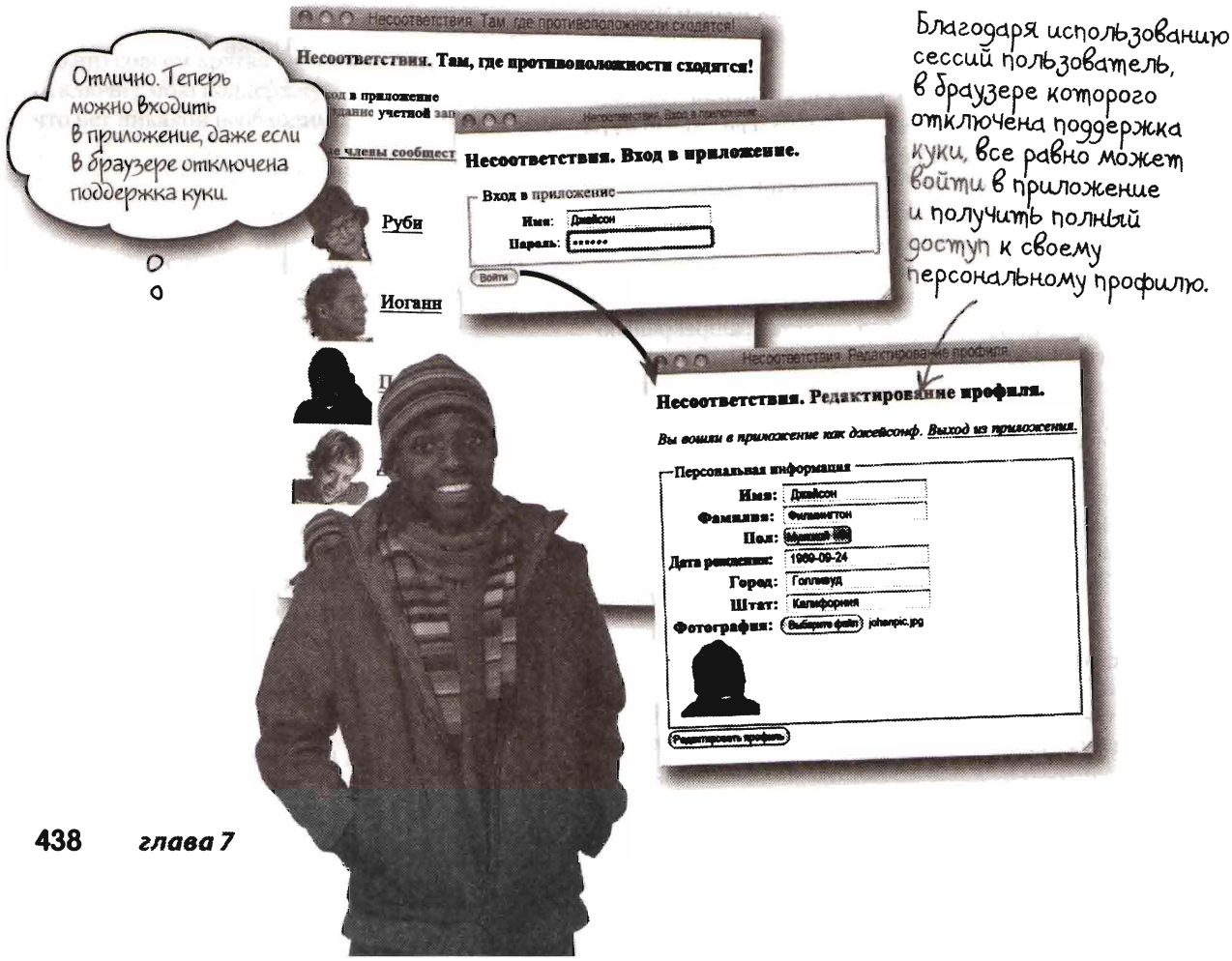
Эй! Ты где?.. Да, истечение срока действия — штука жесткая.

Тест-драйв

Внесите в приложение «Несоответствия» изменения, обеспечивающие использование сессий вместо куки.

Внесите изменения в сценарии приложения «Несоответствия» так, чтобы для обеспечения постоянного существования данных входа в приложение в них использовались сессии вместо куки (или загрузите их с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Переход на использование сессий вместо куки требует внесения изменений в сценарии `index.php`, `login.php`, `logout.php`, `editprofile.php` и `viewprofile.php` и связан, в первую очередь, с включением функции `session_start()` в самое начало каждого из сценариев. Кроме того, необходимо внести изменения, обеспечивающие сохранение входных данных в суперглобальном массиве `$_SESSION` вместо `$_COOKIE`.

Загрузите все сценарии на ваш веб-сервер и откройте главную страницу приложения «Несоответствия» (`index.php`) в браузере. Попробуйте войти в приложение и выйти из него, чтобы убедиться в том, что все работает как прежде. Вы не должны заметить никакой разницы в работе приложения до и после внесения изменений, если только в вашем браузере раньше не была отключена поддержка куки.



Запомни!

Если значения переменных в файле конфигурации PHP php.ini на вашем сервере установлены неправильно, при отключенной поддержке куки сессии могут работать неправильно.

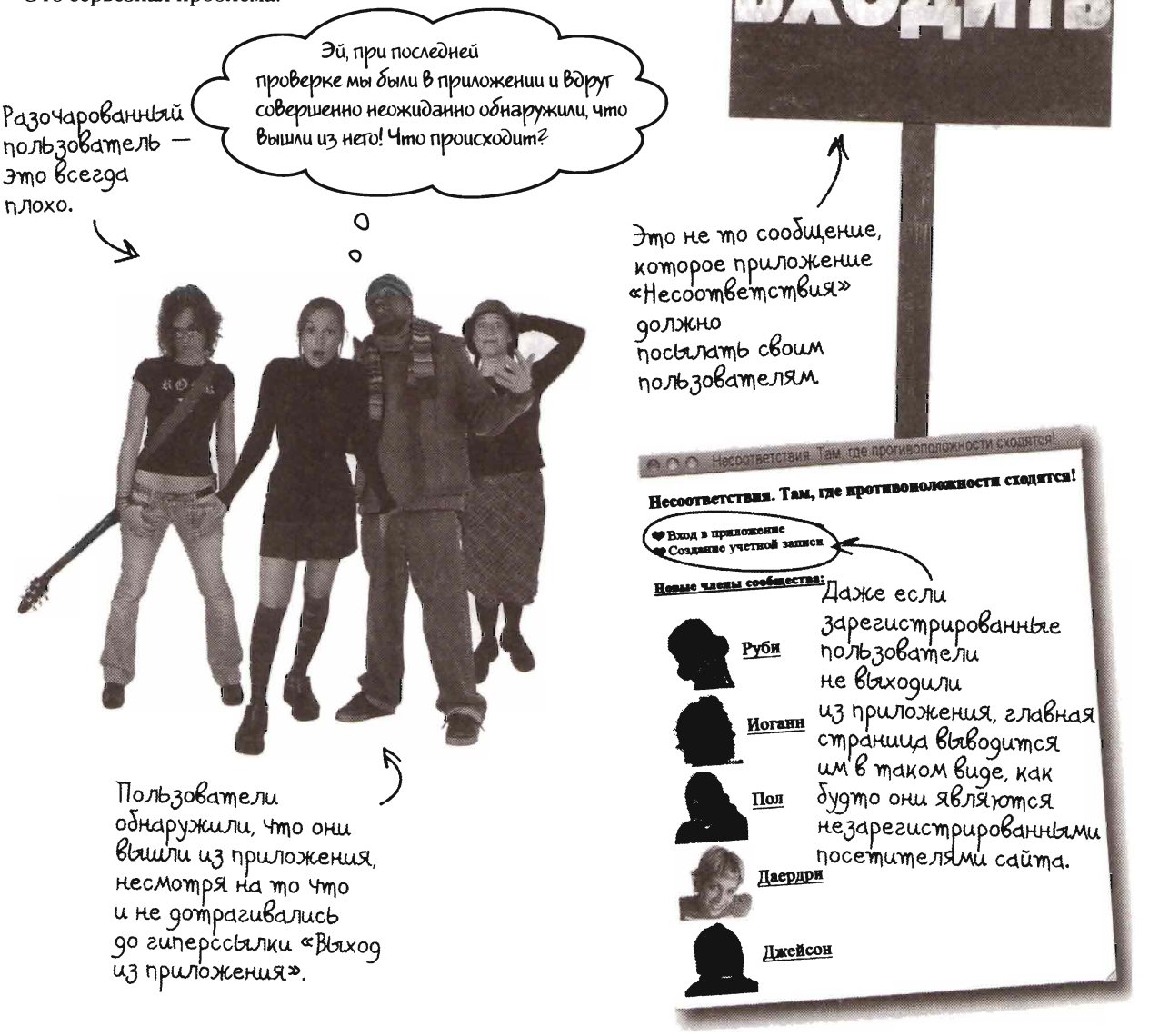
Для того чтобы при отключенной поддержке куки сессии работали правильно, должен быть предусмотрен иной механизм передачи идентификатора сессии между различными сценариями приложения. Он предусматривает включение идентификатора сессии в состав URL каждого сценария. Это происходит автоматически, если значение переменной `session.use_trans_id` *в файле* `php.ini` *на сервере установлено равным* `1` *(*`true`*). Если у вас нет возможности вносить изменения в этот файл, а поддержка куки отключена, вы должны вручную добавить идентификатор сессии к URL каждого сценария, как показано ниже:*

```
<a href="viewprofile.php?<?php echo SID; ?>">Просмотр
профиля</a>
```

Суперглобальная переменная SID содержит идентификатор сессии, который передается в составе URL, чтобы сценарий «Просмотр профиля» смог получить доступ к данным сессии.

Пользователи не чувствуют гостеприимства

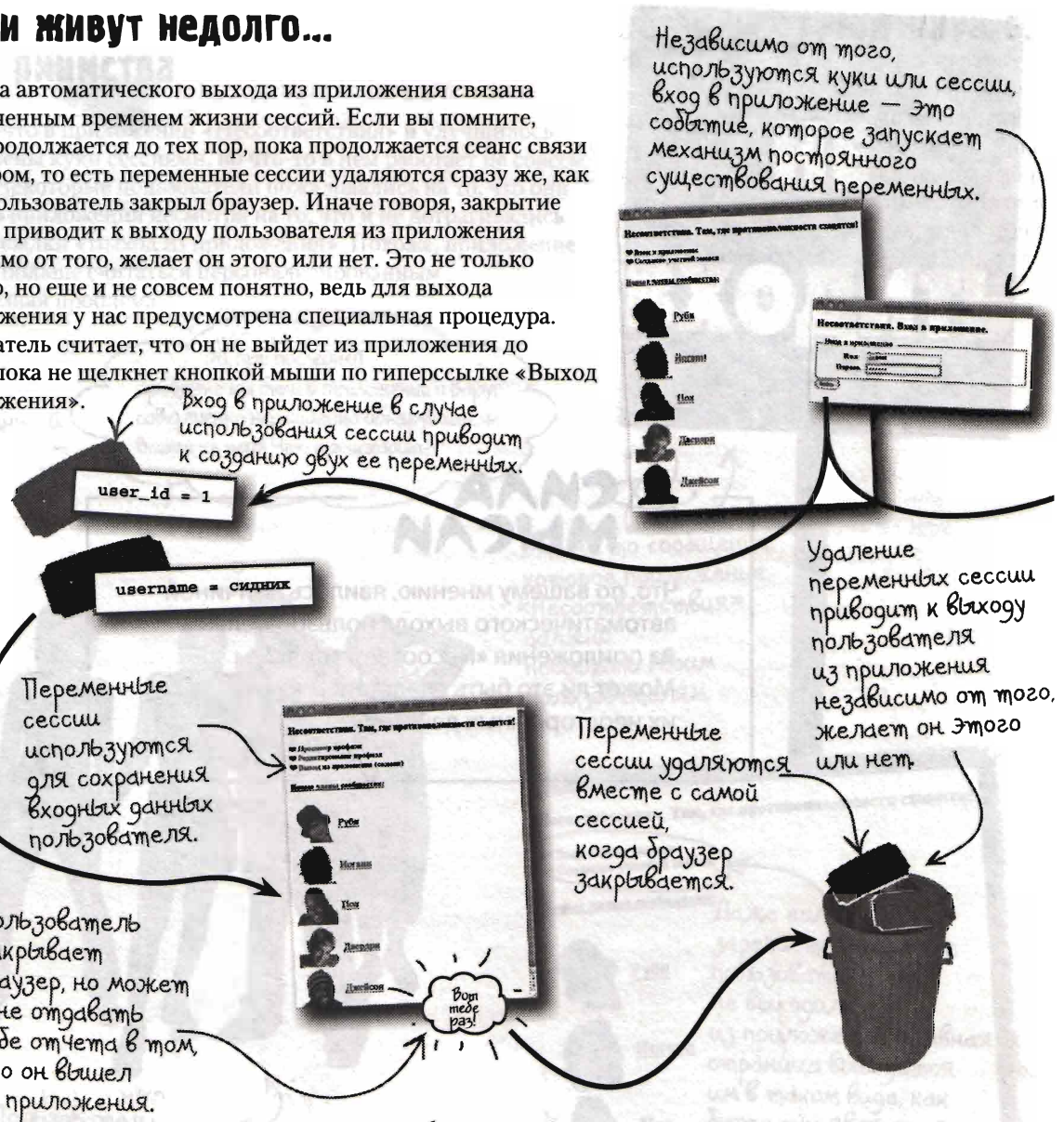
Хотя кое-что в приложении «Несоответствия» и улучшилось после замены куки сессиями, но что-то в нем работает не совсем хорошо. Некоторые пользователи пожаловались на то, что они вышли из приложения несмотря на то, что и не дотрагивались до гиперссылки «Выход из приложения». Похоже, приложение не может больше считаться персонализированным... Это серьезная проблема.



Что, по вашему мнению, явилось причиной автоматического выхода пользователей из приложения «Несоответствия»? Может ли это быть следствием каких-то их неосторожных действий?

Сессии живут недолго...

Проблема автоматического выхода из приложения связана с ограниченным временем жизни сессий. Если вы помните, сессия продолжается до тех пор, пока продолжается сеанс связи с браузером, то есть переменные сессии удаляются сразу же, как только пользователь закрыл браузер. Иначе говоря, закрытие браузера приводит к выходу пользователя из приложения независимо от того, желает он этого или нет. Это не только неудобно, но еще и не совсем понятно, ведь для выхода из приложения у нас предусмотрена специальная процедура. Пользователь считает, что он не выйдет из приложения до тех пор, пока не щелкнет кнопкой мыши по гиперссылке «Выход из приложения».



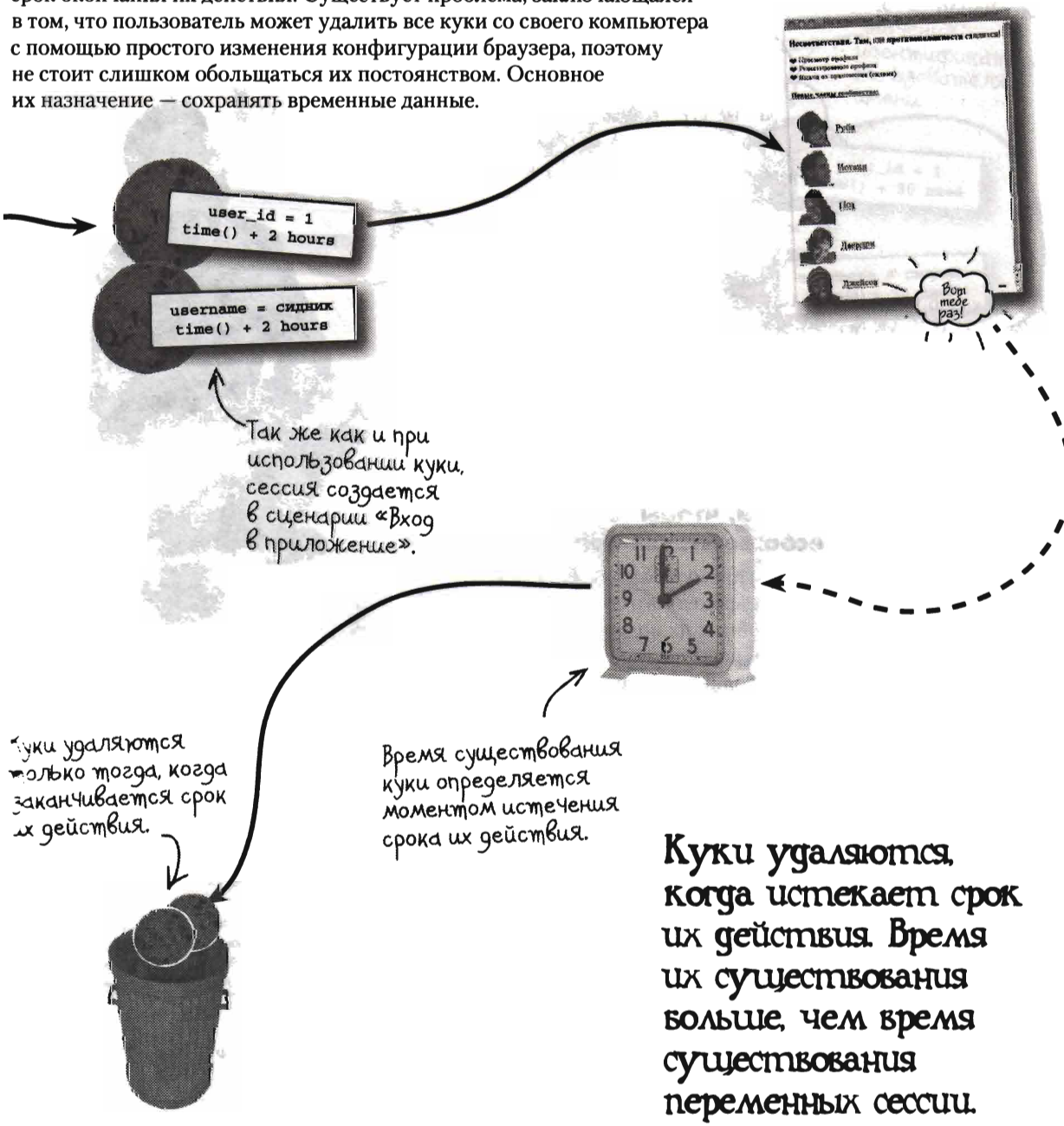
Хотя можно удалить сессию после того, как надобность в ней отпадет, вы не можете оставить ее после того, как прекратили сеанс связи браузера с сервером. Поэтому сессии имеют более короткий срок действия, чем куки, для которых, как мы знаем, этот срок может исчисляться часами, днями, месяцами и даже годами. Означает ли это, что сессии уступают куки в вопросах функциональности? Ни в коем случае. Но это действительно означает, что вы можете столкнуться с проблемами при использовании сессий для сохранения информации, время существования которой простирается за пределы сеанса связи браузера с сервером. Например, для сохранения входных данных!

Переменные сессии удаляются, когда пользователь прекращает сеанс связи с сервером, закрывая браузер.

...а куки могут существовать вечно!

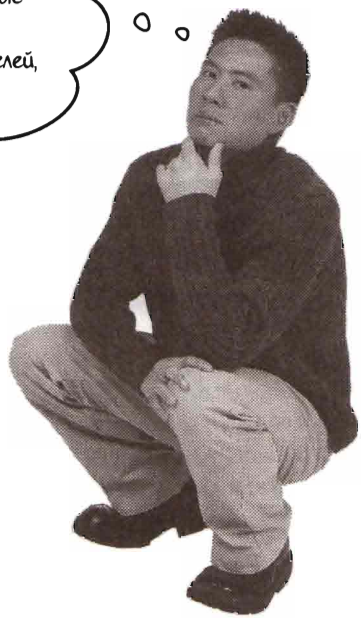
Ну, может, и не вечно, но достаточно долго, чтобы пережить любую сессию.

В отличие от переменных сессии срок действия куки не зависит от сеанса связи браузера с сервером, поэтому куки могут продолжать и продолжать свое существование до тех пор, пока не будет достигнут срок окончания их действия. Существует проблема, заключающаяся в том, что пользователь может удалить все куки со своего компьютера с помощью простого изменения конфигурации браузера, поэтому не стоит слишком обольщаться их постоянством. Основное их назначение — сохранять временные данные.



Куки удаляются, когда истекает срок их действия. Время их существования больше, чем время существования переменных сессии.

А может, имеет смысл использовать и сессии, и куки? При этом куки позволили бы пользователю сохранять входные данные на более длительный период времени. Это бы хорошо работало для тех пользователей, у которых включена поддержка куки.



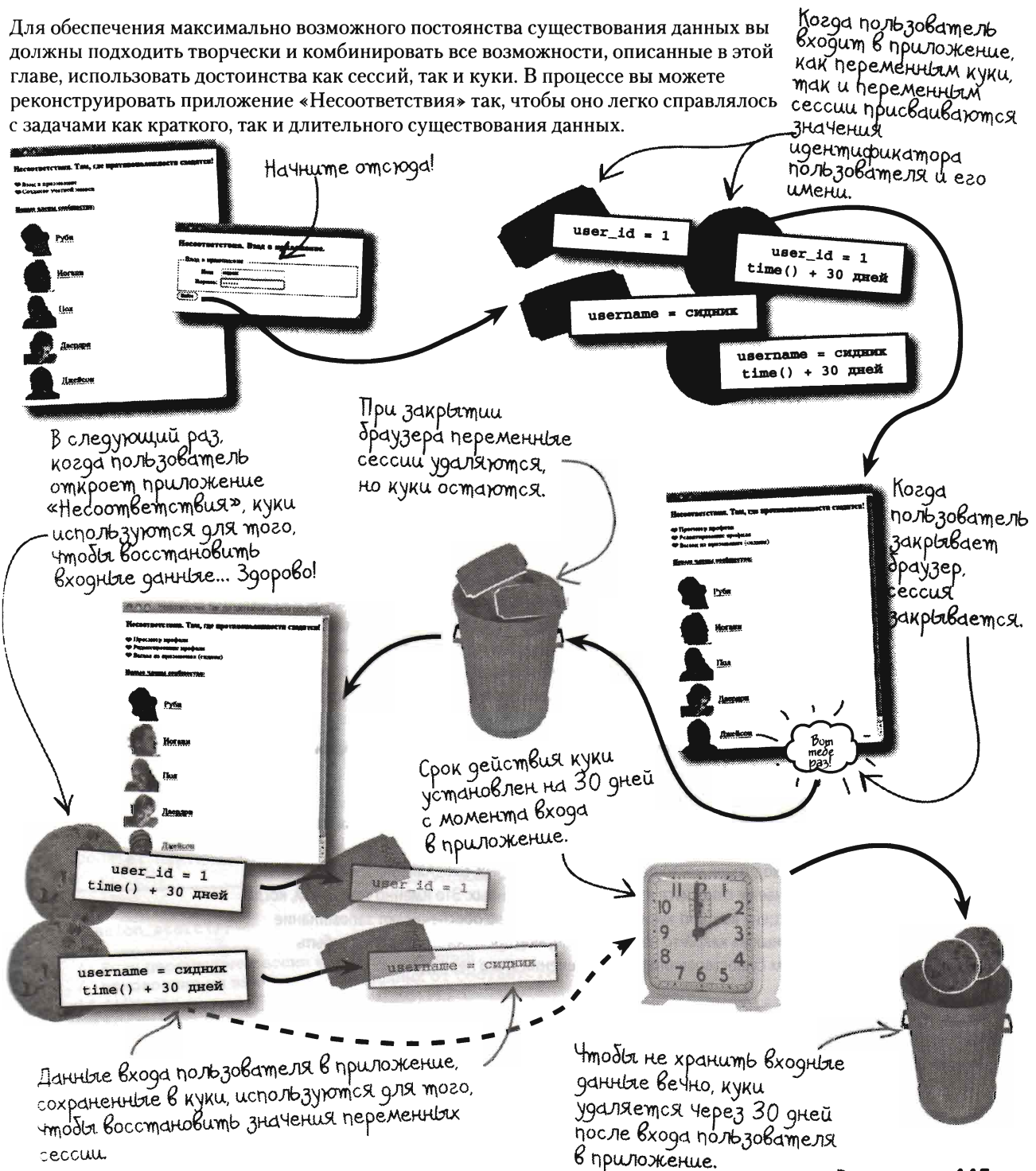
До тех пор пока вы не имеете дела с высоко чувствительными данными, использование куки имеет определенные преимущества по сравнению с использованием сессий.

Действительно. Нет ничего плохого в использовании отдельных достоинств как сессий, так и куки, чтобы сделать приложение «Несоответствия» более гибким.

Фактически здесь мы имеем прямую выгоду. Сессии больше подходят для кратковременного сохранения совместно используемых данных: их поддержка шире, так как не ограничивается конфигурацией браузера. С другой стороны, куки позволяют сохранять эти данные в течение значительно более длительного времени. Конечно, не все пользователи смогут воспользоваться достоинствами куки, но все же таких пользователей достаточно много. Все, что вы можете сделать, чтобы удовлетворить нужды значительной части ваших пользователей не за счет другой ее части, пойдет вам только на пользу.

Сессии + куки = исключительное постоянство существования данных

Для обеспечения максимально возможного постоянства существования данных вы должны подходить творчески и комбинировать все возможности, описанные в этой главе, использовать достоинства как сессий, так и куки. В процессе вы можете реконструировать приложение «Несоответствия» так, чтобы оно легко справлялось с задачами как краткого, так и длительного существования данных.



не бывает глупых вопросов

В: **Можно ли считать, что разница в сроках действия между куки и сессиями определяет их выбор?**

О: Нет. Так случилось, что эта разница сыграла значительную роль в разработке приложения «Несоответствия», но у каждого приложения есть свои особенности, и существуют другие аспекты куки и сессий, которые необходимо учитывать в каждом конкретном случае. Например, данные, сохраненные в сессии, защищены лучше, чем те же данные, сохраненные в куки. Поэтому, если поддержка куки включена и они используются исключительно для сохранения идентификатора сессии, остальные данные, сохраненные в сессии, защищены значительно лучше, чем если бы они были сохранены непосредственно в куки. Причина здесь в том, что данные сессии сохраняются на сервере, добиться несанкционированного доступа к данным которого значительно сложнее, чем к данным клиента. Поэтому, если вы имеете дело с данными, которые требуют серьезной защиты, использование сессий предпочтительнее использования куки.

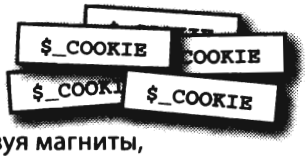
В: **А что по поводу размеров данных? Это играет роль?**

О: Да. Размер данных также играет свою роль. Сессии способны сохранять данные большего размера, чем куки, и это также причина отдавать предпочтение сессиям, если вам необходимо сохранять данные размером, превышающим длину нескольких простых строк текста. Конечно, база данных еще лучше подходит для сохранения больших объемов данных, поэтому будьте внимательны и не сильно увлекайтесь сессиями в подобных ситуациях.

В: **Тогда какие же преимущества у куки и сессий перед базами данных MySQL?**

О: Удобство. Для сохранения данных в базе требуется приложить значительные усилия. Кроме того, не забывайте, что базы данных идеально подходят для постоянного хранения данных. Входная информация в общем случае не относится к такому разряду данных. Это именно тот случай, когда на сцену выходят куки и сессии. Они лучше обеспечивают запоминание данных на какой-то небольшой период времени и позволяют забыть об их существовании после истечения этого периода.

Магниты PHP



В сценарии «Создание учетной записи» приложения «Несоответствия» используется своя собственная форма для предоставления пользователю возможности ввести выбранное им имя и пароль. Проблема в том, что код сценария неполон. Используя магниты, расположенные ниже, завершите код сценария так, чтобы новый пользователь мог создать учетную запись и присоединиться к сообществу «Несоответствия».

```
...
if (mysqli_num_rows($data) == 1) {
  // Вход в приложение прошел успешно, присваиваем значения идентификатора пользователя и его имени
  // переменным сессии (и куки) и переадресуем браузер на главную страницу
  $row = mysqli_fetch_array($data);
  .....['user_id'] = $row['user_id'];
  .....['username'] = $row['username'];
  setcookie('user_id', $row['user_id'], time() + (60 * 60 * 24 * 30));
// срок действия истекает через 30 дней
  setcookie('username', $row['username'], time() + (60 * 60 * 24 * 30));
// срок действия истекает через 30 дней
  $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
  header('Location: ' . $home_url);
}
...
```

login.php

```
<?php
// Если пользователь вошел в приложение, удаление переменных сессии для того,
// чтобы он вышел из приложения
session_start();
if (isset(.....['user_id'])) {
  // Удаление переменных сессии путем присвоения суперглобальному массиву
  // $_SESSION значения пустого массива
  .....= array();

  // Удаление куки, содержащего идентификатор сессии, путем установки
  // срока его действия на час (3600 секунд) ранее текущего времени
  if (isset(..... [session_name()])) {
    setcookie(session_name(), '', time() - 3600);    }

  // Удаление сессии
  session_destroy();
}

// Удаление куки, содержащих идентификатор пользователя и его имя,
// путем установки срока их действия на час (3600 секунд) ранее текущего времени
setcookie('user_id', '', time() - 3600);
setcookie('username', '', time() - 3600);
...
```

logout.php


```
<?php
  session_start();

  // Если переменные сессии не имеют значений,
  // присвоение им значений, полученных от соответствующих куки
  if (!isset(..... ['user_id'])) {
    if (isset(..... ['user_id']) && isset(..... ['username']))
{     .....
      ..... ['user_id'] = ..... ['user_id'];
      ..... ['username'] = ..... ['username'];

    }
  }
?>
...
```

index.php

Магниты PHP

В сценарии «Создание учетной записи» приложения «Несоответствия» используется своя собственная форма для предоставления пользователю возможности ввести выбранное им имя и пароль. Проблема в том, что код сценария неполон. Используя магниты, расположенные ниже, завершите код сценария так, чтобы новый пользователь мог создать учетную запись и присоединиться к сообществу «Несоответствия».

```
...
if (mysqli_num_rows($data) == 1) {
  // Вход в приложение прошел успешно, присваиваем значения идентификатора пользователя и его имени
  // переменным сессии (и куки) и переадресуем браузер на главную страницу
  $row = mysqli_fetch_array($data);
  $_SESSION['user_id'] = $row['user_id'];
  $_SESSION['username'] = $row['username'];
  setcookie('user_id', $row['user_id'], time() + (60 * 60 * 24 * 30));
// срок действия истекает через 30 дней
  setcookie('username', $row['username'], time() + (60 * 60 * 24 * 30));
// срок действия истекает через 30 дней
  $home_url = 'http://' . $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
  header('Location: ' . $home_url);
}
...
```

В дополнение к переменным сессии созданы новые куки.

login.php

```
<?php
// Если пользователь вошел в приложение, удаление переменных сессии для того,
// чтобы он вышел из приложения
session_start();
if (isset($_SESSION['user_id'])) {
  // Удаление переменных сессии путем присвоения суперглобальному массиву
  // $_SESSION значения пустого массива
  $_SESSION = array();

  // Удаление куки, содержащего идентификатор сессии, путем установки
  // срока его действия на час (3600 секунд) ранее текущего времени
  if (isset($_COOKIE[session_name()])) {
    setcookie(session_name(), '', time() - 3600);
  }

  // Удаление сессии
  session_destroy();
}

// Удаление куки, содержащих идентификатор пользователя и его имя,
// путем установки срока их действия на час (3600 секунд) ранее текущего времени
setcookie('user_id', '', time() - 3600);
setcookie('username', '', time() - 3600);
...
```

При выходе из приложения теперь требуется удалять как куки, содержащие идентификатор сессии, так и вновь созданные куки.

logout.php

```
<?php
session_start();

// Если переменные сессии не имеют значений,
// присвоение им значений, полученных от соответствующих куки
if (!isset($_SESSION['user_id'])) {
  if (isset($_COOKIE['user_id']) && isset($_COOKIE['username'])) {
    $_SESSION['user_id'] = $_COOKIE['user_id'];
    $_SESSION['username'] = $_COOKIE['username'];
  }
}
?>
...
```

Если пользователь вошел в приложение не путем открытия новой сессии, необходимо проверить, существуют ли куки.

Точно такой же код с использованием куки и сессии должен быть в сценариях editprofile.php и viewprofile.php.

Присвоение значений переменным сессии с использованием куки.

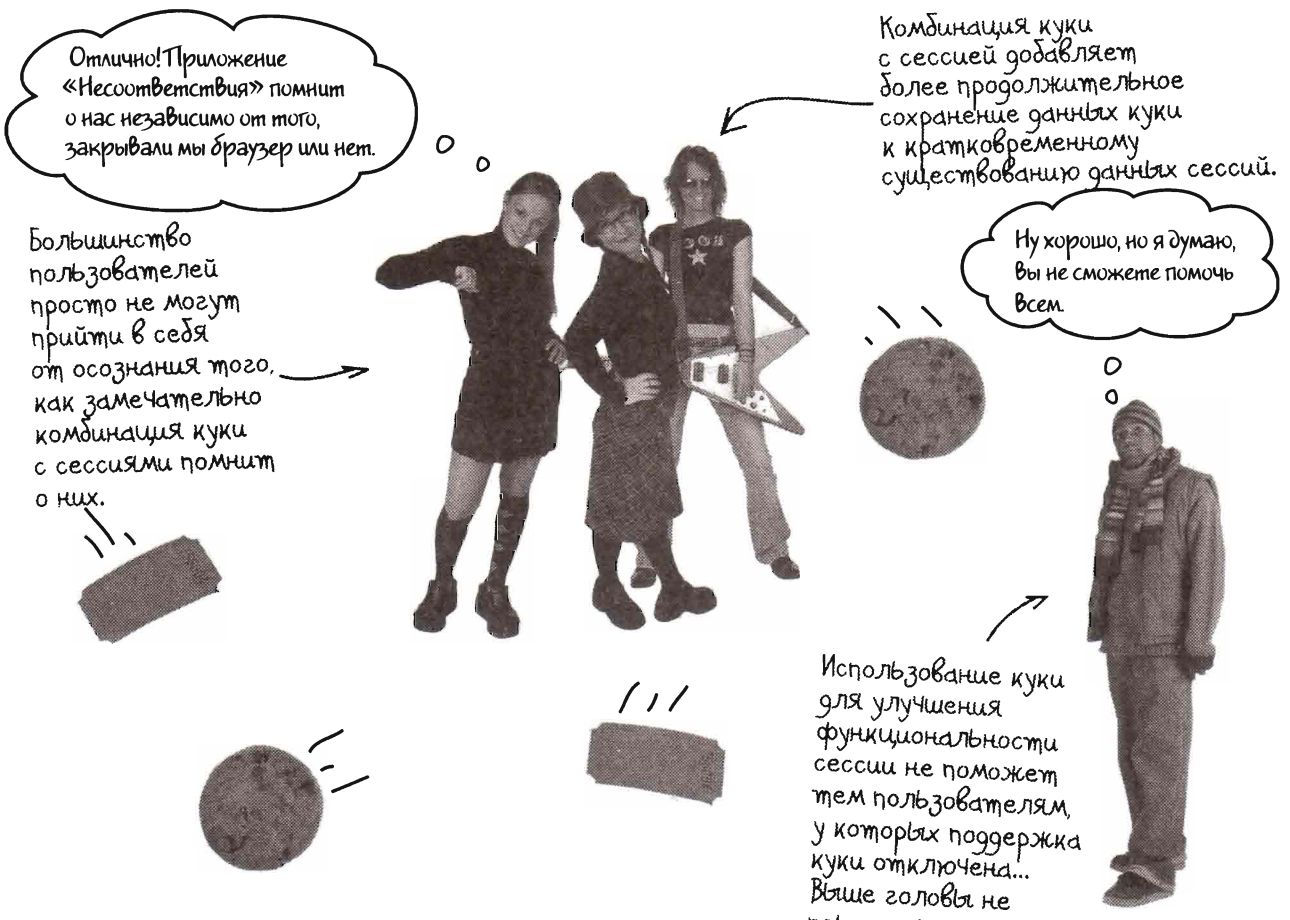
index.php

Тест-драйв

Внесите в приложение «Несоответствия» изменения, обеспечивающие использование и сессий, и куки.

Внесите изменения в сценарии приложения «Несоответствия» так, чтобы в них использовались и сессии, и куки для обеспечения постоянного существования входных данных (или загрузите их с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Переход на использование и сессий, и куки требует внесения изменений в сценарии `index.php`, `login.php`, `logout.php`, `editprofile.php` и `viewprofile.php`.

Загрузите все сценарии на ваш веб-сервер и откройте главную страницу приложения «Несоответствия» (`index.php`) в браузере. Попробуйте войти в приложение и затем выйти из него, в результате чего все переменные сессии будут удалены. Откройте главную страницу заново и проверьте, остались ли вы в приложении. Куки сделали это возможным, так как продолжили свое существование и после того, как браузер был закрыт.



Ваш инструментарий PHP и MySQL

Вы узнали много нового о построении систем управления пользователями как части приложения «Несоответствия». Давайте выделим наиболее важные понятия.

setcookie()

Встроенная PHP-функция, используемая для создания куки на клиенте. Для вновь создаваемого куки может быть определен срок действия, по истечении которого он будет удален. Если такой срок не определен, куки будет удален при закрытии браузера.

$_COOKIE

Встроенная суперглобальная переменная PHP, используемая для доступа к данным куки. Эта переменная является массивом, и данные куки представлены как его элементы. Поэтому для доступа к определенным данным необходимо использовать имя соответствующего куки как индекс в этом массиве.

SHA(*значение*)

Эта MySQL-функция зашифровывает фрагмент текста, в результате чего образуется строка, состоящая из 40 шестнадцатеричных цифр. Функция предоставляет очень удобный способ шифрования данных, которые должны храниться в не распознаваемом для посторонних виде в базе данных. Необходимо принимать во внимание, что эта функция использует однонаправленный алгоритм, то есть не существует функции decrypt (расшифровать).

session_start()

Встроенная PHP-функция, используемая для открытия новой сессии или активации уже открытой. Вы должны вызвать эту функцию до того, как получите доступ к любой переменной сессии.

$_SESSION

Встроенная суперглобальная переменная PHP, используемая для доступа к данным сессии. Эта переменная является массивом, и данные сессии представлены как его элементы. Поэтому для доступа к определенным данным необходимо использовать имя соответствующей переменной как индекс в этом массиве.

session_destroy()

Эта встроенная PHP-функция, используемая для закрытия сессии, должна вызываться, когда вы закончили работать с определенной сессией. Необходимо принимать во внимание, что эта функция не удаляет данные сессии, поэтому очень важно удалить их вручную присвоением суперглобальному массиву $_SESSION значения пустого массива.

Несколько фрагментов кода были извлечены из приложения «Несоответствия», а мы никак не можем вспомнить, что каждый из них делает. Проведите линии, связывающие эти фрагменты с описаниями, что каждый из них должен делать.

PHP/MySQL-код	Описание
`empty($_COOKIE['user_id'])`	Использует переменную сессии для того, чтобы определить, вошел пользователь в приложение или нет.
`setcookie(session_name(), '', time() - 3600);`	Использует куки для того, чтобы определить, вошел пользователь в приложение или нет.
`SHA('$user_password')`	Удаляет куки, содержащий идентификатор сессии, установкой срока действия на один час раньше текущего времени.
`session_destroy()`	Зашифровывает пароль пользователя в форму, недоступную для расшифровки.
`setcookie('user_id', $row['user_id'])`	Сохраняет в куки идентификатор пользователя.
`$_SESSION = array()`	Открывает сессию.
`session_start()`	Закрывает текущую сессию.
`isset($_SESSION['user_id'])`	Удаляет все переменные сессии.

Несколько фрагментов кода были извлечены из приложения «Несоответствия», а мы никак не можем вспомнить, что каждый из них делает. Проведите линии, связывающие эти фрагменты с описаниями, что каждый из них должен делать.

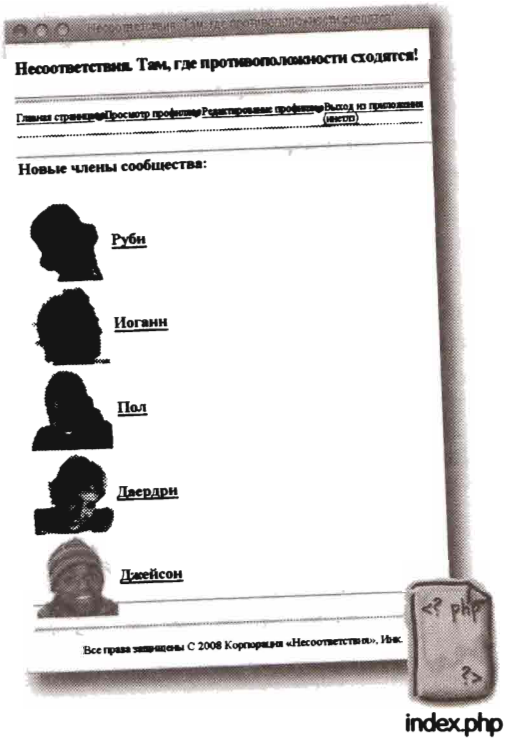
PHP/MySQL-код	Описание
`empty($_COOKIE['user_id'])`	Использует переменную сессии для того, чтобы определить, вошел пользователь в приложение или нет.
`setcookie(session_name(), '', time() - 3600);`	Использует куки для того, чтобы определить, вошел пользователь в приложение или нет.
`SHA('$user_password')`	Удаляет куки, содержащий идентификатор сессии, установкой срока действия на один час раньше текущего времени.
`session_destroy()`	Зашифровывает пароль пользователя в форму, недоступную для расшифровки.
`setcookie('user_id', $row['user_id'])`	Сохраняет в куки идентификатор пользователя.
`$_SESSION = array()`	Открывает сессию.
`session_start()`	Закрывает текущую сессию.
`isset($_SESSION['user_id'])`	Удаляет все переменные сессии.

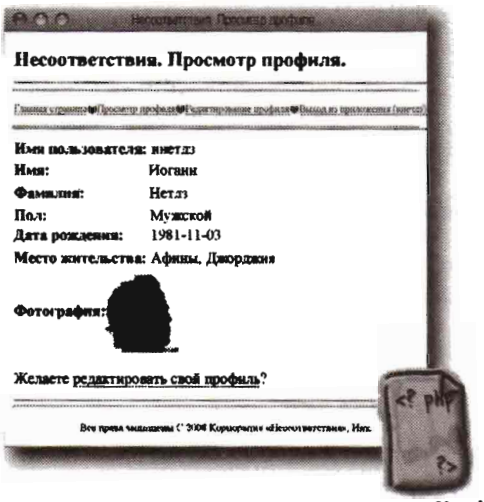
Совместное использование значит забота

Зонтик — не единственная вещь, которой можно пользоваться совместно. В любом веб-приложении вы можете столкнуться с ситуацией, когда *один и тот же фрагмент кода повторяется* более чем в одном месте. И это не только неэкономно: это лишняя **головная боль при технической поддержке приложения**, так как в тех неизбежных ситуациях, когда вам придется вносить изменения, вы должны будете проследить, чтобы эти изменения были внесены во все места, где у вас размещен повторяющийся код. Решение этой проблемы — **исключение дублирования кода путем его совместного использования**. Говоря другими словами, вы размещаете в каком-то одном месте код, необходимый в нескольких местах, а затем просто ссылаетесь на него, когда у вас возникает в этом необходимость.

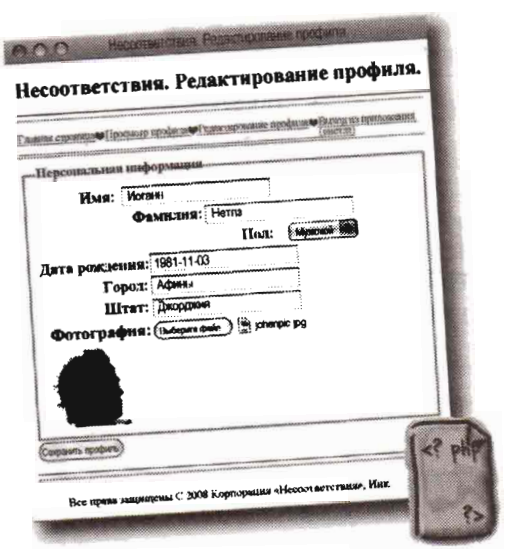
УПРАЖНЕНИЕ

Приложение «Несоответствия» изменилось с того момента, когда вы видели его в последний раз. Навигация стала удобнее, улучшился внешний вид приложения. Но все эти улучшения достигнуты путем… дублирования кода. Просто посмотрев на отдельные страницы приложения «Несоответствия», попробуйте определить, какие его части являются результатом интерпретации дублирующегося кода. Прокомментируйте их. Определите также фрагменты кода, которые не генерируют HTML-код и не выводят никакой информации в окне браузера, но также, по вашему мнению, связаны с проблемой дублирования кода.



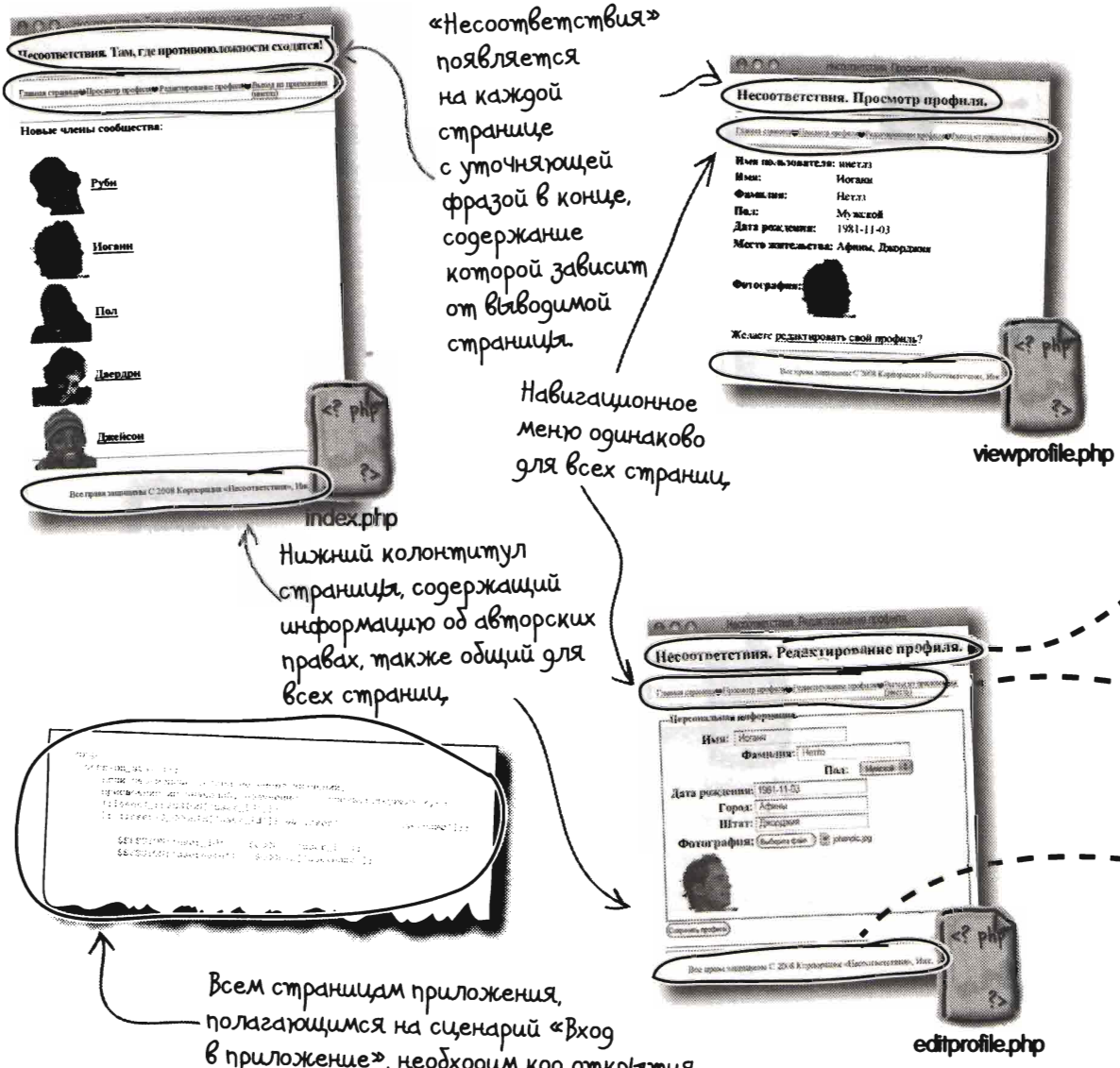


viewprofile.php



editprofile.php

Приложение «Несоответствия» изменилось с того момента, когда вы видели его в последний раз. Навигация стала удобнее, улучшился внешний вид приложения. Но все эти улучшения достигнуты ценой… дублирования кода. Просто посмотрев на отдельные страницы приложения «Несоответствия», попробуйте определить, какие его части являются результатом интерпретации дублирующегося кода. Прокомментируйте их. Определите также фрагменты кода, которые не генерируют HTML-код и не выводят никакой информации в окне браузера, но также, по вашему мнению, связаны с проблемой дублирования кода.



Составные части приложения «Несоответствия»

Итак, у приложения «Несоответствия» есть некоторые общие элементы, которые в настоящий момент дублируются в главных сценариях. Но почему этому уделяется такое большое внимание? Потому что это значительно затрудняет техническую поддержку приложения. Что произойдет, если у вас появится необходимость добавить новую страницу, которая потребует и добавления новой позиции меню? Вам нужно будет пройти через все сценарии и изменить в каждом файле код, отвечающий за вывод новой позиции меню. То же самое касается информации об авторских правах.

Решение этой проблемы заключается в том, чтобы каждый законченный фрагмент кода записывался в приложении только один раз. Тогда, если возникнет необходимость внесения изменений в код, вам нужно будет сделать это только в одном месте. Имея это в виду, есть смысл пересмотреть организацию приложения «Несоответствия» и включить в него совместно используемые компоненты.

Заголовок страницы

header.php

Сценарий `header.php` включает код, выводящий заголовок страницы, который ссылается на переменную, содержащую значение уточняющего добавления к заголовку — различного для различных страниц. Этот сценарий также включает стандартные HTML-теги заголовка веб-страницы и предусматривает использование вспомогательных средств, например каскадных таблиц стилей CSS.

Этот компонент не генерирует HTML-код для вывода какой-либо информации в окне браузера, но играет жизненно важную роль в управлении входом пользователей во всем приложении «Несоответствия».

Навигационное меню

navmenu.php

Сценарий `navmenu.php` содержит код, выводящий навигационное меню приложения, основываясь на том, вошел пользователь в приложение или нет. В зависимости от этих данных навигационное меню предоставляет параметры «Войти» в приложение или «Выйти» из приложения.

Открытие сессии

startsession.php

Сценарий `startsession.php` содержит код, ответственный за открытие сессии и проверку, вошел пользователь в приложение или нет.

Нижний колонтитул страницы

footer.php

Сценарий `footer.php` содержит код, выводящий информацию об авторских правах, а также закрывающий HTML-тег. Соответствующий ему открывающий тег расположен в сценарии `header.php`. То есть сценарии `header.php` и `footer.php`, работая в паре, должны всегда использоваться совместно.

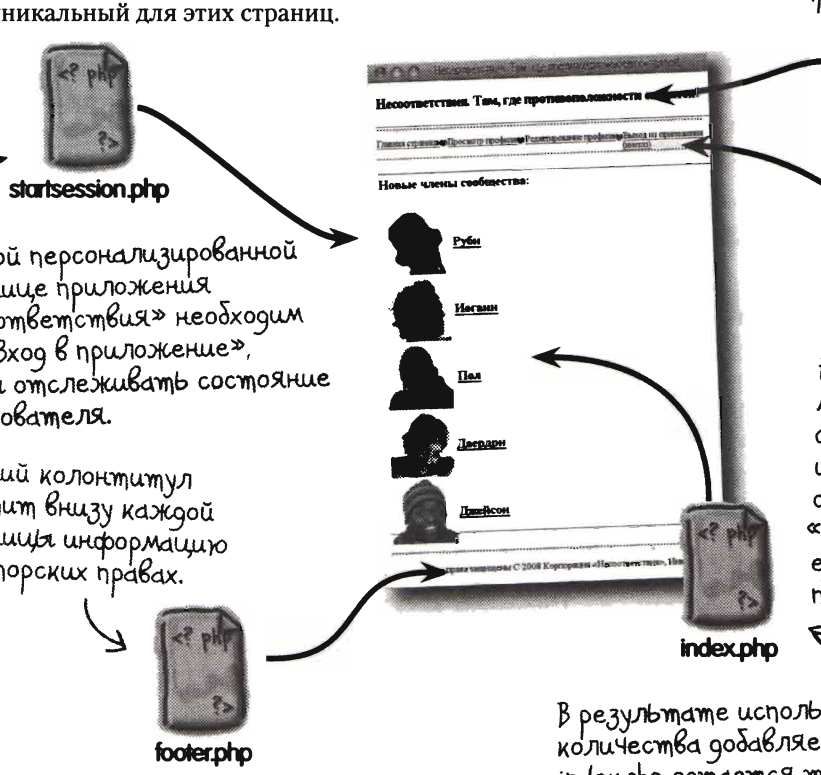
Преобразование приложения «Несоответствия» с использованием шаблонов

Итак, мы разбили приложение «Несоответствия» на множество составных частей, но как мы теперь соберем все это воедино? Вы уже знаете, как используются добавляемые файлы, и в этом заключается часть решения. Но вы должны идти дальше простого понятия «добавляемые файлы»... Вы должны мыслить в терминах шаблонов, которые позволяют создавать отдельную страницу, комбинируя несколько разных добавляемых файлов. Шаблон можно представить себе как образец страницы внутри приложения, в котором весь код хотя и является уникальным для этой страницы, но образуется комбинацией добавляемых файлов.

Версия приложения «Несоответствия», построенная на базе шаблонов, характеризуется тем, что совместно используемый код распределяется между отдельными сценариями, каждый из которых выполняет определенную функцию, иногда генерируя HTML-код, иногда — нет. Идея заключается в том, чтобы перенести как можно больше совместно используемого кода во включаемые файлы шаблонов, оставляя в сценариях, генерирующих конкретные страницы приложения, только код, уникальный для этих страниц.

Шаблоны позволяют создавать PHP-приложения из многократно используемых компонентов.

Заголовок появляется в начале почти каждой страницы приложения «Несоответствия», демонстрируя название приложения, дополненное названием конкретной страницы.



Каждой персонализированной странице приложения «Несоответствия» необходим код «Вход в приложение», чтобы отслеживать состояние пользователя.

Навигационное меню появляется сразу за заголовком и предоставляет всем страницам приложения «Несоответствия» единообразное меню для перехода между ними.

Нижний колонтитул выводит внизу каждой страницы информацию об авторских правах.

В результате использования такого большого количества добавляемых сценариев сценарию index.php остается только сосредоточиться на своей единственной роли, которая заключается в выводе перечня новых членов сообщества.

не бывает глупых вопросов

В: Что такое шаблоны? Разве это не просто группа добавляемых файлов?

О: Да, в сущности, шаблоны — это группа добавляемых файлов, но она разработана специально для разделения приложения на функциональные компоненты. Основная цель заключается в том, чтобы оставить в сценарии страницы только тот код, который является уникальным для этой конкретной страницы. Поэтому заголовок, нижний колонтитул, навигационное меню и некоторые другие части приложения, которые одинаковы или очень похожи для нескольких страниц, — это идеальные кандидаты на включение в состав шаблонов приложения. Вы достигаете этой цели, помещая код шаблонов в добавляемые файлы, на которые ссылаются другие сценарии по мере необходимости.

В: Должен ли совместно используемый код быть совершенно одинаковым в разных сценариях, чтобы его можно было использовать для создания шаблона?

О: Нет. Для использования кода в качестве шаблона совсем необязательно, чтобы он был совершенно одинаковым в разных сценариях. Вполне достаточно, чтобы он был просто похожим. Это становится возможным потому, что вы можете использовать переменные для достижения определенного уровня соответствия требованиям конкретного сценария. Шаблон заголовка един для всех страниц в том, что он начинается с названия приложения «Несоответствия», но далее следует название конкретной страницы, которое может быть добавлено с использованием соответствующей переменной.

Преобразуйте приложение «Несоответствия» с использованием шаблонов

Усилия, потраченные на разбивку приложения на сценарии шаблонов, обычно не пропадают даром. В результате этой работы вы получили группу небольших сценариев шаблонов с узконаправленной функциональностью. Но это привело также к значительному упрощению кода и сокращению размеров основных сценариев приложения, которые теперь ссылаются на шаблоны.

Если переменные сессии не установлены, попытка установить их с использованием куки.

Открытие сессии.

```
<?php
  session_start();
  // Если переменные сессии не установлены, попытка установить их
с использованием куки
  if (!isset($_SESSION['user_id'])) {
    if (isset($_COOKIE['user_id']) &&
        isset($_COOKIE['username'])) {
      $_SESSION['user_id'] = $_COOKIE['user_id'];
      $_SESSION['username'] = $_COOKIE['username'];
    }
  }
?>
```

startsession.php

Ссылка на каскадные таблицы стилей приложения.

HTML-теги DOCTYPE и <html>, образующие HTML-заголовок веб страницы.

Создание заголовка конкретной страницы с добавлением к названию приложения названия соответствующей страницы, которое содержится в переменной $page_title.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="ru">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<?php
  echo '<title>Несоответствия. ' . $page_title . '</title>';
?>

    <link rel="stylesheet" type="text/css" href="style.css" />
  </head>
  <body>

<?php
  echo '<h3>Несоответствия. ' . $page_title . '</h3>';
?>
```

header.php

```
<?php
  // Генерирование навигационного меню
  echo '<hr />';
  if (isset($_SESSION['username'])) {
    echo '<a href="index.php">Главная страница</a> &#10084; ';
    echo '<a href="viewprofile.php">Просмотр профиля</a> &#10084; ';
    echo '<a href="editprofile.php">Редактирование профиля </a> &#10084; ';
    echo '<a href="logout.php">Выход из приложения (' . $_
SESSION['username'] . ')</a>';
  }
  else {
    echo '<a href="login.php">Вход в приложение</a> &#10084; ';
    echo '<a href="signup.php">Создание учетной записи</a>';
  }
  echo '<hr />';
?>
```

navmenu.php

Проверка, вошел пользователь в приложение или нет, и на основании этих данных создание меню соответствующего состава.

Вывод информации об авторских правах и закрытие HTML-кода.

```
  <hr />
  <p class="footer">Все права защищены &copy;
   2008 Корпорация «Несоответствия», Инк.</p>
</body>
</html>
```

footer.php

Сценарий startsession.php должен быть включен в самом начале для того, чтобы открыть сессию и предоставить сценарию доступ к ее данным.

Переменная $page_title определяет заголовок страницы, который будет присоединен к общему заголовку приложения.

```
<?php
  // Открытие сессии
  require_once('startsession.php');

  // Вывод заголовка страницы
  $page_title = 'Там, где противоположности сходятся!';
  require_once('header.php');

  require_once('appvars.php');
  require_once('connectvars.php');

  // Вывод навигационного меню
  require_once('navmenu.php');

  // Соединение с базой данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  // Извлечение данных пользователей из базы данных MySQL
  $query = "SELECT user_id, first_name, picture FROM mismatch_user WHERE first_name IS NOT NULL ORDER
BY join_date DESC LIMIT 5";
  $data = mysqli_query($dbc, $query);

    // Прохождение в цикле массива данных пользователей,
  // форматирование данных в виде HTML
  echo '<h4>Новые члены сообщества:</h4>';
  echo '<table>';
  while ($row = mysqli_fetch_array($data)) {
    if (is_file(MM_UPLOADPATH . $row['picture']) && filesize(MM_UPLOADPATH . $row['picture']) > 0) {
      echo '<tr><td><img src="' . MM_UPLOADPATH . $row['picture'] . '" alt="' . $row['first_name'] . '"
/></td>';
    }
    else {
      echo '<tr><td><img src="' . MM_UPLOADPATH . 'nopic.jpg' . '" alt="' . $row['first_name'] . '"
/></td>';
    }
    if (isset($_SESSION['user_id'])) {
      echo '<td><a href="viewprofile.php?user_id=' . $row['user_id'] . '">' . $row['first_name'] .
'</a></td></tr>';
    }
    else {
      echo '<td>' . $row['first_name'] . '</td></tr>';
    }
  }
  echo '</table>';

  mysqli_close($dbc);
?>

<?php
  // Вывод нижнего колонтитула
  require_once('footer.php');
?>
```

Переменные, содержащие параметры соединения с базой данных, и общие переменные приложения включаются из добавляемых файлов, как и прежде.

Навигационное меню выводится после заголовка, но перед основным содержанием страницы.

Код, не использующий шаблония, теперь действительно уникален для этой страницы, и его стало значительно меньше.

Сценарий нижнего колонтитула завершает страницу и должен быть самым последним, так как содержит HTML-тег, закрывающий страницу.

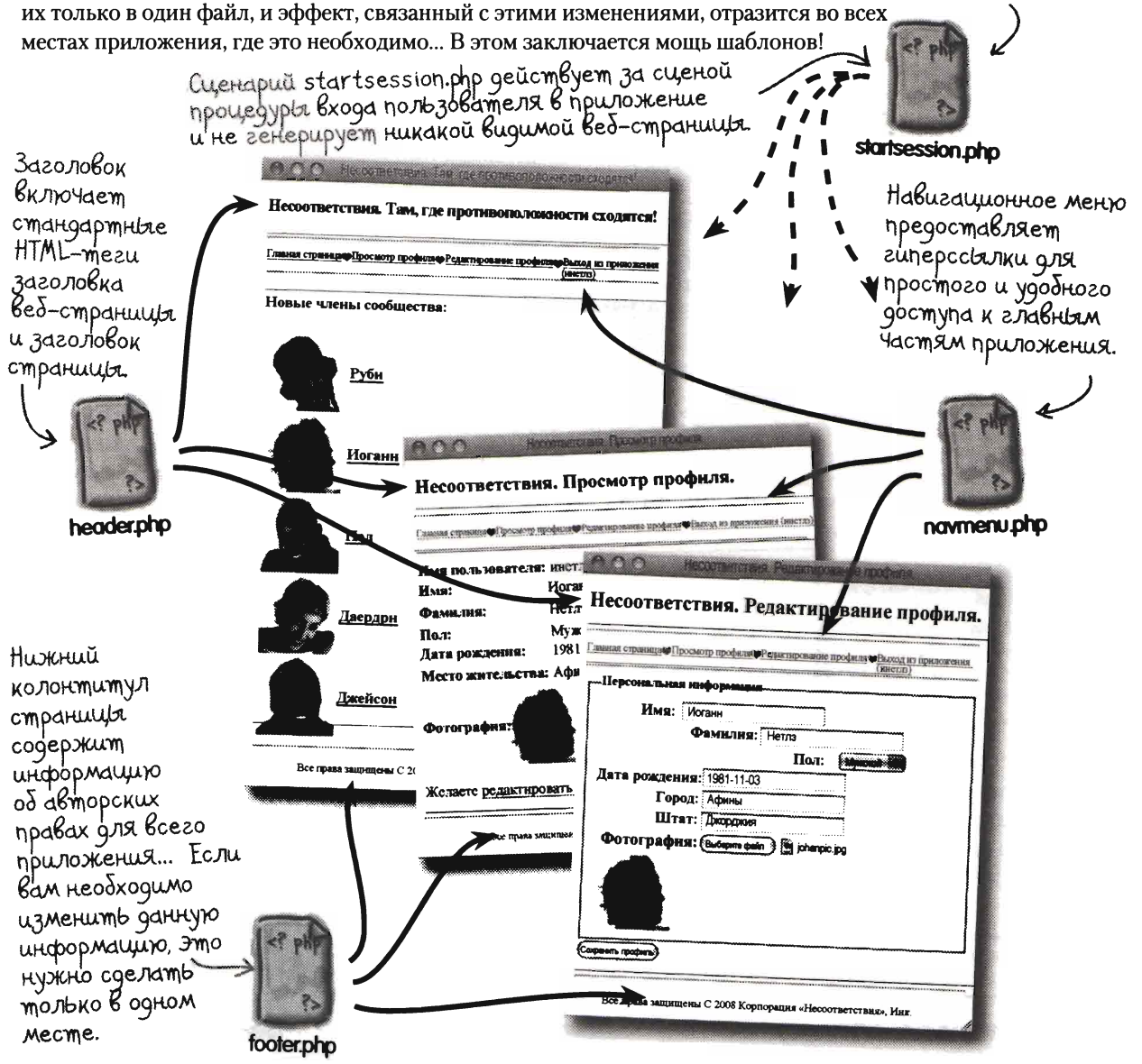
index.php

Приложение «Несоответствия» снова завершено... гораздо и лучше организовано

В то время как процесс деления приложения «Несоответствия» на мелкие кусочки мог вызвать некоторое беспокойство, то, что получилось в конечном итоге, действительно оправдывает затраченные усилия. Приложение теперь распределено между несколькими новыми (добавляемыми) файлами-шаблонами, которые обеспечивают лучшую организацию и повышают степень совместного использования кода сценариев. Если у вас возникнет необходимость внести какие-то изменения, достаточно внести их только в один файл, и эффект, связанный с этими изменениями, отразится во всех местах приложения, где это необходимо... В этом заключается мощь шаблонов!

Код открытия сессии используется любым сценарием, которому необходимы входные данные пользователя.

Сценарий startsession.php действует за сценой процедуры входа пользователя в приложение и не генерирует никакой видимой веб-страницы.



8 управляйте своими данными, управляйте окружающим вас миром

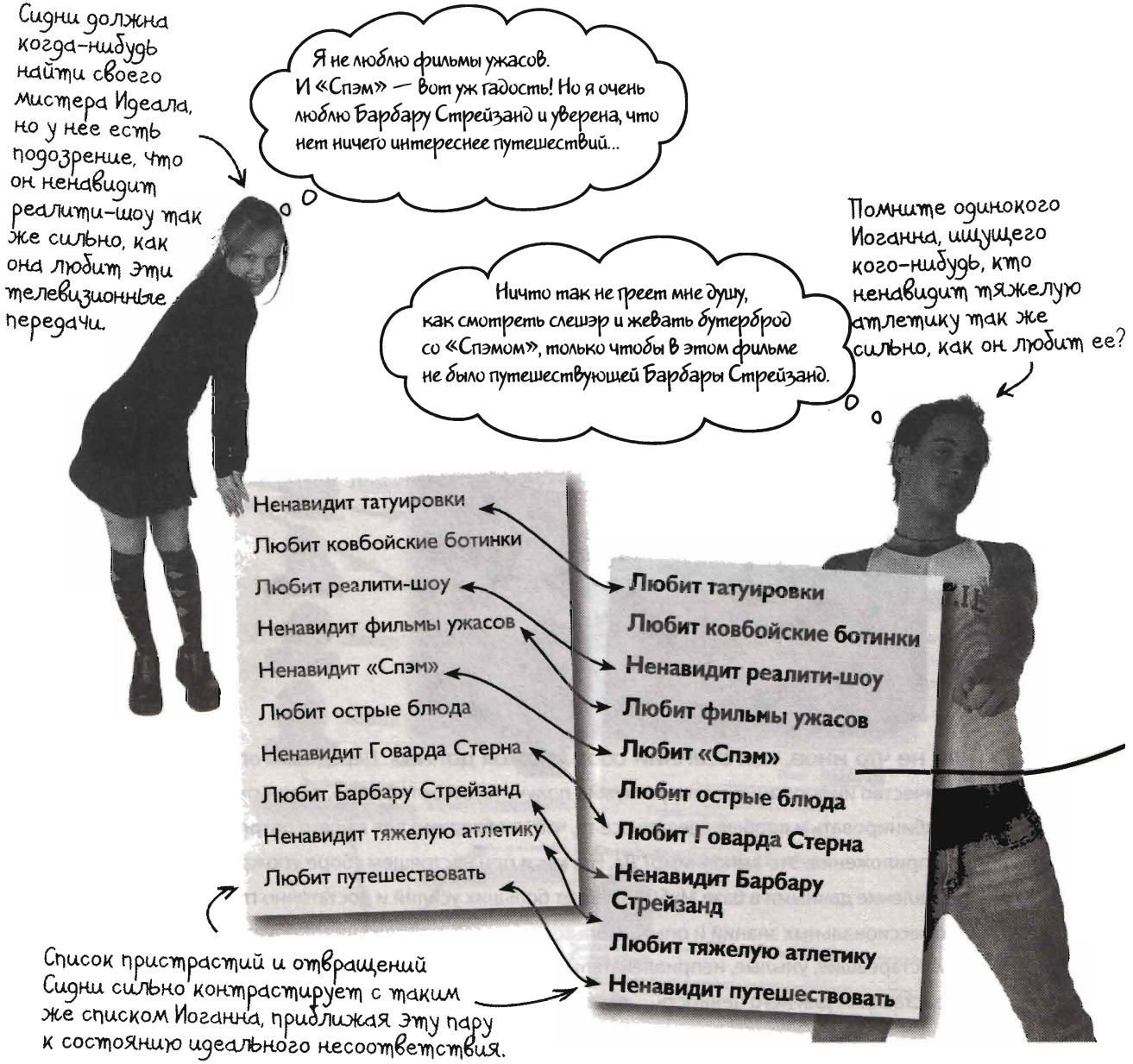
Сбор урожая данных

Это не что иное, как осенний сбор урожая данных. Перед вами огромное количество информации, которую нужно получить, отсортировать, сравнить, скомбинировать и вообще сделать все то, что необходимо для вашего потрясающего веб-приложения. Это выполнимо? Да. Но, как и при настоящем сборе урожая, управление данными в базе MySQL требует больших усилий и достаточно глубоких профессиональных знаний и опыта. Пользователь веб-приложения требует большего, чем устаревшие, унылые, непривлекательные данные. Он ожидает данных, которые представляют для него ценность… данных, которые призывают к действию… данных, которые обогащают. Поэтому чего вы ждете? Заводите свой MySQL-комбайн и принимайтесь за работу!

это новая глава

Создание идеального несоответствия

У приложения «Несоответствия» имеется разросшаяся база, содержащая данные о множестве членов сообщества, которые ожидают увидеть какие-то реальные результаты. Нам необходимо дать пользователям возможность найти себе идеальную пару путем сравнения их пристрастий и отвращений с такими же чувствами других пользователей для поиска идеального несоответствия. Каждая пара, где пристрастию с одной стороны отвечает отвращение с другой, приближается к тому состоянию, которое мы называем идеальным несоответствием.



Несоответствия — это все данные

Прежде чем устанавливать несоответствия между пользователями, мы должны решить, как организовать данные, чтобы они содержали в себе признаки несоответствия (пристрастие или отвращение). Осознания того, что эта информация будет сохранена в базе данных, недостаточно. Мы должны организовать эти признаки так, чтобы они стали более управляемыми, и дать пользователям возможность высказывать свое пристрастие или отвращение к каждому из этих признаков несоответствия, то есть присваивать этим признакам несоответствия значения пристрастия или отвращения.

Напишите, как бы вы организовали отдельные группы данных приложения «Несоответствия» для сохранения их в базе данных:

Классификация данных приложения «Несоответствия»

Разработка модели данных для приложений, подобных «Несоответствиям», является очень важным этапом, так как она в значительной степени определяет, как будет строиться приложение в целом. В случае приложения «Несоответствия» нам необходимо разбить параметры данных на три взаимозависимых группы.

Категории используются для группирования подобных признаков несоответствия.

Признаки несоответствия являются одними из самых существенных данных приложения «Несоответствия», так как именно они определяют, в чем пользователи не соответствуют друг другу.

Категория

Категории используются для организации признаков несоответствия. Хотя они и не определяют напрямую признаки несоответствия пользователей, но значительно облегчат процедуру ввода значений этих признаков в приложение.

Времяпрепровождение

Организация досуга

Признак несоответствия

Несоответствие возникает по различным признакам, например, таким как отношение к татуировке или острым блюдам. К каждому из них пользователь может испытывать либо пристрастие, либо отвращение.

Реалити-шоу

Тяжелая атлетика

Фильмы ужасов

Путешествия

Значение признака несоответствия

Пользователь описывает себя в контексте признаков несоответствий, давая в ответ на запрос приложения каждому из этих признаков одно из двух значений (пристрастие или отвращение).

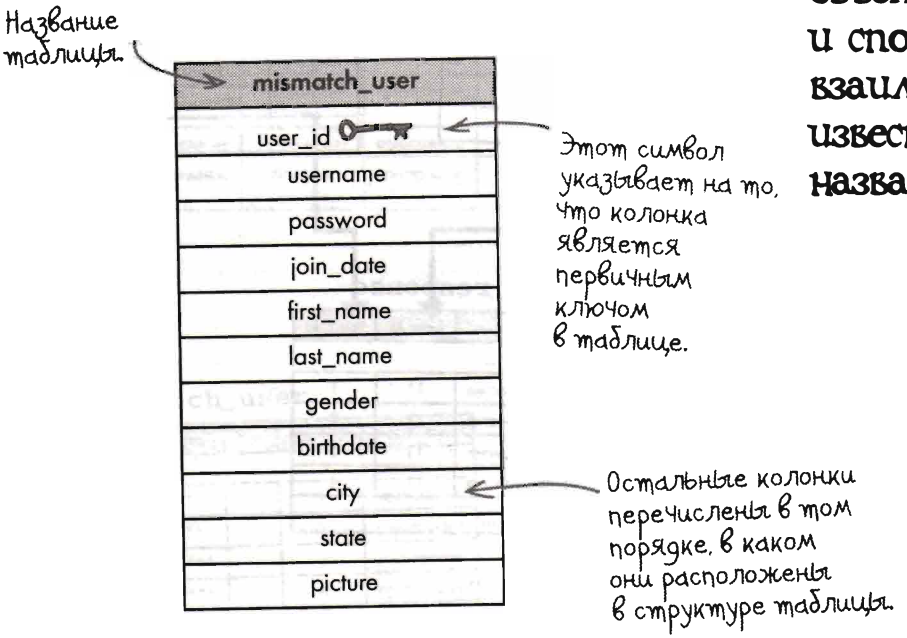
Для каждого из признаков несоответствия ответы могут принимать одно из двух значений (пристрастие или отвращение). Они являются индивидуальными для каждого пользователя приложения «Несоответствия».

Как происходит процесс определения несоответствия между двумя пользователями? Мы сравниваем значения, присвоенные пользователями каждому из признаков несоответствия. Например, раз Сидни и Иоганн дают противоположные ответы на запрос об их отношении к фильмам ужасов, для признака несоответствия «Фильмы ужасов» имеет место успешное несоответствие. Нахождение наилучшего несоответствия для данного пользователя является результатом нахождения другого пользователя, у которого имеется максимальное число несоответствий с ним.

Моделирование базы данных с помощью схемы

Для того чтобы воплотить классификацию данных приложения «Несоответствия», описанную на предыдущей странице, в конкретную структуру базы данных, нам необходима ее схема. Схема — это представление всех объектов базы данных, таких как таблицы с их колонками, вместе с их взаимными связями. Создание визуального представления вашей базы данных может помочь лучше понять, как подключаются различные ее компоненты при выполнении запросов, не говоря уже о том, какие колонки ответственны за эти подключения. В качестве примера давайте рассмотрим схему базы данных первоначальной версии приложения «Несоответствия», показанную в предыдущей главе. В ее состав входит только одна таблица — `mismatch_user`.

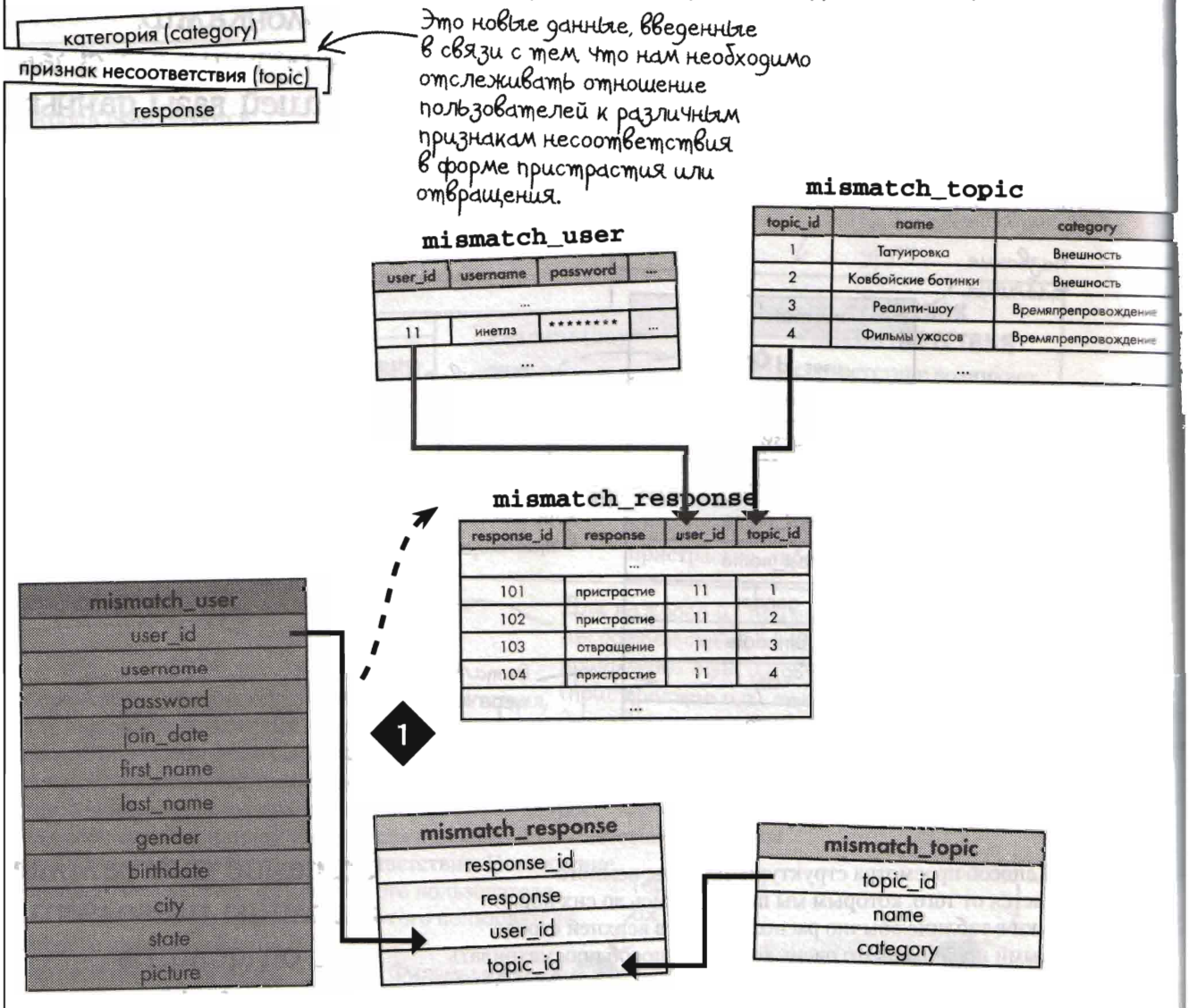
Описание объектов (таблиц с их колонками), входящих в состав вашей базы данных, вместе с другими связанными с ними объектами и способы их взаимодействия известны под названием схемы.

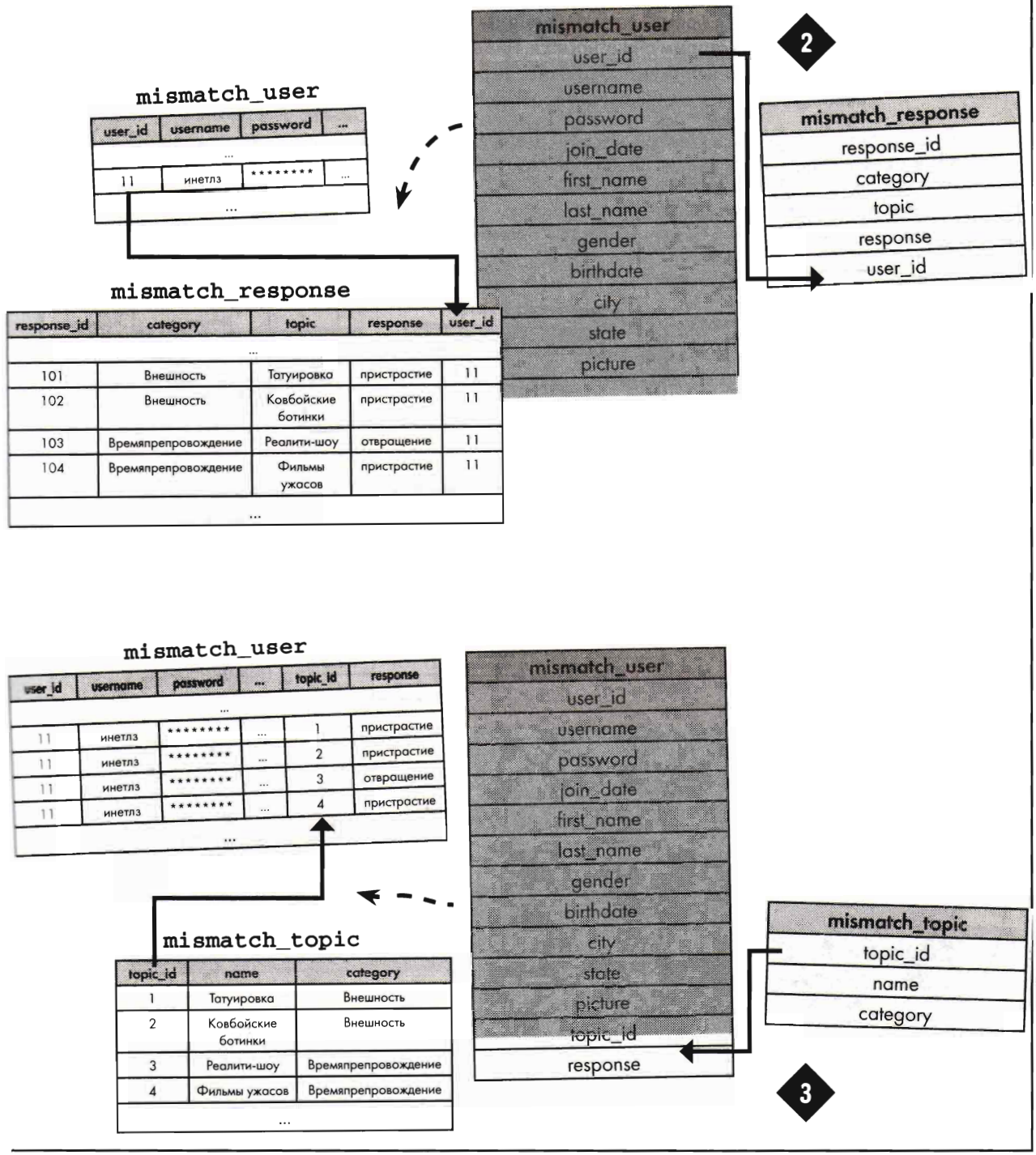


Такой способ просмотра структуры таблицы немного отличается от того, которым мы пользовались до сих пор. Колонки в таблице обычно располагались в верхней строке с данными под ними. Это очень хороший способ просматривать индивидуальные таблицы, а также таблицы вместе с их данными. Но это не слишком удобно в тех случаях, когда мы хотим показать структурную диаграмму базы данных, в состав которой входит множество таблиц, со всеми их связями между собой и с другими объектами. А базе данных приложения «Несоответствия» уже необходимы дополнительные таблицы...

Создание диаграммы таблицы позволяет вам отделить структуру таблицы от сохраненных в ней данных.

Необходимо сохранить в базе данных приложения «Несоответствия» значения признаков несоответствия в форме «пристрастие/отвращение», а также наименования этих признаков и категории, к которым они относятся. Ниже приведены три различные структуры баз данных приложения «Несоответствия». Выберите схему, которая, по вашему мнению, наиболее отвечает поставленным требованиям, и прокомментируйте свой выбор.




mismatch_user
user_id
username
password
join_date
first_name
last_name
gender
birthdate
city
state
picture
2
mismatch_response
response_id
category
topic
response
user_id
mismatch_user
user_id username password ...
...
11 инетлз ******** ...
...
mismatch_response
response_id category topic response user_id
...
101 Внешность Татуировка пристрастие 11
102 Внешность Ковбойские ботинки пристрастие 11
103 Времяпрепровождение Реалити-шоу отвращение 11
104 Времяпрепровождение Фильмы ужасов пристрастие 11
...
mismatch_user
user_id username password ... topic_id response
...
11 инетлз ******** ... 1 пристрастие
11 инетлз ******** ... 2 пристрастие
11 инетлз ******** ... 3 отвращение
11 инетлз ******** ... 4 пристрастие
...
mismatch_topic
topic_id name category
1 Татуировка Внешность
2 Ковбойские ботинки Внешность
3 Реалити-шоу Времяпрепровождение
4 Фильмы ужасов Времяпрепровождение
...
mismatch_user
user_id
username
password
join_date
first_name
last_name
gender
birthdate
city
state
picture
topic_id
response
mismatch_topic
topic_id
name
category
3

Необходимо сохранить в базе данных приложения «Несоответствия» значения признаков несоответствия в форме «пристрастие/отвращение», а также наименования этих признаков и категории, к которым они относятся. Ниже приведены три различные структуры баз данных приложения «Несоответствия». Выберите схему, которая, по вашему мнению, наиболее отвечает поставленным требованиям, и прокомментируйте свой выбор.

Прежде всего необходимо отметить, что новыми являются только данные, требующие от пользователя ответа на вопрос, питает он пристрастие или отвращение к определенным признакам несоответствия. Все остальные данные в базе остаются постоянными, по крайней мере, с точки зрения пользователя.

Кто сказал, что чем проще — тем лучше? В этой схеме значения признаков несоответствия сохраняются в своей собственной таблице, отдельно от остальных данных, на которые они непосредственно не влияют. Здесь нет никакого дублирования, связанного с признаками несоответствия, так как пользователи, связанные с ними признаки несоответствия и категории сохраняются вне таблицы mismatch_response.

mismatch_user

user_id	username	password	...
...			
11	инетлз	********	...
...			

mismatch_topic

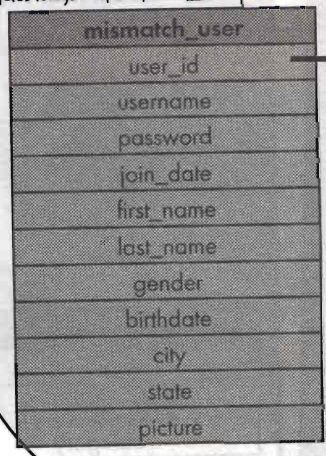
topic_id	name	category
1	Татуировка	Appearance
2	Ковбойские ботинки	Appearance
3	Реалити-шоу	Entertainment
4	Фильмы ужасов	Entertainment
...		

В новой таблице mismatch_topic сохраняются наименования признаков несоответствия и соответствующие им категории.

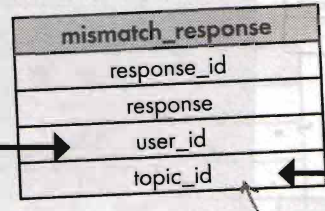
mismatch_response

response_id	response	user_id	topic_id
...			
101	пристрастие	11	1
102	пристрастие	11	2
103	отвращение	11	3
104	пристрастие	11	4
...			

Очень хорошо, что нет дублирования данных для каждого ответа.



1



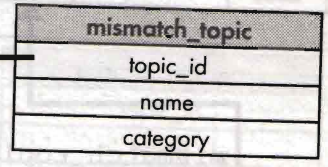


Таблица mismatch_user остается в прежнем виде.

Таблица mismatch_response устанавливает связь между пользователем и признаком несоответствия через колонки user_id и topic_id.

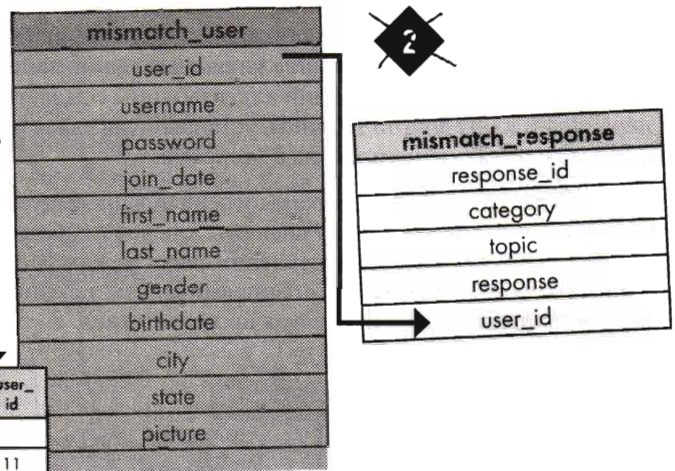
Значения признаков несоответствия сохраняются вне таблицы mismatch_user, и это очень хорошо. Но в таблице mismatch_response огромное количество дублирующихся данных, так как признак несоответствия и категория, которой этот признак принадлежит, сохраняются в каждой записи для каждого ответа.

mismatch_user

user_id	username	password	...
...			
11	инетлз	********	...
...			

mismatch_response

response_id	category	topic	response	user_id
...				
101	Внешность	Татуировка	пристрастие	11
102	Внешность	Ковбойские ботинки	пристрастие	11
103	Времяпрепровождение	Реалити-шоу	отвращение	11
104	Времяпрепровождение	Фильмы	пристрастие	11
...				



Признак несоответствия и категория, к которой этот признак принадлежит, сохраняются в каждой записи для каждого ответа, что крайне расточительно.

Данные пользователя дублируются для каждого ответа.

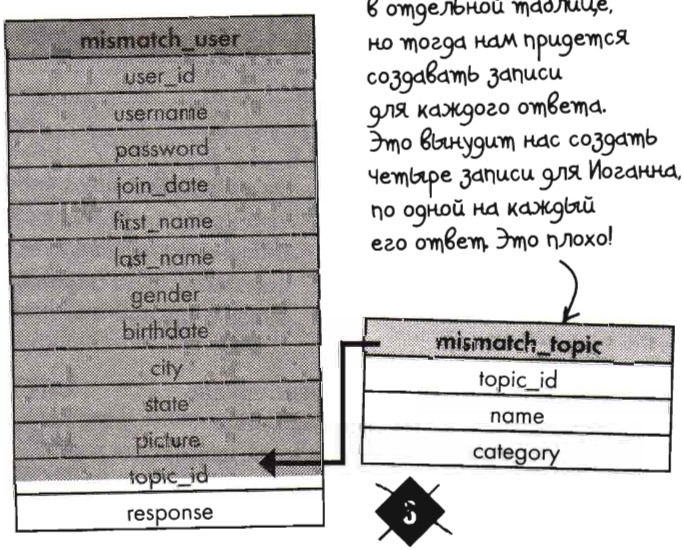
mismatch_user

user_id	username	password	...	topic_id	response
...					
11	инетлз	********	...	1	пристрастие
11	инетлз	********	...	2	пристрастие
11	инетлз	********	...	3	отвращение
11	инетлз	********	...	4	пристрастие
...					

mismatch_topic

topic_id	name	category
1	Татуировка	Внешность
2	Ковбойские ботинки	Внешность
3	Реалити-шоу	Времяпрепровождение
4	Фильмы ужасов	Времяпрепровождение
...		

Неплохая идея — сохранять признаки несоответствия и категории, к которым эти признаки относятся, в отдельной таблице, но тогда нам придется создавать записи для каждого ответа. Это вынудит нас создать четыре записи для Иоганна, по одной на каждый его ответ. Это плохо!

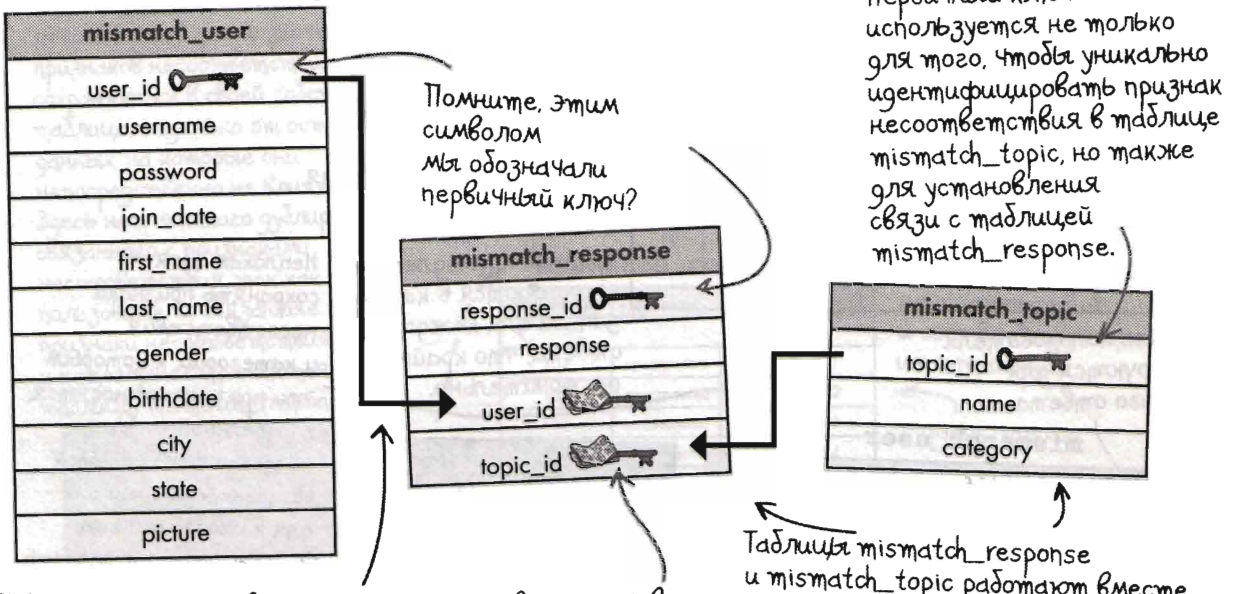


Свяжите друг с другом несколько таблиц

Связывание таблиц друг с другом с целью создания единой системы данных происходит с помощью ключей. Мы уже использовали первичные ключи для создания уникальных идентификаторов, но сейчас нам необходимы внешние ключи для организации связи между двумя таблицами. Внешний ключ таблицы ссылается на первичный ключ другой таблицы, устанавливая связь между ними, которая может быть использована в запросах.

Внешний ключ — это колонка таблицы, которая ссылается на первичный ключ другой таблицы.

Схема базы данных `mismatch`, рассмотренная в предыдущем упражнении, полагается на пару внешних ключей в таблице `mismatch_response` для установления связи между записями этой таблицы, содержащими ответы, с записями других таблиц, содержащих данные о пользователях и признаках несоответствия.



Без внешних ключей было бы очень трудно ассоциировать данные из одной таблицы с данными из другой. А распределение данных между несколькими таблицами позволяет нам избежать дублирования информации и повысить эффективность базы данных. Таким образом, внешние ключи играют огромную роль во всех базах данных, за исключением совсем уж простых.

Стрелки, изображенные толстыми линиями, показывают связь между таблицами, устанавливаемую с помощью первичных и внешних ключей.

Внешние ключи в действии

Визуализация процесса движения данных в базе и связи таблиц друг с другом с помощью первичных и внешних ключей часто помогают лучше разобраться в задаче. Более детальное рассмотрение таблиц, содержащих конкретные данные, поможет лучше понять, как первичные и внешние ключи связаны друг с другом.

Первичный ключ, колонка user_id, должен быть уникальным в таблице mismatch_user. Фактически, в этом заключается одна из его функций: предоставлять уникальный идентификатор записи пользователя.

Внешний ключ, колонка user_id, служит в качестве ссылки на запись пользователя в таблице mismatch_user, давая вам знать, какой пользователь связан с данным ответом.

Первичный ключ, колонка topic_id, служит в качестве уникального идентификатора записи признака в таблице mismatch_topic.

mismatch_user

user_id	username	password	...
...			
11	инетлз	********	...
...			

mismatch_topic

topic_id	name	category
1	Татуировка	Внешность
2	Ковбойские ботинки	Внешность
3	Реалити-шоу	Времяпрепровождение
4	Фильмы ужасов	Времяпрепровождение
...		

Не забывайте: каждая запись в этой таблице содержит данные одного пользователя.

Каждая запись в этой таблице содержит наименование признака несоответствия и его категорию, но не содержит значение признака (пристрастие/отвращение).

mismatch_response

response_id	response	user_id	topic_id
...			
101	пристрастие	11	
102	пристрастие	11	2
103	отвращение	11	3
104	пристрастие	11	4
...			

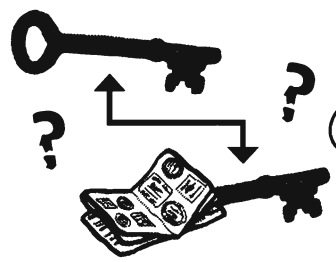
Каждая запись в этой таблице содержит ответ конкретного пользователя (любое из значений пристрастие/отвращение).

Внешний ключ, колонка topic_id, служит в качестве ссылки на запись в таблице mismatch_topic и не является уникальным, так как множество разных пользователей могут иметь одинаковые признаки несоответствий.

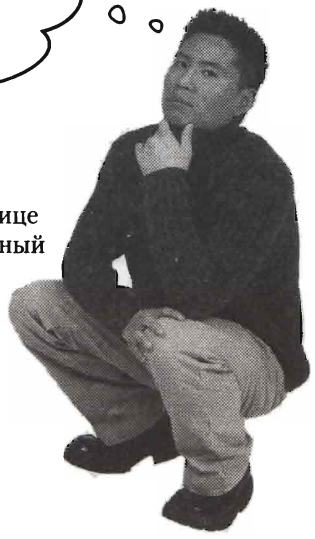
Просматривая признак несоответствия, введенный пользователем в таблицу `mismatch_response`, вы можете получить об этом пользователе больше информации, изучая его запись в таблице `mismatch_user`, на которую указывает внешний ключ `user_id` таблицы `mismatch_response`.

Связывание таблиц друг с другом с помощью первичного и внешнего ключей позволяет нам соединять данные, сохраненные в этих таблицах. Вы можете даже создать такую структуру базы данных, при которой соответствие между первичными и внешними ключами **должно обязательно** соблюдаться. Это называется поддержанием **целостности** базы данных **на уровне ссылочных данных**, что в переводе на «человеческий» язык означает: соответствия между первичными и внешними ключами должны быть действительными.

Внешний ключ, колонка user_id в таблице mismatch_response, связывает запись, содержащую информацию о признаках несоответствия пользователя, с записью этого пользователя в таблице mismatch_user.



Я понимаю, что первичный и внешний ключи связывают таблицы друг с другом, но имеет ли какое-то значение в этих диаграммах направление стрелок, соединяющих ключи?

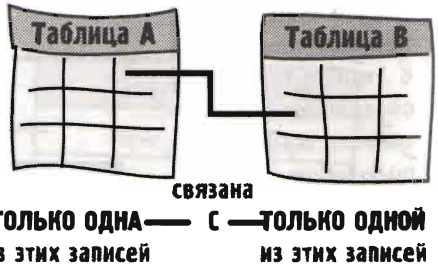


Да, направление стрелки говорит о том, как записи в таблицах зависят друг от друга.

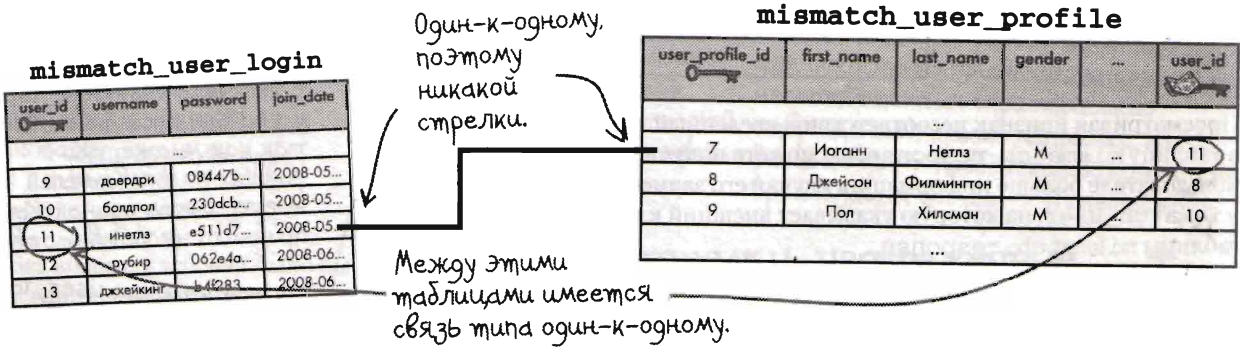
Если подробнее они говорят нам о том, сколько записей в одной таблице должно быть связано с записями в другой, и наоборот. Это очень важный аспект схемы базы данных, включающий три различных типа связей: один-к-одному, один-ко-многим и многие-ко-многим.

Таблицы могут связываться между собой по принципу «одна запись одной таблицы соответствует одной записи другой таблицы»

Первый тип связи, один-к-одному, устанавливает, что любая запись в таблице A может иметь не больше одной записи, которая соответствует записи в таблице B, и наоборот. Следовательно, существует только одно соответствие в каждой таблице для каждой записи.

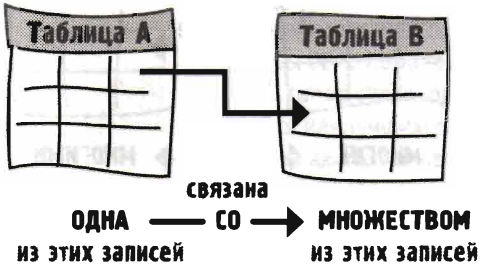


В качестве примера предположим, что в базе данных таблица, содержащая информацию о пользователях, разделена на две: первая (таблица A) содержит входные данные пользователей, а вторая (таблица B) — данные их профилей. Обе таблицы содержат идентификаторы пользователей, связывающие их с данными их профилей. Колонка `user_id` в таблице с входными данными является первичным ключом, что обеспечивает уникальность входных данных пользователей. Колонка `user_id` в таблице с данными профилей является внешним ключом, и ее роль заключается в обеспечении связи данных профилей с входными данными.



Что касается колонок `user_id` в этих двух таблицах, таблица с входными данными считается родительской, в то время как таблица с данными профилей — дочерней. Таблица с первичным ключом имеет связь типа родительская — дочерняя с таблицей с соответствующим внешним ключом.

Одна запись ведет ко многим



Связь один-ко-многим означает, что запись в таблице А может быть связана со множеством записей в таблице В, но запись в таблице В может быть связана только с одной записью в таблице А. Направление стрелки на диаграмме всегда показывает от таблицы с одной записью к таблице со множеством записей.

Рассматривая базу данных приложения «Несоответствия» снова, мы можем увидеть связь типа один-ко-многим. Так как пользователь может иметь множество значений признаков несоответствия (питает пристрастие к татуировке, отвращение к путешествиям и т. д.), здесь имеется связь типа один-ко-многим между таблицей, содержащей записи с входной информацией, и таблицей с записями, содержащими данные о признаках несоответствия.

Первичный ключ!

mismatch_user

user_id	username	password	...
9	даердри	08447b...	...
10	болдпол	230dcd...	...
11	инетлз	e511d7...	...
12	рубир	062e4a...	...
13	джхейкинг	b4f283...	...

Между этими таблицами имеется связь типа один-ко-многим через колонку user_id.

Внешний ключ!

mismatch_response

response_id	response	user_id	topic_id
101	пристрастие	11	1
102	пристрастие	11	2
103	отвращение	11	3
104	пристрастие	11	4

не бывает глупых вопросов

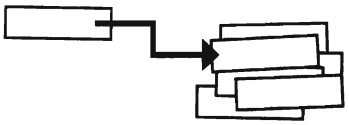
В: Как мне узнать, в каких случаях какой тип связи, один-к-одному или один-ко-многим, необходимо использовать?

О: Существует тенденция использовать связь типа один-ко-многим значительно чаще, чем связь типа один-к-одному и, в основном, по следующим причинам. Как правило, всегда существует главная (родительская) таблица, содержащая первичный ключ, такая, например, как `mismatch_user` в базе данных приложения «Несоответствия». Эта таблица связана со второй (дочерней) таблицей по типу один-ко-многим.

В выбранной нами схеме базы данных приложения «Несоответствия» такое наблюдается дважды, когда таблицы, содержащие данные о пользователях и наименования признаков несоответствия, имеют связи типа один-ко-многим с таблицей, содержащей значения признаков несоответствия.

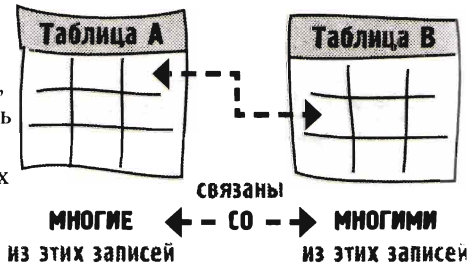
Во многих случаях таблицы, связанные по типу один-к-одному, могут быть объединены в одну. Тем не менее, конечно, могут возникать ситуации, когда использование двух таблиц, связанных по типу один-к-одному, будет оправданным, как в гипотетическом примере с таблицей, содержащей данные профилей пользователей и упомянутой на предыдущей странице. Здесь, исходя из соображений безопасности, некоторые данные перемещены в свою собственную таблицу.

Один-к-одному: только одна запись родительской таблицы связана с одной записью дочерней таблицы.

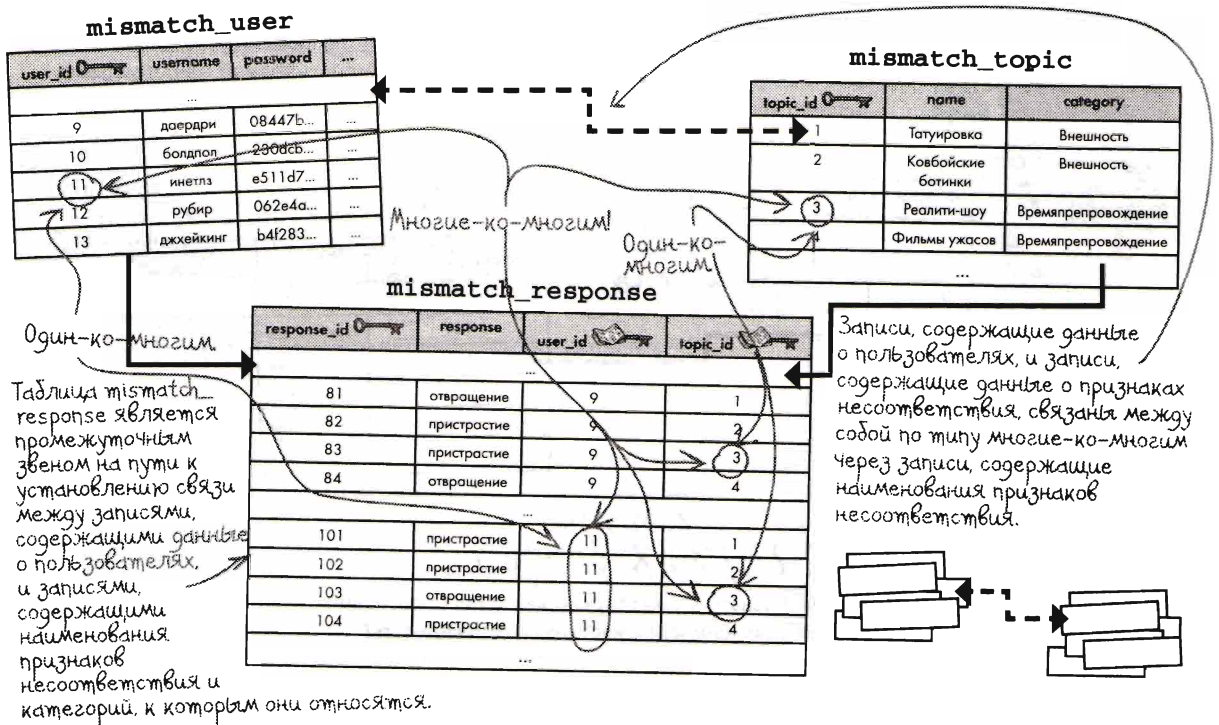


Один-ко-многим: только одна запись родительской таблицы связана со множеством записей дочерней таблицы.

Связь типа многие-ко-многим



Третьим и последним типом связи является тип многие-ко-многим, при котором множество записей в таблице А может соответствовать множеству записей в таблице В... Очень похоже на переполнение данными? Необязательно. Имеется множество ситуаций, в которых наличие связи типа многие-ко-многим практически обязательно. Может, приложение «Несоответствия» именно такое? Давайте рассмотрим подробнее.



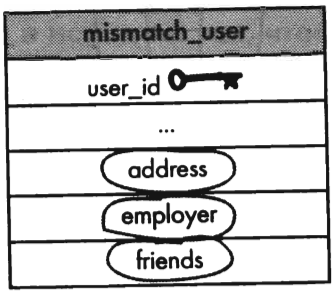
Многие-ко-многим: множество записей родительской таблицы связано со множеством записей дочерней таблицы

Связь типа многие-ко-многим между записями, содержащими данные о пользователях, и записями, содержащими данные о признаках несоответствия, является опосредованной, то есть она устанавливается через таблицу `mismatch_response`. Но как бы то ни было, в конечном счете это приводит к возникновению связи типа многие-ко-многим. Просто взгляните, сколько записей с одинаковыми значениями колонок `user_id` и `topic_id` содержится в таблице `mismatch_response`.

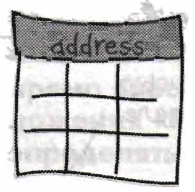
В дополнение к тому, что таблица `mismatch_response` сохраняет значения признаков несоответствия, она выступает также в роли удобного промежуточного звена на пути к установлению связи между записями, содержащими данные о пользователях, и записями, содержащими наименования признаков несоответствия и категорий, к которым они относятся. Отсутствие такой промежуточной таблицы привело бы к многократному дублированию данных, что является очень неприятным явлением. Если вы еще не убедились в этом, вернитесь к упражнению, описывающему схемы баз данных и приведенному в начале этой главы, и рассмотрите более внимательно вариант 2. В этой схеме колонки таблицы `mismatch_topic` включены с состав таблицы `mismatch_response`, результатом чего явилось многократное дублирование данных.

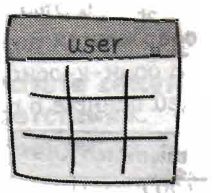
НАЗОВИТЕ ЭТУ СВЯЗЬ

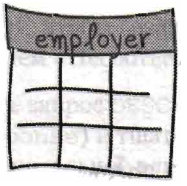
В каждой из таблиц, показанных ниже, имеются выделенные колонки, которые могли бы быть перенесены в свои собственные таблицы. Напишите, какой из трех типов связи, один-к-одному, один-ко-многим или многие-ко-многим, является наилучшим для связи новых таблиц с прежней в каждом случае. Затем покажите эти связи графически, начертив линии, соединяющие обе таблицы, с соответствующими направлениями стрелок.

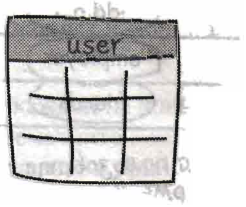


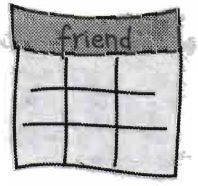
СВЯЗЬ

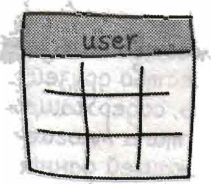


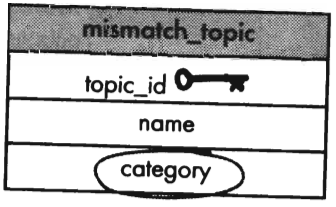


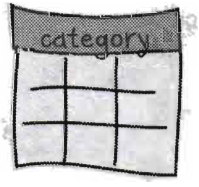


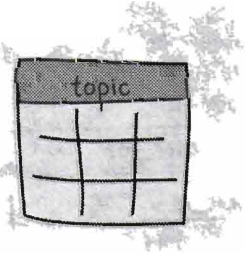










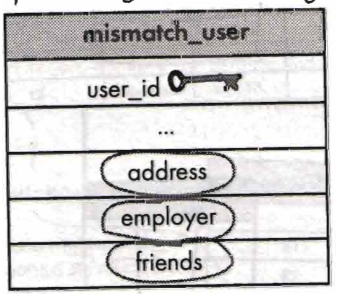


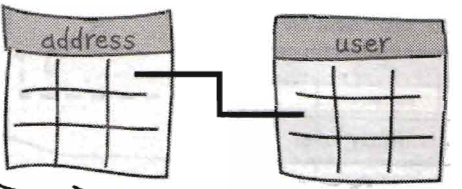
НАЗОВИТЕ ЭТУ СВЯЗЬ. РЕШЕНИЕ

В каждой из таблиц, показанных ниже, имеются выделенные колонки, которые могли бы быть перенесены в свои собственные таблицы. Напишите, какой из трех типов связи, один-к-одному, один-ко-многим или многие-ко-многим, является наилучшим для связи новых таблиц с прежней в каждом случае. Затем покажите эти связи графически, начертив линии, соединяющие обе таблицы, с соответствующими направлениями стрелок.

СВЯЗЬ

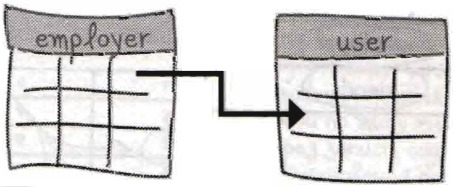
У каждого пользователя только один домашний адрес, и это означает, что запись, содержащая адрес, имеет связь типа один-к-одному с записью, содержащей данные о пользователе.





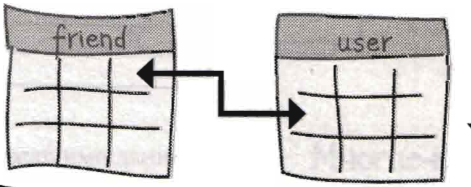
.....один-к-одному.....

Множество пользователей могут работать в одной фирме, в результате чего запись, содержащая данные о работодателе, имеет связь типа один-ко-многим с записью, содержащей данные о пользователе.



.....один-ко-многим....

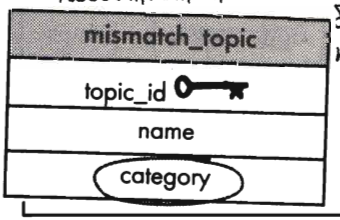
Множество пользователей могут иметь множество друзей, в результате чего запись, содержащая данные о друге, имеет связь типа многие-ко-многим с записью, содержащей данные о пользователе.

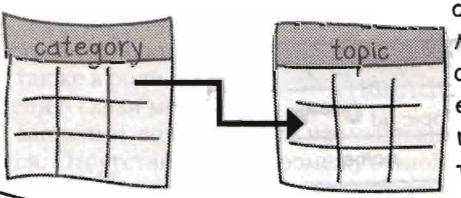


.....многие-ко-многим

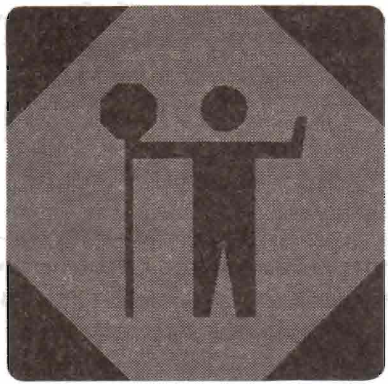
Таблицы, между которыми установлена связь типа многие-ко-многим обычно реализуют ее через промежуточную таблицу, которая здесь не показана

Множество признаков несоответствия могут принадлежать к одной и той же категории, в результате чего запись, содержащая данные о категории, имеет связь типа один-ко-многим с записью, содержащей данные о признаке несоответствия. Но признак несоответствия не может принадлежать больше чем одной категории.





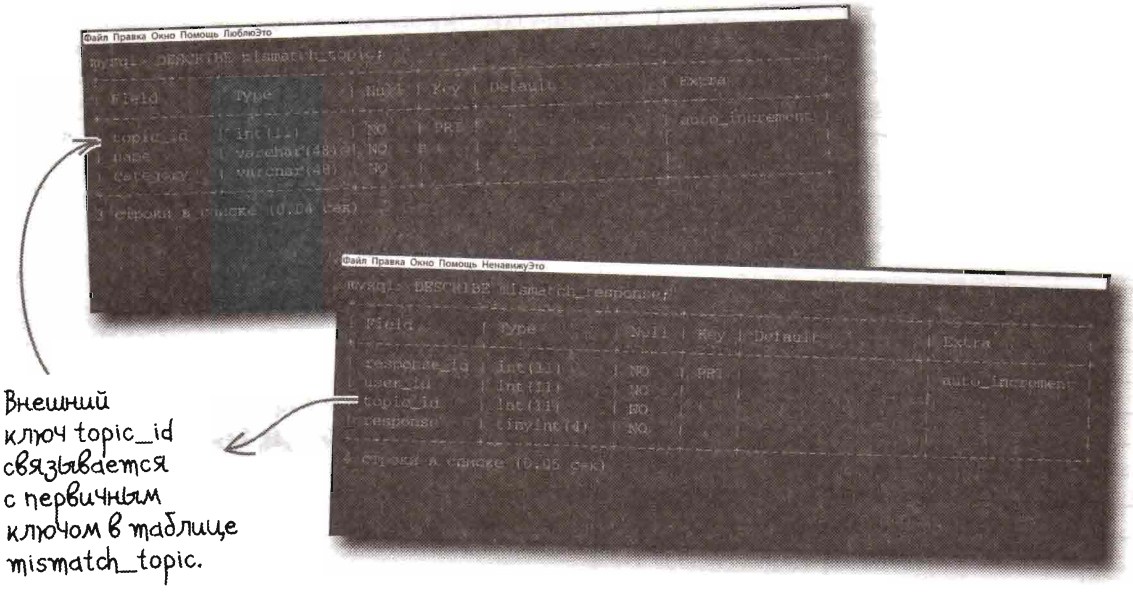
.....один-ко-многим....



Давайте остановимся ненадолго прямо здесь! Потратьте секунду, чтобы привести базу данных приложения «Несоответствия» в порядок: так, чтобы мы могли определять несоответствия.

Загрузите файлы `.sql` для приложения «Несоответствия» с сайта, расположенного по адресу `www.headfirstlabs.com/books/hfphp`. Эти файлы содержат SQL-запросы, необходимые для создания таблиц, используемых приложением «Несоответствия»: `mismatch_user`, `mismatch_topic` и `mismatch_response`. Выполните все эти запросы в любой инструментальной программе MySQL, чтобы создать в базе данных набор таблиц, необходимых для начала работы с приложением «Несоответствия».

Когда все запросы будут выполнены, сделайте запрос `DESCRIBE` для каждой из новых таблиц (`mismatch_topic` и `mismatch_response`) и тщательно просмотрите их структуру. Эти таблицы оказывают исключительное влияние на сценарии приложения «Несоответствия», к работе с которыми мы приступаем.



Отлично. Итак, у нас есть великолепно разработанная база данных, содержащая информацию о пользователях, категории, признаки несоответствия и значения этих признаков. Но как все это поможет нам найти конкретные несоответствия?

Если вы начнете с тщательно разработанной базы данных, остальные части приложения будет создавать и соединять друг с другом уже значительно проще.

Создание оптимальной базы данных в самом начале разработки приложения — это, пожалуй, самое лучшее, что вы можете предпринять, чтобы сделать процесс разработки более логичным и последовательным. Может показаться, что процесс разработки схемы базы данных (составление эскизов визуального представления различных способов того, как будут размещаться данные, с целью выбора наиболее оптимального из них) занимает слишком много времени. Но подумайте о том, насколько усложнится ваша задача, если вы придете к выводу, что структура базы данных выбрана неудачно и требует переделки, в то время как все остальные части приложения полностью согласованы с прежней структурой и, следовательно, также потребуют существенной переделки.

Вот хорошая иллюстрация тех преимуществ, которые вы получаете при тщательной разработке структуры базы данных. В базе данных приложения «Несоответствия» у нас имеется таблица, в которую пользователи самостоятельно заносят данные для входа в приложение, а также данные своего профиля. Кроме того, у нас есть новая таблица, содержащая достаточное количество признаков несоответствия и категории, к которым они относятся, для составления полного представления о человеке. Чего у нас все еще не хватает для нахождения пар с наилучшими несоответствиями, так это предоставления пользователям возможности вводить значения своих признаков несоответствий (пристрастие или отвращение), которые мы могли бы сохранять в таблице базы данных.

mismatch_topic

topic_id	name	category
1	Татуировка	Внешность
2	Золотые цепочки	Внешность
3	Пирсинг	Внешность
4	Ковбойские ботинки	Внешность
5	Длинные волосы	Внешность
6	Реалити-шоу	Времяпрепровождение
7	Профессиональная борьба	Времяпрепровождение
8	Фильмы ужасов	Времяпрепровождение
9	Легкая музыка	Времяпрепровождение
10	Опера	Времяпрепровождение
11	Суши	Еда
12	Спэм	Еда
13	Острые блюда	Еда
14	Сэндвичи с ореховым маслом и бананом	Еда
15	Мартини	Еда
16	Говард Стерн	Люди
17	Билл Гейтс	Люди
18	Барбара Стрейзанд	Люди
19	Хью Хефнер	Люди
20	Марта Стюарт	Люди
21	Йога	Организация досуга
22	Тяжелая атлетика	Организация досуга
23	Кубик Рубика	Организация досуга
24	Караоке	Организация досуга
25	Путешествия	Организация досуга

В таблице mismatch_topic содержится 25 признаков несоответствия, распределенных между пятью категориями... Это наши «5 уровней возможного несоответствия»!

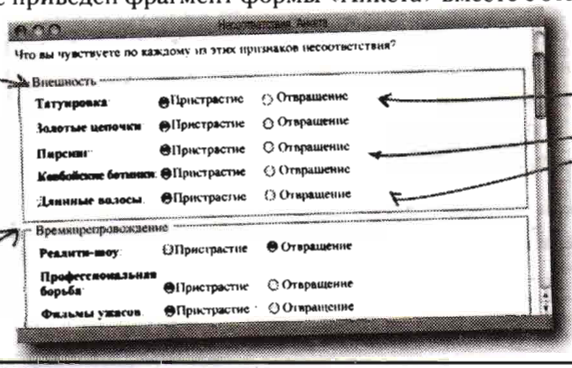
СИЛА МЫСЛИ

Как бы вы преобразовали этот список признаков несоответствия и их категорий в перечень вопросов, на которые пользователь мог бы давать ответы вида «пристрастие/отвращение»?

Создание анкеты приложения «Несоответствия»

Итак, как же нам получить от пользователей ответы на вопросы об их отношении к каждому из признаков несоответствия? Ответ заключается в создании формы в виде анкеты, в которой пользователь может выбрать одно из двух значений (пристрастие или отвращение) для каждого признака несоответствия, содержащегося в таблице `mismatch_topic`. Эта форма может быть создана с использованием признаков несоответствия, уже имеющихся в базе данных, с сохранением новых или измененных данных. Фактически задача, стоящая перед сценарием, ответственным за эту форму, заключается в чтении значений признаков из таблицы `mismatch_response` и записи измененных или новых значений этих признаков в таблицу. Ниже приведен фрагмент формы «Анкета» вместе с этапами ее разработки.

Для группирования близких по смыслу признаков несоответствия они подразделяются по принадлежности к различным категориям.



По каждому из признаков несоответствия в таблице `mismatch_response` имеется запись со значениями «пристрастие/отвращение» для всех пользователей.

1 Используйте SQL-запрос INSERT, чтобы добавить записи в таблицу с пустыми значениями признаков несоответствия для каждого нового пользователя, открывшего форму.

Мы намерены генерировать формы анкеты, исходя из данных соответствующих колонок таблицы `mismatch_response`, даже если пользователь еще не вводил ни одного значения признака несоответствия. Это значит, что мы должны добавить в таблицу `mismatch_response` записи для всех признаков несоответствия, но с пустыми значениями этих признаков в момент, когда пользователь впервые открывает форму «Анкета». Так как колонки `response` для этих записей содержат пустые значения, ни одна из кнопок с зависимой фиксацией («Пристрастие» и «Отвращение») не будет нажата, когда форма будет представлена пользователю впервые.

2 Используйте SQL-запрос UPDATE для изменения значений признаков в соответствии со значениями, введенными пользователем в форму.

Как только данные формы будут отправлены на сервер для обработки, мы должны внести в базу данных значения признаков несоответствия, введенные пользователем. Но при этом изменены будут только те значения признаков, для которых нажата одна из кнопок с зависимой фиксацией. Иначе говоря, база данных имеет дело только с теми признаками несоответствия, для которых от пользователя получен ответ.

3 Используйте SQL-запрос SELECT для извлечения данных таблицы mismatch_response, необходимых для создания формы «Анкета».

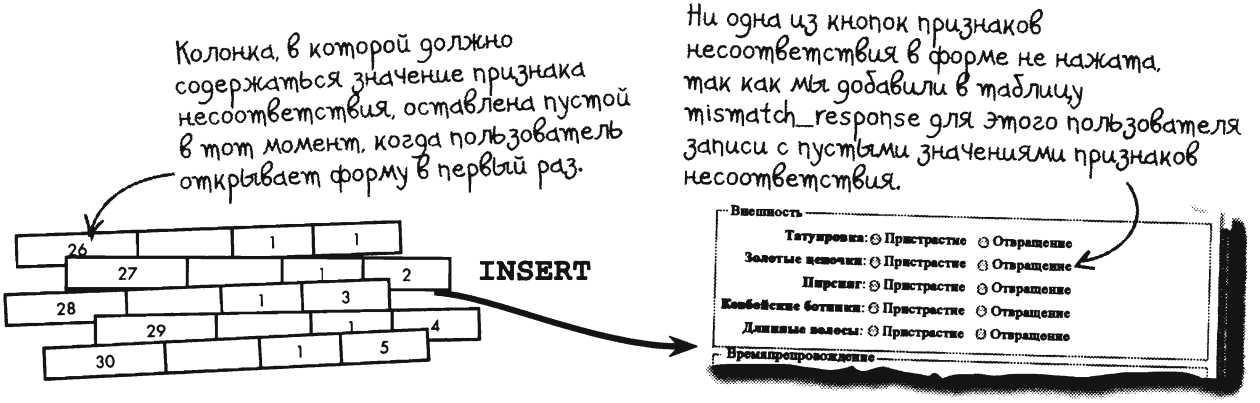
Для создания формы «Анкета» нам необходимы значения всех признаков несоответствия для пользователя. Но это еще не все. Мы также должны найти категории и имена всех признаков несоответствия для того, чтобы вывести эти данные в форме. Они имеются в таблице `mismatch_topic`, а не в таблице `mismatch_response`.

4 Создайте HTML-форму «Анкета», используя все данные признаков несоответствия.

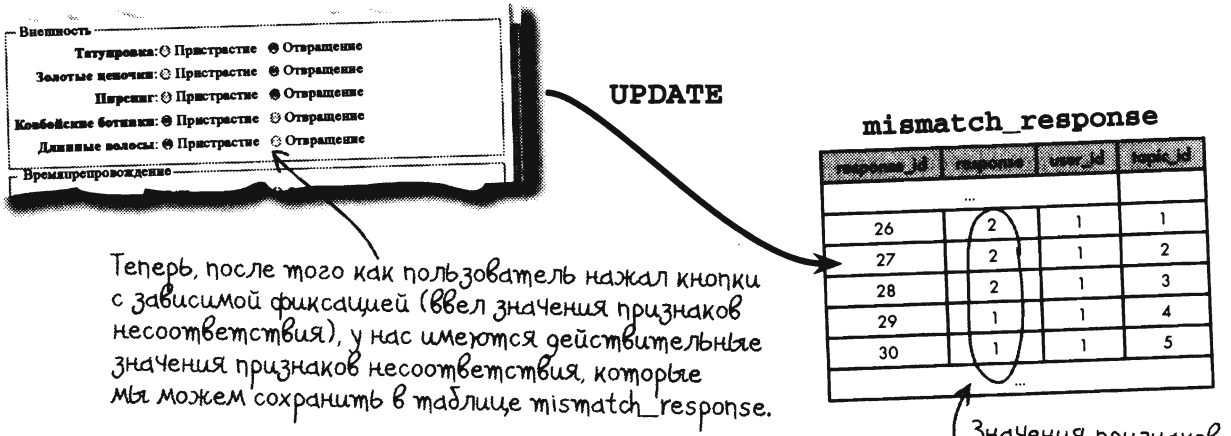
Имея все данные признаков несоответствия, мы можем создать HTML-форму «Анкета», включив в нее группу пар кнопок с зависимой фиксацией для каждого признака, оставляя нажатой либо кнопку «Пристрастие», либо «Отвращение» в зависимости от значения соответствующего признака.

Добавление признаков несоответствия в базу данных

Может показаться, что нам следует начать с создания формы «Анкета», но она сильно зависит от признаков несоответствия, имеющихся в таблице `mismatch_response`. Поэтому мы начнем с главного: нам необходимо добавить в таблицу `mismatch_response` записи для всех признаков несоответствия, но с пустыми значениями этих признаков в момент, когда пользователь впервые открывает форму «Анкета». Это позволит нам создавать форму, используя данные таблицы `mismatch_response`, не думая о том, ввел уже пользователь какие-либо значения признаков несоответствия или еще нет.



Таким образом, с точки зрения формы «Анкета» в таблице `mismatch_response` всегда имеется запись для каждого из признаков несоответствия. Из этого следует, что, когда пользователь отправит заполненную форму на сервер для обработки, все, что нам нужно будет сделать, — это всего лишь изменить записи для каждого из признаков несоответствия, исходя из данных формы.



Хотя в конечном счете сохранение признаков несоответствия в базе данных приложения «Несоответствия» является процессом двухэтапным, первый этап (`INSERT`) для каждого пользователя осуществляется только один раз. И раз первоначально записи с пустыми значениями признаков несоответствия введены, все последующие изменения этих данных, проводимые через форму «Анкета», осуществляются на втором этапе с использованием SQL-запроса `UPDATE`.

Магниты PHP и MySQL

В результате выполнения кода, приведенного ниже, при первом открытии пользователем формы «Анкета» в таблицу `mismatch_user` будут добавлены записи с пустыми значениями колонок, в которых должны содержаться значения признаков несоответствия. В случае, если пользователь изменил значения какого-либо признака несоответствия и отправил форму на сервер для обработки, код также предусматривает обновление таблицы. К сожалению, часть кода утрачена и требуется произвести корректировку. Используя магниты, восстановите утраченный код.

```
...
  // Если этот пользователь ни разу не вводил данные в анкету,
  // добавление записей с пустыми значениями признаков несоответствия
  // в таблицу mismatch_response
  $query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";
  $data = mysqli_query($dbc, $query);
  if (_____($data) == 0) {
    // Извлечение списка идентификаторов признаков несоответствия из таблицы mismatch_topic
    $query = "SELECT _____FROM mismatch_topic ORDER BY category_id, topic_id";
    $data = mysqli_query($dbc, $query);
    $topicIDs = array();
    while ($row = mysqli_fetch_array($data)) {
      array_push($topicIDs, $row['topic_id']);
    }

    // Добавление записей с пустыми значениями
    // признаков несоответствия в таблицу mismatch_response
    foreach ($topicIDs as $topic_id) {
      $query = "_____mismatch_response
        (_____, _____) VALUES ('" . $_SESSION['user_id']. "', '$topic_id')";
      mysqli_query($dbc, $query);
    }
  }

  // Если форма «Анкета» отправлена на сервер для обработки,
  // обновление признаков несоответствия в таблице mismatch_response
  if (isset($_POST['submit'])) {
    // Обновление признаков несоответствия в таблице mismatch_response
    foreach ($_POST as $response_id => $response) {
      $query = "_____mismatch_response_____response = '$response' WHERE
_____= '$response_id'";
      mysqli_query($dbc, $query);
    }
    echo '<p>Ваши признаки несоответствия сохранены.</p>';
  }
...
```

user_id topic_id INSERT INTO mysqli_num_rows SET topic_id response_id UPDATE

Магниты PHP и MySQL

В результате выполнения кода, приведенного ниже, при первом открытии пользователем формы «Анкета» в таблицу `mismatch_user` будут добавлены записи с пустыми значениями колонок, в которых должны содержаться значения признаков несоответствия. В случае, если пользователь изменил значения какого-либо признака несоответствия и отправил форму на сервер для обработки, код также предусматривает обновление таблицы. К сожалению, часть кода утрачена и требуется произвести корректировку. Используя магниты, восстановите утраченный код.

```
...
// Если этот пользователь ни разу не вводил данные в анкету,
// добавление записей с пустыми значениями признаков несоответствия
// в таблицу mismatch_response
$query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
if [mysqli_num_rows] ata) == 0) {
  // Извлечение списка идентификаторов признаков несоответствия из таблицы mismatch_topic
  $query = "SELECT [topic_id] ROM mismatch_topic ORDER BY category_id, topic_id";
  $data = mysqli_query($dbc, $query);
  $topicIDs = array();
  while ($row = mysqli_fetch_array($data)) {
    array_push($topicIDs, $row['topic_id']);
  }

  // Добавление записей с пустыми значениями
  // признаков несоответствия в таблицу mismatch_response
  foreach ($topicIDs as $topic_id) {
    $query = [INSERT INTO] mismatch_response
      ([user_id], [topic_id]) VALUES ('" $_SESSION['user_id']. "', '$topic_id')";
    mysqli_query($dbc, $query);
  }
}

// Если форма «Анкета» отправлена на сервер для обработки,
// обновление признаков несоответствия в таблице mismatch_response
if (isset($_POST['submit'])) {
  // Обновление признаков несоответствия в таблице mismatch_response
  foreach ($_P[UPDATE] response_id => $response) {
    $query = "[SET] sponse = '$response' WHERE
[response_id] $response_id'";
    mysqli_query($dbc, $query);
  }
  echo '<p>Ваши признаки несоответствия сохранены.</p>';
}
...
```

Проверка, имеются ли записи с такими данными в таблице.

Для того чтобы создать массив с пустыми значениями признаков несоответствия, вначале нужно извлечь все признаки несоответствия из таблицы mismatch_topic.

К этому моменту колонка response во всех записях пуста, так как пользователь еще не нажал ни одной кнопки с зависимой фиксацией на форме (не определил свое отношение к признакам несоответствия).

Все, что изменяет свое значение, когда пользователь отправляет форму на сервер для обработки, — это колонка response таблицы mismatch_response.

не бывает глупых вопросов

В: Что это за функция — array_push()? Я не помню, чтобы мы использовали ее раньше.

О: Мы еще не использовали эту функцию. И это просто потому, что у нас еще не было необходимости создавать массив динамически по одному элементу за раз. Функция `array_push()` добавляет новый элемент в конец массива, в результате чего количество элементов в массиве увеличивается на единицу. В коде сценария, приведенного на предыдущей странице, мы используем функцию `array_push()` для создания массива идентификаторов признаков несоответствия, извлекая их из таблицы `mismatch_topic`. Этот массив используется затем для добавления в таблицу `mismatch_response` записей с пустыми значениями признаков несоответствия… по одной на каждый признак.

1 **Используйте SQL-запрос INSERT, чтобы добавить записи в таблицу с пустыми значениями признаков несоответствия для каждого нового пользователя, открывшего форму.**

СДЕЛАНО

Ого! Мы только что убили двух виртуальных зайцев, и теперь сценарий нашей анкеты наполовину готов.

СДЕЛАНО

2 **Используйте SQL-запрос UPDATE для изменения значений признаков в соответствии со значениями, введенными пользователем в форму.**

Но осталось выполнить еще два этапа, прежде чем мы сможем использовать нашу анкету.

3 **Используйте SQL-запрос SELECT для извлечения данных таблицы mismatch_response, необходимых для создания формы «Анкета».**

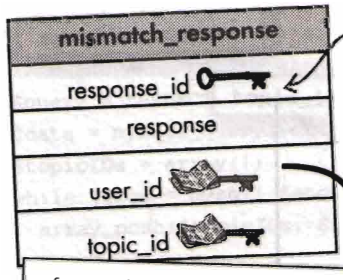
4 **Создайте HTML-форму «Анкета», используя все данные признаков несоответствия.**

Данные могут управлять созданием формы

Нет ничего нового в том, что формы используются для получения данных от пользователя через текстовые поля, списки выбора, кнопки с зависимой фиксацией и т. д., но не совсем очевидно то, что вы можете генерировать HTML-формы, используя данные базы и PHP. В приложении «Несоответствия» HTML-форма «Анкета» генерируется на основании данных таблицы `mismatch_response`. Сценарий, создающий форму «Анкета», исходит из того, что признаки несоответствия уже существуют в таблице `mismatch_response`, и это позволяет ему использовать эти данные при создании формы. Мы знаем, что такой подход оправдан, потому что мы только что написали код, ответственный за добавление в таблицу `mismatch_response` записей с пустыми колонками `response` при самом первом открытии пользователем формы.

Формы, управляемые данными, полагаются на информацию, содержащуюся в базе данных MySQL, при создании полей ввода.

Поля ввода данных связаны с записями таблицы базы данных путем присвоения именам каждого поля значения первичного ключа таблицы базы данных.

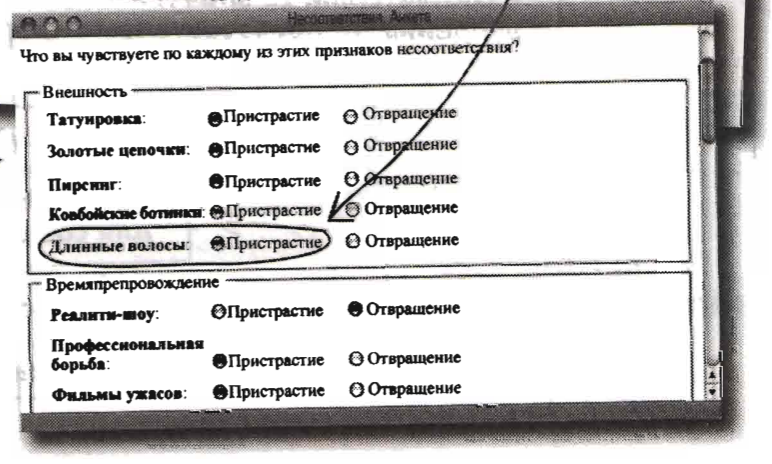


Первичный ключ response_id используется для того, чтобы уникально идентифицировать поля ввода данных HTML-формы и связать каждое такое поле с записью в таблице базы данных.

HTML-код формы генерируется на основании данных таблицы mismatch_response.

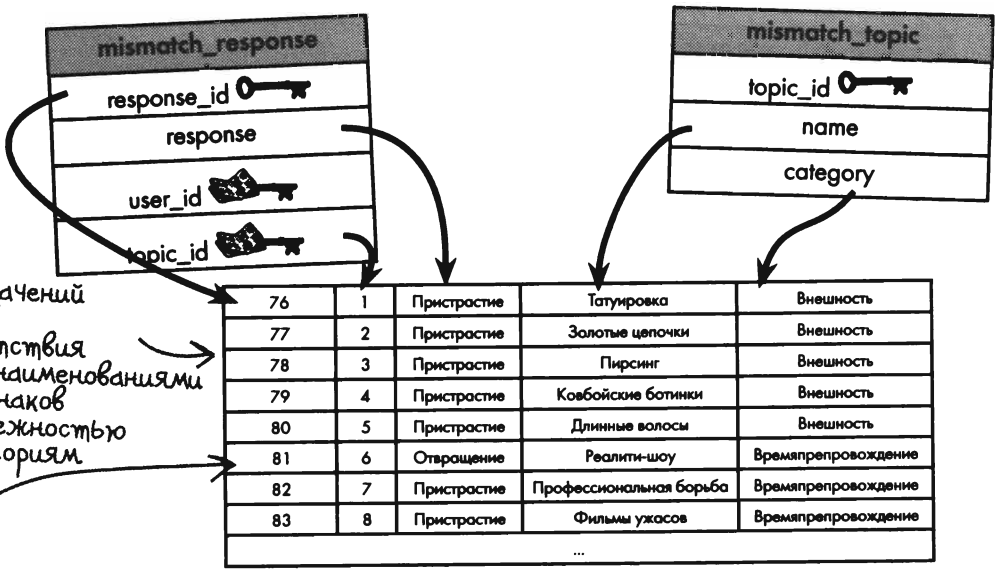
```
<form method="post" action="">
  <p>Что вы чувствуете по каждому из этих признаков несоответствия?</p>
  <fieldset>
    <legend>Внешность</legend>
   <label for="76">Татуировка:</label>
      <input type="radio" id="76" name="76" value="1" checked="checked" />Пристрастие
      <input type="radio" id="76" name="76" value="2"  />Отвращение<br />
   <label for="77">Золотые цепочки:</label>
      <input type="radio" id="77" name="77" value="1" checked="checked" />Пристрастие
      <input type="radio" id="77" name="77" value="2"  />Отвращение<br />
   <label for="78">Пирсинг:</label>
      <input type="radio" id="78" name="78" value="1" checked="checked" />Пристрастие
      <input type="radio" id="78" name="78" value="2"  />Отвращение<br />
   <label for="79">Ковбойские ботинки:</label>
      <input type="radio" id="79" name="79" value="1" checked="checked" />Пристрастие
      <input type="radio" id="79" name="79" value="2"  />Отвращение<br />
   <label for="80">Длинные волосы:</label>
      <input type="radio" id="80" name="80" value="1" checked="checked" />Пристрастие
      <input type="radio" id="80" name="80" value="2"  />Отвращение<br />
  </fieldset>
  <fieldset>
    <legend>Времяпрепровождение</legend>
    ...
</form>
```

Атрибут checked управляет выбором кнопки с зависимой фиксацией.



Форма отражает выбор пользователем одного из двух значений по каждому признаку несоответствия.

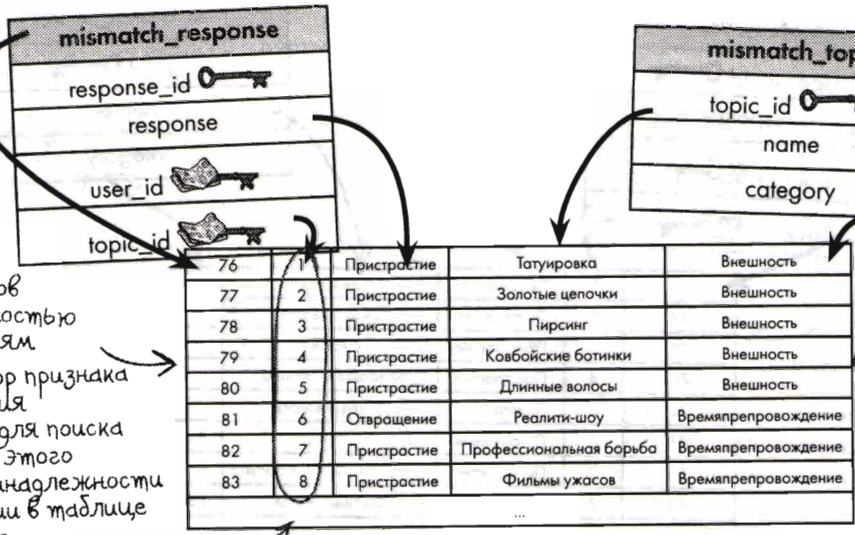
Анкета приложения «Несоответствия» генерируется на основании значений признаков несоответствия, сохраненных в таблице `mismatch_response`. Для того чтобы создать код HTML-формы, необходимо извлечь значения признаков из этой таблицы, а также наименования и принадлежность к категории для каждого признака. В результате выполнения фрагмента кода, приведенного ниже, создается массив признаков несоответствия с их наименованиями и принадлежностью к категориям путем выполнения двух SQL-запросов. В результате выполнения первого запроса извлекаются значения признаков несоответствий для пользователя, в то время как второй ответственен за поиск наименований признаков и принадлежности их к категориям. Проблема заключается в том, что часть кода утрачена… Заполните пропущенные места, чтобы привести этот фрагмент кода в рабочее состояние!




```
// Извлечение данных из базы для создания формы
$query = "SELECT response_id, topic_id, response FROM mismatch_response " .
                "WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
  // Извлечение наименования признаков несоответствия и принадлежности их
к категориям
  // из таблицы mismatch_topic
  $query2 = " ..... ";
                "WHERE topic_id = '" . $row['topic_id'] . "'";
  $data2 = mysqli_query($dbc, ..... );
  if (mysqli_num_rows( ..... ) == 1) {
    $row2 = mysqli_fetch_array($data2);
    $row['topic_name'] = .....
    $row['category_name'] = .....
    array_push($responses, $row);
  }
}
```

Эта PHP-функция говорит вам о том, какое количество записей получено в результате выполнения запроса.

Анкета приложения «Несоответствия» генерируется на основании значений признаков несоответствия, сохраненных в таблице `mismatch_response`. Для того чтобы создать код HTML-формы, необходимо извлечь значения признаков из этой таблицы, а также наименования и принадлежность к категории для каждого признака. В результате выполнения фрагмента кода, приведенного ниже, создается массив признаков несоответствия с их наименованиями и принадлежностью к категориям путем выполнения двух SQL-запросов. В результате выполнения первого запроса извлекаются значения признаков несоответствий для пользователя, в то время как второй ответственен за поиск наименований признаков и принадлежности их к категориям. Проблема заключается в том, что часть кода утрачена… Заполните пропущенные места, чтобы привести этот фрагмент кода в рабочее состояние!



Массив значений признаков несоответствия дополнен наименованиями этих признаков и принадлежностью их к категориям

Идентификатор признака несоответствия используется для поиска наименования этого признака и принадлежности его к категории в таблице mismatch_topic.

Массив $responses служит в качестве временной таблицы с данными признаков несоответствия, используемых для создания формы «Анкета».

```
// Извлечение данных из базы для создания формы
$query = "SELECT response_id, topic_id, response FROM mismatch_response " .
                 "WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
  // Извлечение наименования признаков несоответствия и принадлежности их
к категориям
  // из таблицы mismatch_topic
  $query2 = "SELECT name, category FROM mismatch_topic " .
                 "WHERE topic_id = '" . $row['topic_id'] . "'";
  $data2 = mysqli_query($dbc, $query2);
  if (mysqli_num_rows($data2) == 1) {
    $row2 = mysqli_fetch_array($data2);
    $row['topic_name'] = $row2['name'];
    $row['category_name'] = $row2['category'];
    array_push($responses, $row);
  }
}
```

Очень важно использовать новую переменную для сохранения результатов второго (внутреннего) запроса, чтобы не оказывать влияния на результаты оригинального запроса.

Убеждаемся, что данные признаков несоответствия действительно существуют.

Функция array_push() (push в переводе с английского языка «заталкивать») добавляет элемент в конец массива.

Наименование признака несоответствия и категория, которой он принадлежит, добавлены в массив признаков несоответствия присвоением соответствующих значений, взятых из результатов второго запроса.

СДЕЛАНО

3 **Использование SQL-запрос SELECT для извлечения данных таблицы mismatch_response, необходимых для создания формы «Анкета».**

Значения признаков несоответствия сохраняются в базе данных в текстовом формате, в виде строчных переменных «Пристрастие» и «Отвращение»? Если это так — разве это эффективно?

И нет, и да. Поэтому очень важно использовать, по возможности, наиболее эффективные типы для сохранения данных в базе MySQL.

Если посмотреть более внимательно, тип данных признаков несоответствия в приложении «Несоответствия» очень напоминает булев тип (true/false — истина/ложь), потому что эти признаки могут принимать либо одно значение (пристрастие), либо другое (отвращение). Полезным является и третье значение (неизвестно), так как оно дает приложению знать, что пользователь еще не вносил данные по определенному признаку несоответствия при заполнении анкеты (или еще не заполнял ее совсем). Идеальным вариантом для подобных данных является сохранение их в виде числа, например типа `TINYINT`. В этом случае вы просто используете различные числа для представления различных значений признаков несоответствия.

Неизвестно = 0 Пристрастие = 1 Отвращение = 2

Минимизация ресурсов сервера, необходимых для сохранения данных, является очень важным требованием, которое необходимо соблюдать при разработке структуры базы данных. Выбранный нами тип данных `TINYINT` для сохранения значений признаков несоответствия вносит хотя и небольшой, но важный вклад в повышение общей эффективности приложения «Несоответствия». Эти числовые значения признаков играют непосредственную роль в создании полей ввода данных в форме «Анкета».

Решение задачи

Не беспокойтесь пока по поводу кнопок «Отвращение». Они генерируются точно так же.

Минимизация ресурсов сервера, необходимых для сохранения данных, является очень важным требованием, которое необходимо соблюдать при разработке структуры базы данных. Выбранный нами тип данных `TINYINT` для сохранения значений признаков несоответствия вносит хотя и небольшой, но важный вклад в повышение общей эффективности приложения «Несоответствия». Эти числовые значения признаков играют непосредственную роль в создании полей ввода данных в форме «Анкета».

```
foreach ($responses as $response) {
  …
  if (.....) {
    echo '<input type="radio" name="' . $response['response_id'] .
         '" value=..... checked=...../>Пристрастие ';
  }
  else {
    echo '<input type="radio" name="' . $response['response_id'] .
         '" value=...../>Пристрастие ';
  }
}
```

Решение задачи

Минимизация ресурсов сервера, необходимых для сохранения данных, является очень важным требованием, которое необходимо соблюдать при разработке структуры базы данных. Выбранный нами тип данных TINYINT для сохранения значений признаков несоответствия вносит хотя и небольшой, но важный вклад в повышение общей эффективности приложения «Несоответствия». Эти числовые значения признаков играют непосредственную роль в создании полей ввода данных в форме «Анкета».

Состояние кнопки с зависимой фиксацией «Пристрастие» определяется значением признака несоответствия (значение 1, сохраненное в базе данных, соответствует значению «Пристрастие»).

Если значение этого признака несоответствия равно 1, присвойте атрибуту checked тега <input> значение checked.

```
foreach ($responses as $response) {
  …
  if ($response['response'] == 1) {
    echo '<input type="radio" name="' . $response['response_id'] .
         '" value="1" checked="checked" />Пристрастие ';
  }
  else {
    echo '<input type="radio" name="' . $response['response_id'] .
         '" value="1" />Пристрастие ';
  }
}
```

Атрибут value тега <input> принимает значение 1, что повышает эффективность процесса доступа к этим данным в базе.

Удаление выражения checked="checked" приводит к тому, что кнопка не будет нажата (выбрана), если значение признака несоответствия не равно 1.

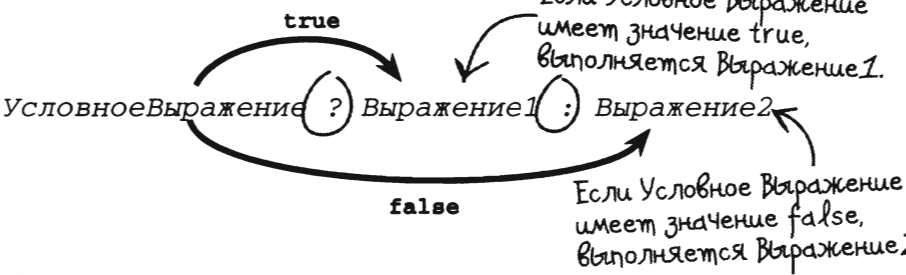
```
foreach ($responses as $response) {
  …
  if ($response['response'] == 2) {
    echo '<input type="radio" name="' . $response['response_id'] .
         '" value="2" checked="checked" />Отвращение ';
  }
  else {
    echo '<input type="radio" name="' . $response['response_id'] .
         '" value="2" />Отвращение ';
  }
}
```

Если вам интересно, код, генерирующий кнопку «Отвращение», вторую кнопку из группы кнопок с зависимой фиксацией, работает точно так же. Он только проверяет немного другое значение признака несоответствия...

Говоря об эффективности...

Эффективность базы данных не является единственным видом эффективности, которую следует принимать во внимание. Существует также понятие **эффективности кода**, которая может проявляться в разных формах. Одна из возможностей заключается в использовании сокращенного варианта управляющей конструкции `if-else`. **Тернарный (трехчленный) оператор** `?:` предлагает удобный способ составления простых управляющих конструкций `if-else` в более компактном виде.

Тернарный (трехчленный) оператор ?: может быть использован для кодирования простых управляющих конструкций типа if-else в более компактной форме.



Тернарный оператор `?:` фактически является всего лишь сокращенным вариантом управляющей конструкции `if-else`. Он может оказаться полезным для упрощения конструкции `if-else`, особенно в тех случаях, когда вы присваиваете значения переменным или генерируете HTML-код, зависящий от значений условных выражений. Ниже представлен код, генерирующий в форме код кнопки «Пристрастие», переписанный с использованием тернарного оператора `?:`.

```
}
echo '<input type="radio" name="' . $response['response_id'] . '" value="1"' .
    ($response['response'] == 1 ? 'checked="checked"' : '') . ' />Пристрастие ';
```

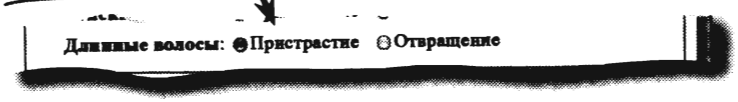
Значение этого условного выражения определяет значение, возвращаемое тернарным оператором ?:.

Значение атрибута checked тега <input> теперь определяется тернарным оператором ?:, а не управляющей конструкцией if-else.

Если значение признака несоответствия, присвоенное переменной `$response['response']` равно 1, то атрибут `checked` будет включен в состав тега `<input>` со значением `checked`, что приведет к тому, что кнопка «Пристрастие», первая из двух кнопок с зависимой фиксацией, будет выведена в форме в нажатом состоянии.

```
<input type="radio" name="279" value="1" checked="checked" /> Пристрастие
```

Вывод в форме этой части тега <input> контролируется тернарным оператором ?:.



С другой стороны, любое значение признака несоответствия, кроме единицы, предотвратит включение атрибута `checked` в состав тега `<input>`, и это приведет к тому, что кнопка «Пристрастие» будет выведена в форме в ненажатом состоянии.

Создание формы «Анкета» приложения «Несоответствия»

Теперь у нас есть достаточно фрагментов кода для того, чтобы полностью составить из них форму «Анкета» приложения «Несоответствия», используя при этом массив данных признаков несоответствия (`$responses`), который мы создали ранее. Если вы помните, этот массив был создан путем извлечения текущих значений признаков несоответствия из таблицы `mismatch_response`. Давайте пойдем дальше и посмотрим, как будет создаваться форма при выполнении сценария `questionnaire.php`.

questionnaire.php

```
<?php
  // Открытие сессии
  require_once('startsession.php');

  // Вывод заголовка страницы
  $page_title = 'Анкета';
  require_once('header.php');

  require_once('appvars.php');
  require_once('connectvars.php');

 // Прежде чем продолжать сценарий, необходимо проверить, вошел ли пользователь в приложение
  if (!isset($_SESSION['user_id'])) {
    echo '<p class="login">Пожалуйста, <a href="login.php">войдите в приложение </a>' .
         'чтобы получить доступ к этой странице.</p>';
    exit();
  }

 // Вывод навигационного меню
  require_once('navmenu.php');

  // Соединение с базой данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  // Если пользователь еще не вводил ни одного признака несоответствия в анкету, добавление в таблицу
  // базы данных записей с пустыми значениями признаков несоответствия
  $query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";
  $data = mysqli_query($dbc, $query);
  if (mysqli_num_rows($data) == 0) {
    // Вначале извлечение списка идентификаторов признаков несоответствия из таблицы mismatch_topic
    $query = "SELECT topic_id FROM mismatch_topic ORDER BY category_id, topic_id";
    $data = mysqli_query($dbc, $query);
    $topicIDs = array();
    while ($row = mysqli_fetch_array($data)) {
      array_push($topicIDs, $row['topic_id']);
    }

   // Добавление записей с пустыми значениями признаков несоответствия в таблицу mismatch_response
   foreach ($topicIDs as $topic_id) {
      $query = "INSERT INTO mismatch_response (user_id, topic_id) VALUES ('" . $_SESSION['user_id']. "',
'$topic_id')";
      mysqli_query($dbc, $query);
    }

  // Если форма «Анкета» отправлена на сервер для обработки,
  // обновление признаков несоответствия в таблице mismatch_response
  if (isset($_POST['submit'])) {
    // Обновление признаков несоответствия в таблице mismatch_response
    foreach ($_POST as $response_id => $response) {
      $query = "UPDATE mismatch_response SET response = '$response' " .
```

Включение добавляемых файлов открытия сессии и вывода заголовка страницы.

Ограничение доступа к странице для тех пользователей, которые не вошли в приложение.

1

2

```
      "WHERE response_id = '$response_id'";
    mysqli_query($dbc, $query);
  }
  echo '<p> Ваши признаки несоответствия сохранены.</p>';
}

// Извлечение данных признаков несоответствия из базы для создания формы
$query = "SELECT response_id, topic_id, response FROM mismatch_response WHERE user_id = '" . $_
SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
  // Поиск имен признаков несоответствия в таблице mismatch_topic для создания формы
  $query2 = "SELECT name, category FROM mismatch_topic WHERE topic_id = '" . $row['topic_id'] .
"'";
  $data2 = mysqli_query($dbc, $query2);
  if (mysqli_num_rows($data2) == 1) {
    $row2 = mysqli_fetch_array($data2);
    $row['topic_name'] = $row2['name'];
    $row['category_name'] = $row2['category'];
    array_push($responses, $row);
  }
}

mysqli_close($dbc);
```

3

Прежде чем войти в цикл, необходимо извлечь наименование категории первого признака несоответствия.

Для каждой группы признаков несоответствия, принадлежащих к одной категории, создается своя группа с использованием тега <fieldset>.

```
// Создание формы «Анкета» путем прохождения в цикле
// массива с данными признаков несоответствия
echo '<form method="post" action="' . $_SERVER['PHP_SELF'] . '">';
echo '<p> Что вы чувствуете по каждому из этих признаков несоответствия?</p>';
$category = $responses[0]['category_name'];
echo '<fieldset><legend>' . $responses[0]['category_name'] . '</legend>';
foreach ($responses as $response) {
  // Начинайте новую группу признаков несоответствия только в том случае,
  // если изменилась категория, к которой они относятся
  if ($category != $response['category_name']) {
    $category = $response['category_name'];
    echo '</fieldset><fieldset><legend>' . $response['category_name'] . '</legend>';
  }

// Вывод кнопок с зависимой фиксацией для выбора признаков несоответствия
echo '<label ' . ($response['response'] == NULL ? 'class="error"' : '') . ' for="' .
    $response['response_id'] . '">' . $response['topic_name'] . ':</label>';
echo '<input type="radio" id="' . $response['response_id'] . '" name="' . $response['response_id'] .
  '" value="1" ' . ($response['response'] == 1 ? 'checked="checked"' : '') . ' />Предпочтение ';
echo '<input type="radio" id="' . $response['response_id'] . '" name="' . $response['response_id'] .
  '" value="2" ' . ($response['response'] == 2 ? 'checked="checked"' : '') . ' />Отвращение<br />';
}
echo '</fieldset>';
echo '<input type="submit" value="Сохранение анкеты" name="submit" />';
echo '</form>';
```

4

Здесь используется тернарный оператор ?: для определения состояний кнопок с зависимой фиксацией.

Наименование каждого признака несоответствия создается с помощью тега <label> с последующим выводом пары кнопок с зависимой фиксацией «Пристрастие» и «Отвращение».

```
// Добавление нижнего колонтитула
require_once('footer.php');
?>
```

Запомните:
1 = Пристрастие
2 = Отвращение

В результате вызова каждой из этих двух функций создается пара кнопок. Нажатие первой приводит к сохранению в базе данных значения признака несоответствия «Пристрастие», второй — «Отвращение».

СДЕЛАНО

4 **Создайте HTML-форму «Анкета», используя все данные признаков несоответствия.**

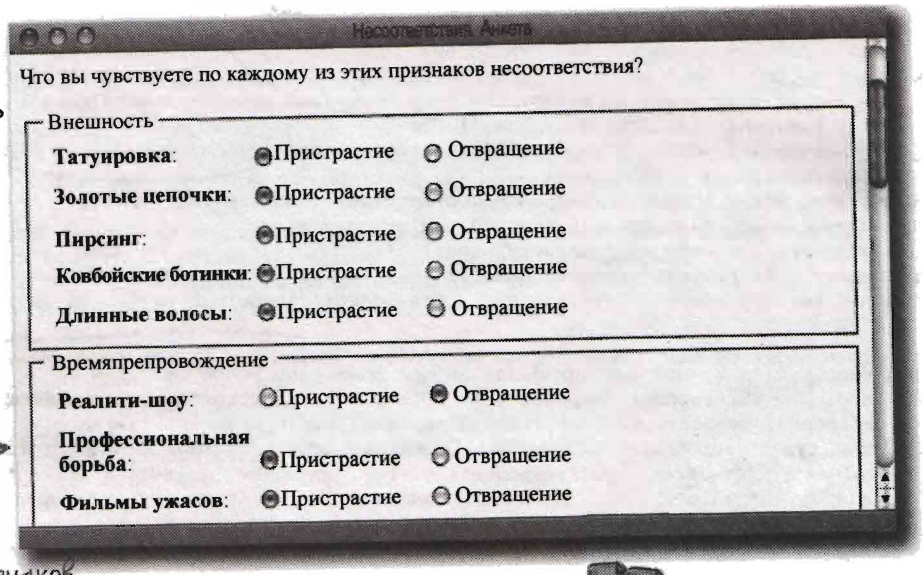
Тест-драйв

Проверьте новый сценарий «Анкета» приложения «Несоответствия».

Внесите в приложение «Несоответствия» изменения, обеспечивающие использование в нем новой формы «Анкета» (или загрузите приложение с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Это требует создания нового сценария `questionnarie.php`, а также добавления строки «Анкета» в сценарий `navmenu.php`, чтобы у пользователя была возможность получить доступ к форме «Анкета».

Загрузите все сценарии на ваш веб-сервер и откройте главную страницу (`index.php`) приложении «Несоответствия» в браузере. Войдите в приложение и щелкните кнопкой мыши на гиперссылке `Анкета`, чтобы получить доступ к форме «Анкета». Обратите внимание на то, что ни одна из кнопок с зависимой фиксацией не нажата (ни один из признаков несоответствия вами не выбран), так как это ваш первый доступ к форме. Нажмите выбранные вами кнопки, а затем кнопку «Сохранение анкеты». Вернитесь к главной странице, а затем откройте форму «Анкета» еще раз, чтобы убедиться, что все выбранные вами в предыдущий раз признаки несоответствия сохранились в базе данных.

Сценарий «Анкета» предоставляет пользователю возможность установить значения признаков несоответствия «пристрастие/отвращение» и сохранить эту информацию в базе данных.



Наименования признаков несоответствия и пары кнопок с зависимой фиксацией для установки их значений генерируются динамически, на основании информации, извлеченной из базы данных. Если вы добавите новый признак несоответствия, форма изменится.

Загрузите это!

Весь код для приложения «Несоответствия» доступен для загрузки на сайте по адресу:

`www.headfirstlabs.com/books/hfphp`

не бывает глупых вопросов

В: **Как код кнопки с зависимой фиксацией «Пристрастие» определяет, что результат, возвращаемый тернарным оператором `?:`, является строкой текста?**

О: Тернарный оператор всегда возвращает одно из двух значений выражений, расположенных по обе стороны двоеточия, основываясь на значении условного выражения (`true`/`false`). Если значения этих выражений — строки текста, то и возвращаемое тернарным оператором значение также будет строкой текста. Это делает оператор очень удобным в использовании. Вы можете вставить его в середину оператора присваивания или конкатенации.

В: **Использование тернарного оператора `?:` ускоряет выполнение сценария?**

О: Похоже, нет. Тернарный оператор скорее повышает стилистическую эффективность текста программы за счет более выразительного и эффективного изложения ее логики. Часто использование этого оператора вместо полнофункциональной управляющей конструкции `if-else` делает программу лаконичнее, хотя обе конструкции логически эквивалентны. Но не слишком увлекайтесь использованием тернарного оператора `?:`, потому что в некоторых случаях он может сделать код более трудным для чтения и понимания: например, если вы захотите заменить им не слишком простую управляющую конструкцию `if-else`. Идея, лежащая в основе его использования, заключается в замене управляющих конструкций `if-else` этим оператором в тех местах, где это позволит упростить код, а не сделает его более сложным для чтения и понимания. Обычно тернарный оператор `?:` используют, когда нужно выбрать одно из двух значений и присвоить его переменной или непосредственно вставить в выражение. В случае кнопок с зависимой фиксацией в приложении «Несоответствия» тернарный оператор используется для того, чтобы выбрать одно из двух значений и вставить его в HTML-тег `<insert>` в то место, где должно быть значение атрибута `checked`.

В: **Как можно генерировать форму «Анкета» в приложении «Несоответствия», используя данные таблицы mismatch_response до того, как пользователь ввел какие-либо данные?**

О: Отличный вопрос. Форма «Анкета» используется в двух различных ситуациях: когда пользователь открывает форму для внесения данных первый раз и когда он открывает форму повторно для внесения изменений в свою анкету. В первом случае пользователь еще не вносил никаких данных, следовательно, и в таблице `mismatch_response` никаких данных по нему еще нет. Но независимо от этого нам необходимо динамически создать форму. Мы могли бы использовать данные таблицы `mismatch_topic` для такого первоначального создания формы. Но этот способ не годится для всех последующих случаев открытия формы, так как нам необходимо отразить в форме значения всех признаков несоответствия, которые пользователь уже ввел во время предыдущих открытий формы. Не забывайте, что кнопки с зависимой фиксацией «Пристрастие»/«Отвращение» создаются как часть формы, а их состояние (нажата/снята) зависит от значения признака несоответствия. Перед нами возникает проблема, связанная с тем, что код, ответственный за создание формы, будет различным в зависимости от того, вносил уже пользователь данные в форму или еще нет. Более того, а что делать, если он внес только часть данных? Все это очень запутывает дело. Решение, выбранное в приложении «Несоответствия», заключается в предварительном внесении записей в таблицу `mismatch_response` для всех наименований признаков несоответствия, но с пустыми значениями этих признаков при первом открытии пользователем формы. Это дает нам возможность всегда создавать форму, основываясь на данных таблицы `mismatch_response`, и не усложнять код, используя различные алгоритмы создания формы в зависимости от того, вносил ли уже пользователь признаки несоответствия и какие конкретно. Конечно, код создания формы все равно получается не слишком простым, но все же значительно проще, чем если бы мы использовали данные не только таблицы `mismatch_response`.

Проверьте это!

Для упрощения в приложении «Несоответствия» не предусмотрено автоматическое согласование данных при добавлении нового признака несоответствия, по крайней мере, для записей тех пользователей, которые уже вносили данные в анкету. Поэтому необходимо удалить все записи в таблице mismatch_response после добавления нового признака несоответствия.

1. **Добавьте новый признак несоответствия в вашу собственную таблицу mismatch_topic с помощью SQL-запроса:**

```
INSERT INTO mismatch_topic (name, category)
   VALUES ('Виртуальные гитары', 'Времяпрепровождение')
```

2. **Удалите все записи из таблицы mismatch_response с помощью SQL-запроса:**

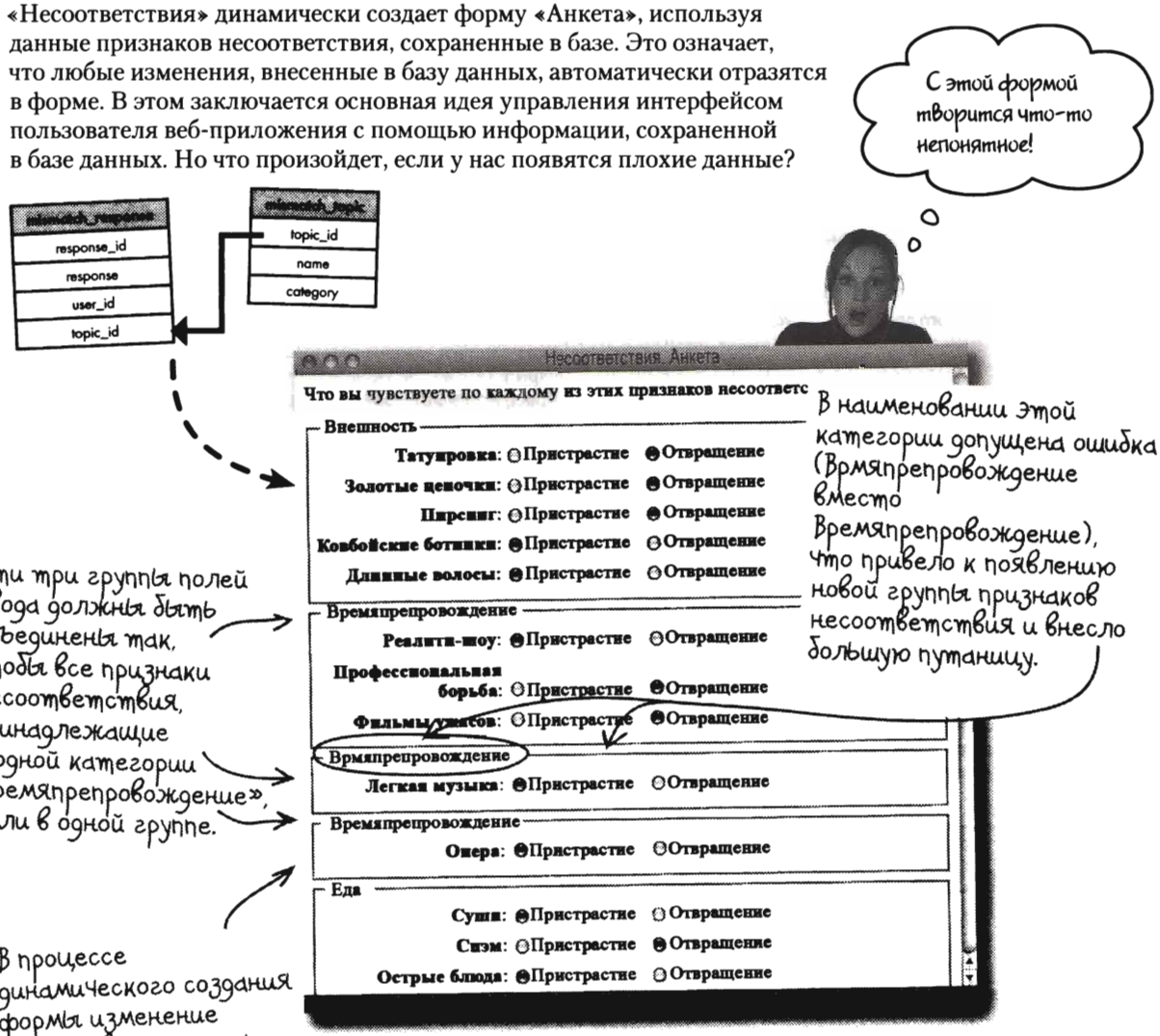
```
DELETE FROM mismatch_response
```

3. **Убедитесь в том, что в форме «Анкета» приложения «Несоответствия» появилось поле ввода данных для нового признака несоответствия.**

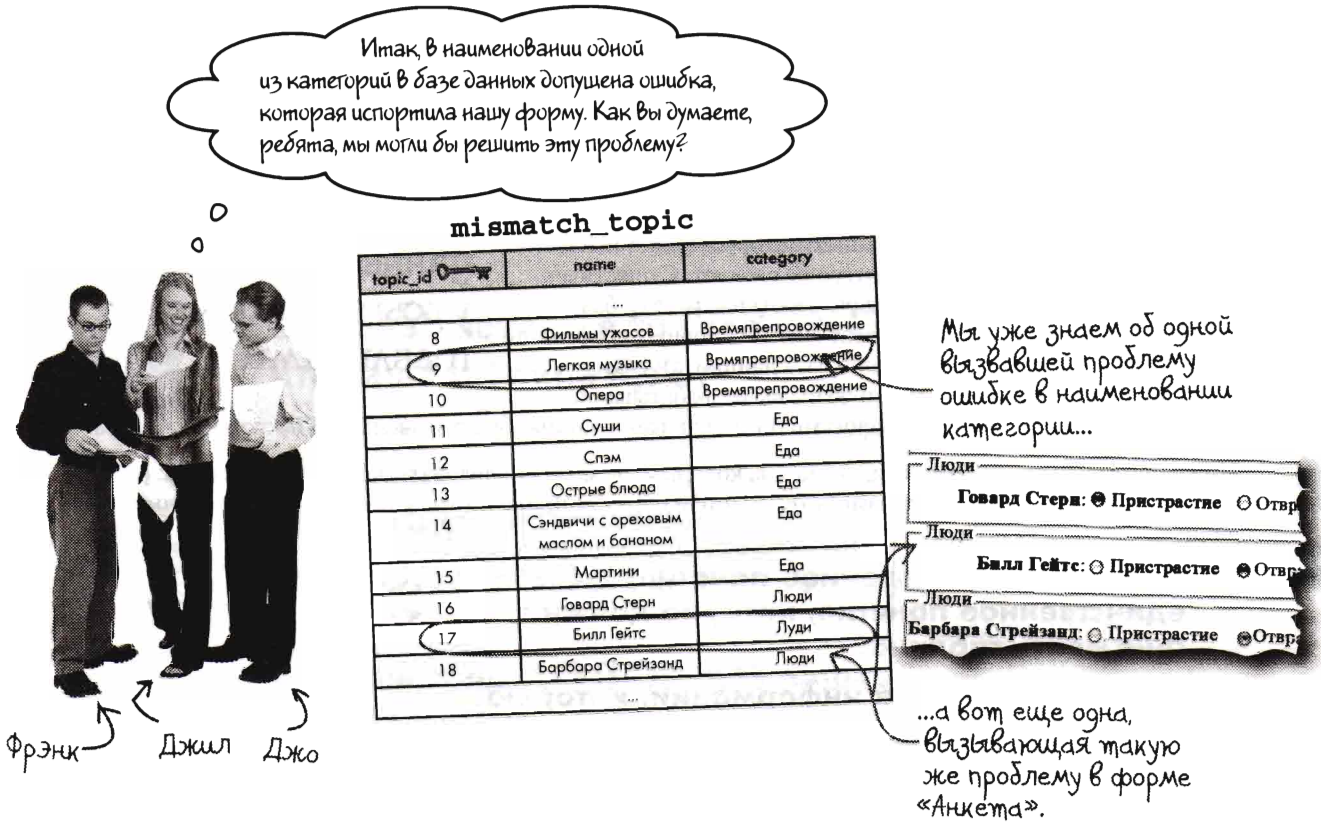
4. **Введите какое-либо значение для нового признака несоответствия, отправьте форму на сервер для обработки и проверьте, сохранилось ли введенное вами значение.**

Теперь данные управляют формой

Это потребовало некоторых усилий, но теперь приложение «Несоответствия» динамически создает форму «Анкета», используя данные признаков несоответствия, сохраненные в базе. Это означает, что любые изменения, внесенные в базу данных, автоматически отразятся в форме. В этом заключается основная идея управления интерфейсом пользователя веб-приложения с помощью информации, сохраненной в базе данных. Но что произойдет, если у нас появятся плохие данные?



Данные, конечно, хорошо управляют созданием формы, но все же что-то здесь не так. В одной из записей в наименовании категории была допущена ошибка, в результате чего PHP создал новую группу признаков несоответствия для этой «новой» категории. Это серьезная проблема, так как нарушается принцип объединения близких по значению признаков несоответствия в смысловые группы, что мешает лучше организовать форму и облегчить ввод информации.



Фрэнк: Это элементарно. Просто надо исправить ошибки в наименованиях категорий в таблице `mismatch_topic`.

Джо: Но у нас больше одного неправильного наименования категорий. И вообще, чем больше я думаю об этом, тем меньше понимаю, почему мы должны сохранять наименования одних и тех же категорий больше чем один раз.

Джил: Я согласна. Мы решали проблемы с дублированием данных при разработке схемы базы данных, а теперь у нас дублируются наименования категорий. И не просто дублируются — в них ошибки.

Фрэнк: Ладно, что вы скажете, если мы просто откажемся от наименований категорий и будем ссылаться на них по номерам? Тогда у нас не будет опечаток.

Джо: Согласен, только нам все равно будут нужны их наименования для вывода в форме.

Джил: А может нам ссылаться на категории по номерам, но оставить их наименования? Что-то вроде того, что мы уже делаем, когда имеем дело с таблицей `mismatch_topic`, а?

Джо: Точно! Мы не захотели сохранять множество повторяющихся наименований признаков несоответствия в таблице `mismatch_response`, поэтому сохранили их в таблице `mismatch_topic` и связали числовыми ключами записи с данными по признакам несоответствия с наименованиями этих признаков.

Фрэнк: Ты хочешь сказать, что мы могли бы решить проблему дублирования наименований путем создания таблицы с данными по ним?

Джил: Именно это он и говорит. Мы можем создать новую таблицу `mismatch_category`, в которой сохранить каждое наименование категории только один раз. А затем связать записи с данными по категориям с записями, содержащими данные по признакам несоответствия, используя первичный и внешний ключи в таблицах `mismatch_topic` и `mismatch_category`. Великолепно!

Стремитесь к нормализации

В теории баз данных нормализация означает уменьшение дублирования данных и повышение уровня связи между таблицами.

Процесс изменения структуры базы данных приложения «Несоответствия», направленный на устранение дублирования данных, разбиение их на элементарные части и логическое, согласованное связывание таблиц, известен под названием нормализации. Нормализация — это достаточно сложный и глубокий раздел теории баз данных, который может показаться даже немного пугающим. Но бояться не следует. Существует множество простых приемов для разработки структур баз данных, и вы можете позаимствовать их из основных положений нормализации, чтобы сделать вашу базу данных MySQL значительно лучше, чем та, которую бы вы разрабатывали, руководствуясь только соображениями о том, как ваши данные будут выводиться на экран монитора.

Ниже приведены несколько общих этапов, которым вы можете следовать, начиная процесс разработки структуры базы данных, и которые естественным образом приведут вас к более нормализованной базе данных.

1. **Выберите определенное понятие, единственное понятие, которое должна описывать таблица.**

В чем заключается то главное, что эта таблица будет описывать?

2. **Определите состав информации, которую вам необходимо узнать об упомянутом понятии, когда вы обращаетесь к этой таблице.**

Как вы будете использовать эту таблицу?

3. **Разбейте информацию об этом понятии на отдельные структурные единицы данных, которые вы можете использовать при организации структуры таблицы.**

Как добиться наибольшего упрощения запроса к этой таблице?

Одним из фундаментальных принципов нормализации является понятие атомарных данных. Атомарными считаются данные, не обладающие внутренней структурой, то есть такие данные, которые нельзя разбить на части без потери их смысла для базы данных и приложения. Например, данные, сохраняемые в колонках `first_name` и `last_name` в таблице `mismatch_user`, являются атомарными, так как они сохраняют смысл для приложения «Несоответствия» в такой минимизированной форме, что дальнейшее разбиение их на элементарные составляющие невозможно, потому что приведет к потере этого смысла.

Атомарные данные — это данные, разбитые до минимального размера, необходимого приложению, при котором они сохраняют свой смысл для этого приложения.

Может оказаться, что для приложения не всегда есть необходимость разбивать данные до их минимального размера, и если воспользоваться предыдущим примером, то не стоит разбивать полное имя пользователя на имя и фамилию. Если приложение всегда обращается к пользователю по его полному имени, то полное имя может считаться достаточно атомарным для этого случая.

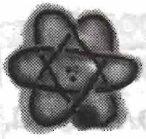
В процессе нормализации думайте об атомизации

Для того чтобы воплотить в жизнь результаты мозгового штурма по проблеме структуры базы данных, полезно поставить несколько вопросов по поводу ваших данных. Это поможет понять, насколько структура таблицы соответствует набору данных, и насколько та степень, в которой данные разбиты на составные части, соответствует принципу атомизации. Никто еще не говорил, что расщепление атома — простая задача, но этот список вопросов может немного помочь.

Обеспечение вашим данным атомарности — это первый шаг на пути нормализации базы данных.

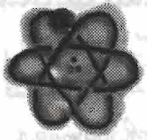
1. В чем заключается то главное, что ваша таблица будет описывать?

Что описывает ваша таблица: наблюдение космических пришельцев, лист рассылки, рейтинги видеоигр, беспомощных романтиков?



2. Как вы будете использовать эту таблицу?

Разрабатывайте структуру вашей таблицы так, чтобы к ней легко можно было делать запросы!



3. Содержат ли колонки вашей таблицы атомарные данные для обеспечения краткости и результативности запросов?

Убедитесь, что данные настолько малы, насколько это возможно.

Не бывает глупых вопросов

В: Должен ли я пытаться разбить свои данные до минимально возможного размера?

О: Необязательно. Обеспечение атомарности данных означает разбиение данных до такой степени, в которой обеспечивается максимальная эффективность, а не просто до минимально возможной.
Не разбивайте данные в большей степени, чем это вам необходимо. Если вам не нужны лишние колонки, не создавайте их просто потому, что у вас есть такая возможность.

В: В чем атомизация данных может мне помочь?

О: Она поможет убедиться в том, что данные в вашей таблице сохранены без ошибок. Например, если в вашей колонке для сохранения адреса имеется адрес человека, наблюдавшего посещение космических пришельцев, вы, возможно, захотите разбить эту колонку на две: одна — для сохранения наименования улицы, а вторая — номера дома. После этого вы можете наладить проверку того, что в колонку с номером дома заносятся только числа. Атомарные данные также позволяют вам делать ваши запросы более эффективными, так как их легче создавать и их выполнение требует меньше серверных ресурсов, причем по мере разрастания вашей базы данных значение этих факторов увеличивается.

Кому, в конце концов, нужна эта нормализация

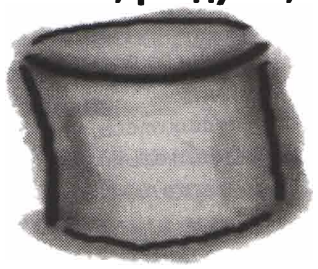
Если все эти слова по поводу атомизации и нормализации вашей скромной базы данных покажутся чем-нибудь вроде стрельбы из пушки по воробьям, подумайте о том времени, когда ваша база данных увеличится в размерах. Что если она стремительно разрастется за очень короткий период времени, усугубив все слабости, присутствующие в ее структуре? Вам бы лучше тогда заняться поиском в интернет-магазинах какого-нибудь сверхмодного автомобиля, чем пытаться с помощью допотопного лейкопластыря собирать ваши данные, которые вносятся туда-сюда в полном беспорядке. Вот тогда вы вспомните о нормализации.

Если вы все-таки не убедились в целесообразности такого подхода или вас в течение всего дня не оставляет мечта об этом «Макларене» желто-канареечного цвета, ниже приведены два довода в пользу нормализации вашей базы данных.

Нормализация имеет свои достоинства, которые заключаются в оптимизации размеров базы данных и скорости обработки информации, содержащейся в ней.

1. В нормализованной таблице отсутствует дублирование данных, что уменьшает размеры вашей базы данных.

Огромная, раздутая, отвратительная база данных...



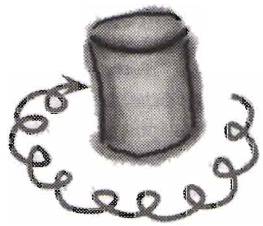
Нормализованная база данных обычно значительно меньше по размеру некачественно разработанной базы.

Как говорится, «размеры — деньги»!

...компактная, эффективная, замечательная база данных!

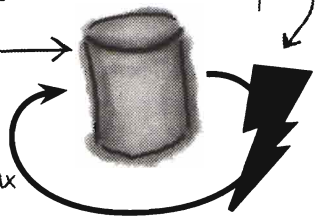
2. С уменьшением количества данных для поиска выполнение запросов проходит быстрее.

Медленные запросы, застрявшие в отвратительной трясине многократно дублированных данных...



Когда речь идет о базах данных, высокая скорость обработки информации — это всегда хорошо.

Стоп, подождите минутку, может, «время — деньги»?

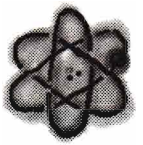


...молниеносно быстрые запросы!

Три этапа нормализации базы данных

Вы подумали над вашими данными какое-то время и сейчас имеете полное представление, почему ваша база данных должна быть нормализована, но пока все это только общие мысли. Ниже приведен краткий перечень правил, которые вам необходимо выполнять для достижения этой цели. Причем этот перечень применим для нормализации любой базы данных. Это что-то вроде инструкции, которую вы должны соблюдать для обеспечения нормализации базы данных. Итак, приступим.

Нормализация базы данных включает строгое выполнение нескольких этапов.



1 Убедитесь, что данные колонок атомарны.

Данные колонок считаются атомарными, если они не обладают внутренней структурой, то есть не содержат нескольких значений одного и того же типа.

Пристрастие	Отвращение
ковбойские ботинки, длинные волосы, реалити-шоу, легкая музыка, опера	татуировка, золотые цепочки, пирсинг, профессиональная борьба, фильмы ужасов
...	
татуировка, золотые цепочки, пирсинг, ковбойские ботинки, длинные волосы, профессиональная борьба, фильмы ужасов	реалити-шоу, легкая музыка, опера
...	

Множество однотипных данных в одной колонке и одновременно несколько колонок с аналогичными данными... Это большая проблема!

2 Создайте для каждой таблицы первичный ключ.

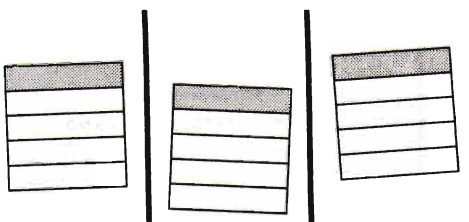
Первичный ключ гарантирует уникальный доступ к записям с данными в таблице. Необходимо, чтобы первичный ключ содержал одну колонку, которая в идеале должна хранить числовые данные, чтобы выполнение запроса происходило с максимальной эффективностью.

username	password	...
даердри	08447b...	...
болдпол	230dcd...	...
инетлз	e511d7...	...
рубир	062e4a...	...
джхейкинг	b4f283...	...

Без первичного ключа отсутствует возможность обеспечить уникальность записей в таблице.

3 Убедитесь, что неключевые колонки не зависят друг от друга.

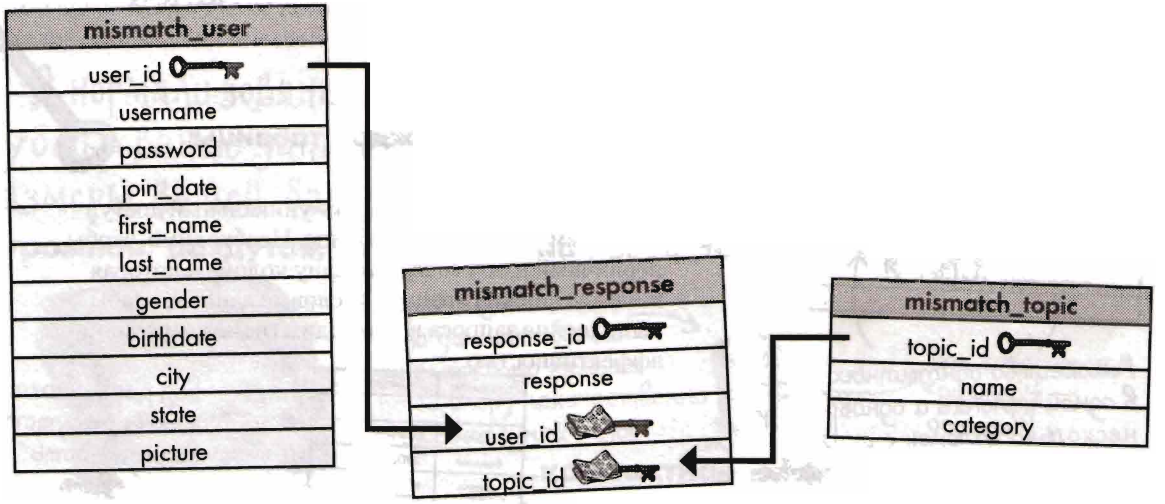
Это наиболее сложное требование в обеспечении нормализации базы данных, и оно не относится к тем требованиям, которые необходимо соблюдать слишком строго. Вам нужно очень внимательно проверять, как данные разных колонок внутри таблицы зависят друг от друга. Основная идея заключается в том, что изменение данных в одной колонке не должно вызывать необходимость изменять данные в другой.



username	password	...	city	state	zip	picture
даердри	08447b...	...	Cambridge	MA	02138	dierdrepic.jpg
болдпол	230dcb...	...	Charleston	SC	29401	paulpic.jpg
инетлз	e511d7...	...	Athens	GA	30601	johanpic.jpg
рубир	062e4a...	...	Conundrum	AZ	85399	rubypic.jpg
джхейкинг	b4f283...	...	Tupelo	MS	38801	elmerpic.jpg

Колонка, содержащая гипотетический ZIP-код, зависит от колонок, содержащих наименования города и штата. Это означает, что изменение значения в одной из них потребует изменения в остальных двух. Чтобы разрешить эту проблему, мы должны перенести данные о месте жительства пользователя в свою собственную таблицу, используя ZIP-код в качестве первичного ключа.

База данных приложения «Несоответствия» требует глубокой реконструкции с целью ее нормализации и решения проблемы дублированных наименований категорий. Используя существующую структуру базы данных, разработайте эскиз новой структуры, в которой проблема дублирования наименований категорий будет решена, что устранит риск ошибок в них. Не забудьте прокомментировать схему, чтобы объяснить, как все это действует.



Не бывает глупых вопросов

В: Что мне нужно сделать для выполнения третьего этапа по нормализации базы данных приложения «Несоответствия» и решить проблему «город/штат/ZIP-код»?

О: Решение заключается в переносе данных о месте жительства пользователя в свою собственную таблицу и связывании таблицы `mismatch_user` с новой таблицей через внешний ключ. Вы можете создать таблицу с именем `mismatch_location` с первичным ключом `location_id` и колонками для сохранения города и штата, в которых проживает пользователь. После этого нужно будет удалить колонки `city` и `state` из таблицы `mismatch_user` и вместо них добавить колонку `location_id`, которая должна быть внешним ключом. Тогда проблема будет решена! Что делает эту схему работающей — так это то, что колонка `location_id` использует ZIP-код в качестве первичного ключа, решая проблему неключевой зависимости данных о месте жительства пользователя.

В: Ух ты, похоже, нужно затратить немалые усилия, чтобы выполнить необязательное требование по нормализации базы данных. Это действительно так необходимо?

О: И да, и нет. Первые два этапа по нормализации базы данных являются безусловными потому, что атомизированные данные и наличие первичных ключей считаются критическими условиями любой хорошей структуры базы данных. Что касается третьего этапа, здесь вас, с одной стороны, соблазняет желание создать совершенную структуру базы данных, с другой — сдерживают практические соображения, связанные с тем, что в действительности является более важным для вашего приложения. В случае приложения «Несоответствия» проблему «город/штат/ZIP-код» для упрощения, возможно, стоило бы оставить нерешенной. Это решение не из легких, и многие пуристы от баз данных оспорили бы его, настаивая на строгом соблюдении всех трех этапов нормализации. Хорошая новость заключается в том, что ZIP-код является величиной гипотетической и такой колонки в действительности нет в структуре таблицы `mismatch_user`, поэтому нам не о чем особенно беспокоиться.

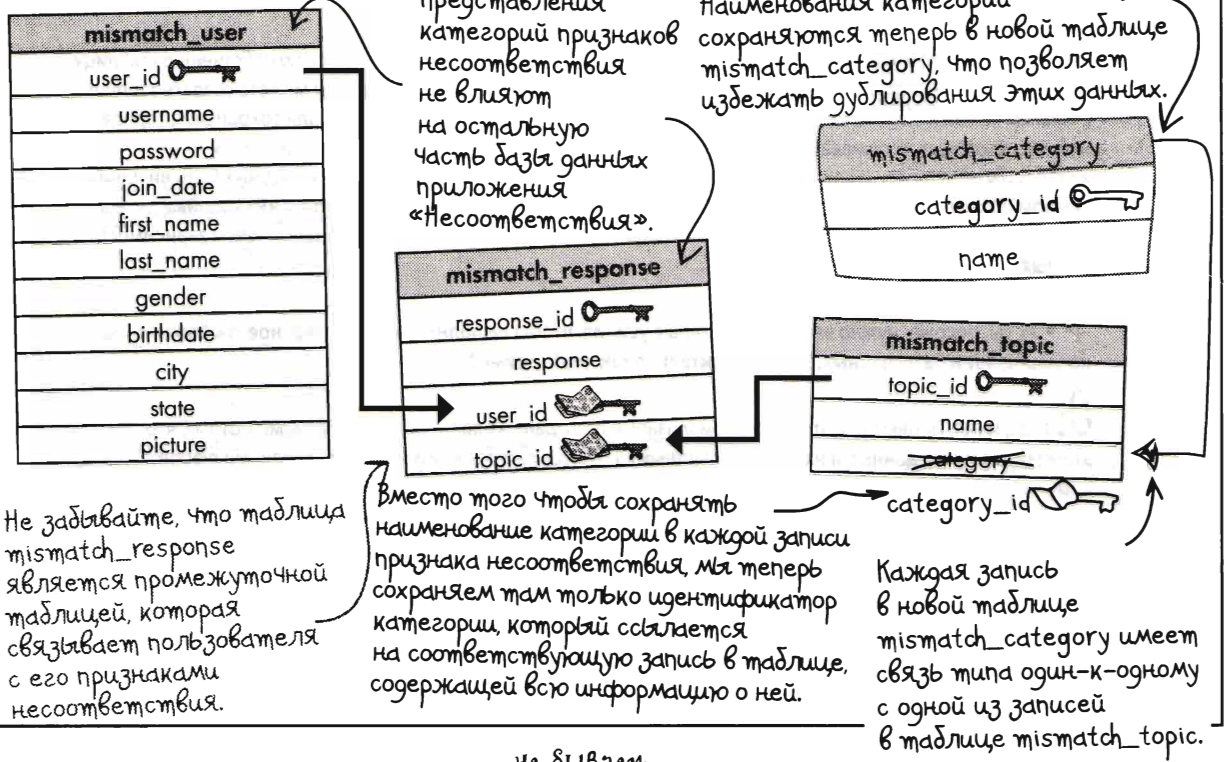
В: Даже в отсутствие колонки, содержащей ZIP-код, не следует ли перенести наименования городов и штатов в свою собственную таблицу, чтобы выполнить требования третьего этапа нормализации?

О: Возможно. Совершенно очевидно, что вы столкнетесь с проблемой дублирования наименований городов и штатов в таблице `mismatch_user`. Проблема, связанная с их переносом в свою собственную таблицу в отсутствии ZIP-кодов, заключается в том, что вам придется внести в эту таблицу наименования всех существующих городов и штатов. В противном случае пользователь, без сомнения, рано или поздно допустит ошибку, и вы опять столкнетесь с той же самой проблемой. Это хороший пример того, что для каждого конкретного приложения вам необходимо тщательно сопоставлять все достоинства строгой нормализации базы данных с условиями реальной жизни.

Один из возможных и весьма интересных способов решения этой проблемы заключается в использовании ZIP-кода в таблице `mismatch_user` вместо наименования города и штата и извлечении этих наименований при необходимости из отдельной, специальной таблицы или поиске и использовании подходящего веб-сервиса. Это решение достаточно сложное по сравнению с тем, что нам сейчас необходимо, поэтому мы просто оставим наименования городов и штатов в таблице `mismatch_user`.

РЕШЕНИЕ К УПРАЖНЕНИЮ

База данных приложения «Несоответствия» требует глубокой реконструкции с целью ее нормализации и решения проблемы дублированных наименований категорий. Используя существующую структуру базы данных, разработайте эскиз новой структуры, в которой проблема дублирования наименований категорий будет решена, что устранит риск ошибок в них. Не забудьте прокомментировать схему, чтобы объяснить, как все это действует.



Не бывает глупых вопросов

В: Как новая таблица mismatch_category разрешает проблему дублирования данных?

О: Новая таблица сохраняет наименования категорий, вынося их за пределы таблицы `mismatch_topic`. Так как категории сохранены в своей собственной таблице, нет никакой необходимости дублировать их, сохраняя в этой таблице. Записи в ней теперь просто ссылаются на соответствующие записи в таблице `mismatch_category`, содержащие всю необходимую информацию о категории. Это означает, что каждая запись в таблице `mismatch_category` имеет связь типа один-к-одному с одной из записей в таблице `mismatch_topic`.

В: Значит ли это, что в таблице mismatch_category всего пять записей?

О: Так и есть.

mismatch_category

category_id	name
1	Внешность
2	Времяпрепровождение
3	Еда
4	Люди
5	Организация досуга

Наименование каждой категории сохранено только один раз!

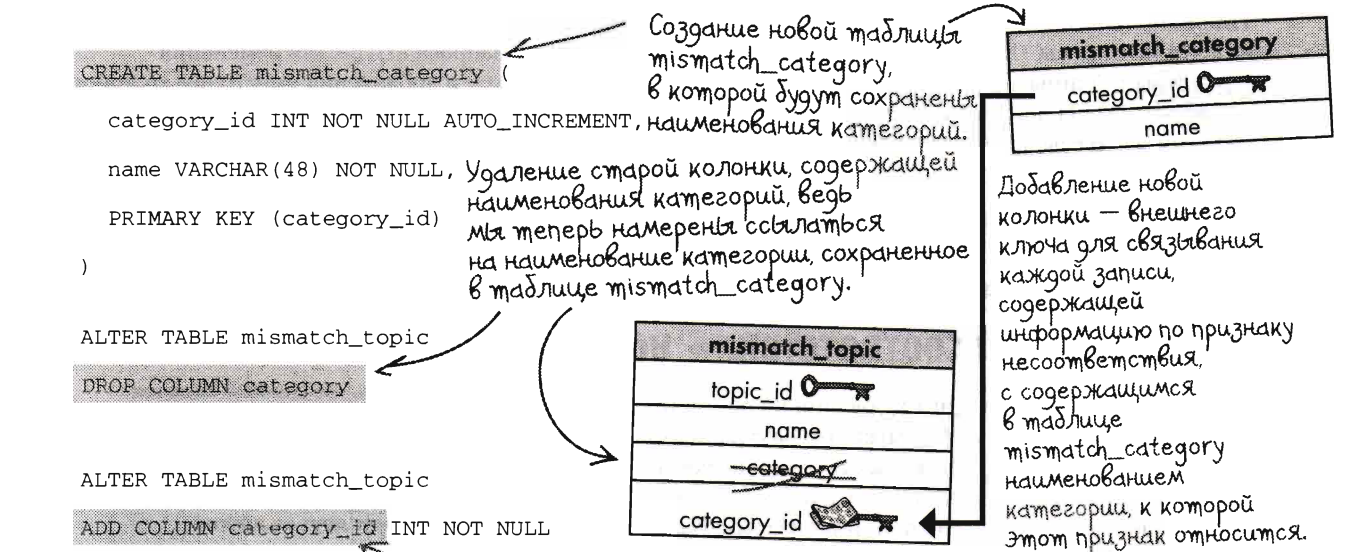
Изменение структуры базы данных приложения «Несоответствия»

Для того чтобы воспользоваться теми преимуществами, которые предоставляет нам новая схема, необходимо внести некоторые изменения в структуру базы данных. Конкретней, мы должны создать новую таблицу `mismatch_category` и связать ее с таблицей `mismatch_topic` по внешнему ключу. А так как колонка `category` в таблице `mismatch_topic` больше использоваться не будет, ее необходимо удалить.

```
CREATE TABLE mismatch_category (
  category_id INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(48) NOT NULL,
  PRIMARY KEY (category_id)
)

ALTER TABLE mismatch_topic
DROP COLUMN category

ALTER TABLE mismatch_topic
ADD COLUMN category_id INT NOT NULL
```

Создание новой таблицы mismatch_category, в которой будут сохранены наименования категорий.

Удаление старой колонки, содержащей наименования категорий, ведь мы теперь намерены ссылаться на наименование категории, сохраненное в таблице mismatch_category.



В новую таблицу `mismatch_category` необходимо занести наименования категорий, что может быть сделано с помощью хорошо нам знакомого SQL-запроса `INSERT`.

```
INSERT INTO mismatch_category (name) VALUES ('Внешность')
INSERT INTO mismatch_category (name) VALUES
('Времяпрепровождение')
INSERT INTO mismatch_category (name) VALUES ('Еда')
INSERT INTO mismatch_category (name) VALUES ('Люди')
INSERT INTO mismatch_category (name) VALUES
('Организация досуга')
```

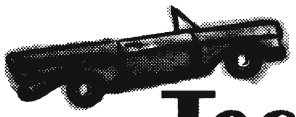
mismatch_topic

topic_id	name	category_id
...		
8	Фильмы ужасов	2
9	Легкая музыка	2
10	Опера	2
11	Суши	3
12	Спэм	3
13	Острые блюда	3
14	Сэндвичи с ореховым маслом и бананом	3
15	Мартини	3
16	Говард Стерн	4
17	Билл Гейтс	4
18	Барбара Стрейзанд	4
...		

В новую колонку `category_id` таблицы `mismatch_category` необходимо также занести идентификаторы категорий так, чтобы каждая запись, содержащая данные по признаку несоответствия, связывалась с категорией, к которой этот признак относится.

```
UPDATE mismatch_topic SET category_id = 3
  WHERE name = 'Мартини'
```

Значение этого идентификатора категории в таблице mismatch_topic (внешний ключ) должно соответствовать значению автоинкрементного идентификатора категории записи с таким наименованием категории в таблице mismatch_category (первичный ключ).



Тест-драйв

Создайте новую таблицу mismatch_category и занесите в нее данные.

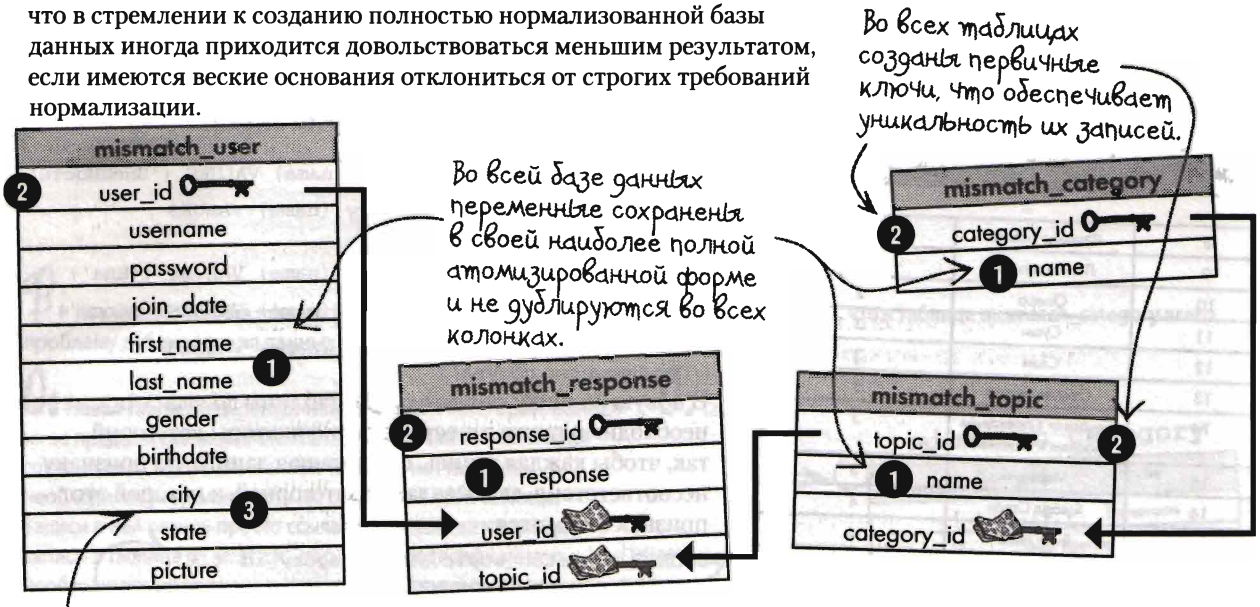
Используя любую инструментальную программу MySQL, добавьте к базе данных приложения «Несоответствия» новую таблицу `mismatch_category`, используя для этого SQL-запрос `CREATE TABLE`, приведенный на предыдущей странице. Затем занесите во вновь созданную таблицу данные, выполнив запросы `INSERT`, приведенные там же. Выполните два запроса `ALTER` для изменения структуры таблицы `mismatch_topic`: удаления колонки `category` и добавления вместо нее колонки `category_id`. И, наконец, измените каждую запись в таблице `mismatch_topic` так, чтобы ее колонка `category_id` ссылалась на запись в таблице `mismatch_category`, содержащую то наименование категории признака несоответствия, к которому этот признак относится.

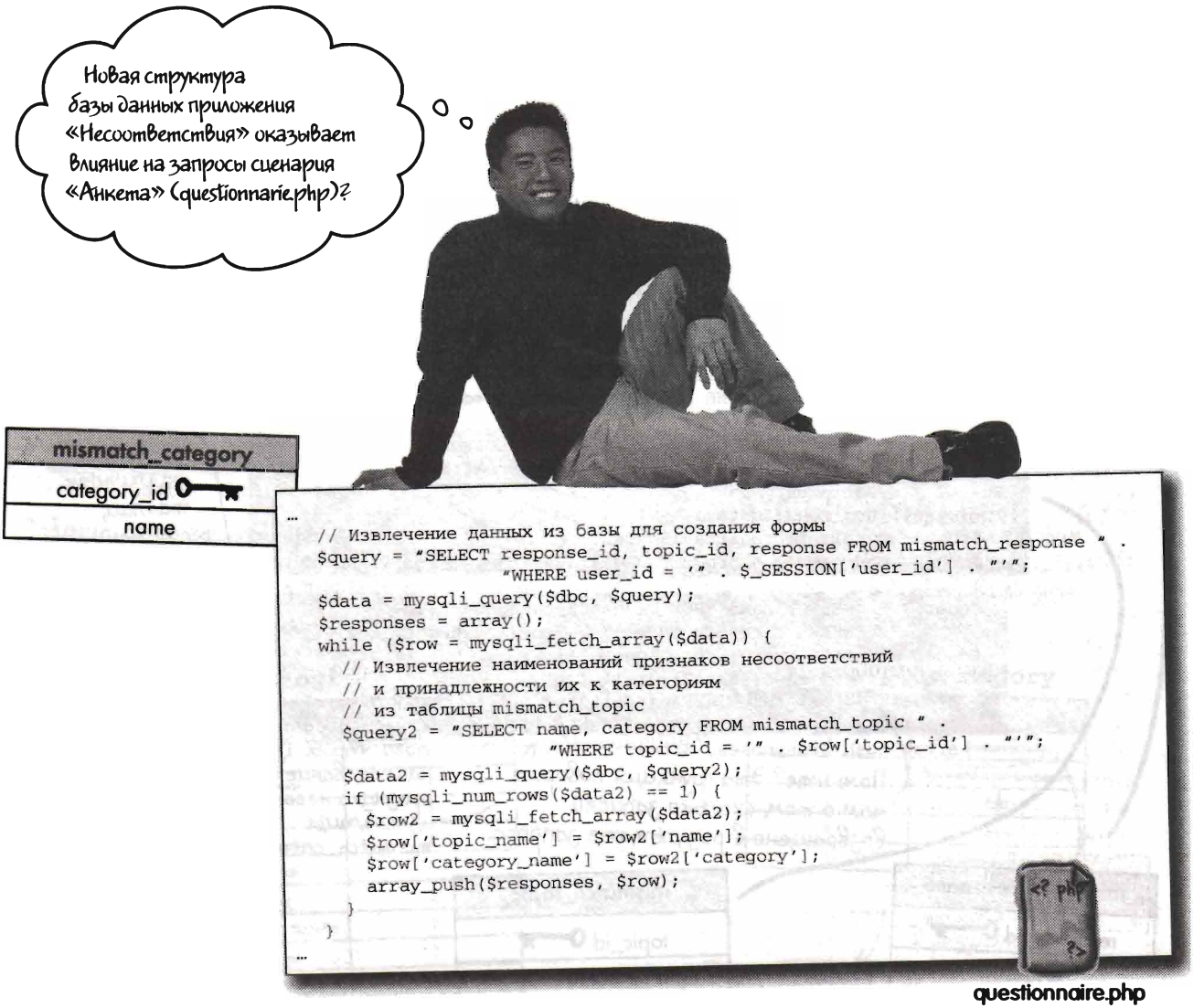
А теперь выполните запрос `SELECT` для каждой таблицы, чтобы убедиться, что все в порядке.

Итак, действительно ли база данных приложения «Несоответствия» нормализована?

Да. Если вы выполните все три требования по нормализации для каждой из ваших таблиц, вы убедитесь, что база данных качественно изменилась в лучшую сторону. Но даже если и не совсем так, не все потеряно. Как и по отношению к людям, по отношению к базам данных можно говорить о степени нормализации. Важно учитывать, что в стремлении к созданию полностью нормализованной базы данных иногда приходится довольствоваться меньшим результатом, если имеются веские основания отклониться от строгих требований нормализации.

1. **Убедитесь, что данные колонок атомарны.**
2. **Создайте для каждой таблицы первичный ключ.**
3. **Убедитесь, что неключевые колонки не зависят друг от друга.**





questionnaire.php

Да. В большинстве случаев изменение структуры базы данных вынуждает нас корректировать любой запрос к ней.

В этом случае изменение структуры базы данных, связанное с добавлением новой таблицы `mismatch_category`, влияет на любой запрос, в котором имеется ссылка на таблицу `mismatch_topic`. Это происходит потому, что прежняя структура базы данных предусматривала сохранение категорий признаков несоответствия непосредственно в таблице `mismatch_topic`. После того как для сохранения категорий признаков несоответствия была создана своя собственная таблица, что само по себе является удачным решением, значительно повышающим нормализацию базы данных, появляется необходимость в пересмотре запросов и кода, ответственных за работу с новой таблицей (`mismatch_category`).

Запрос в запросе в запросе...

Одна из проблем, связанных с нормализацией базы данных, заключается в том, что запросы часто требуют включения подзапросов, так как вам необходимо обращаться за данными к множеству таблиц. Это может приводить к усложнению и запутыванию кода. Рассмотрим новую версию запроса, в результате выполнения которого создается массив признаков несоответствия для построения формы анкеты в приложении «Несоответствия»:

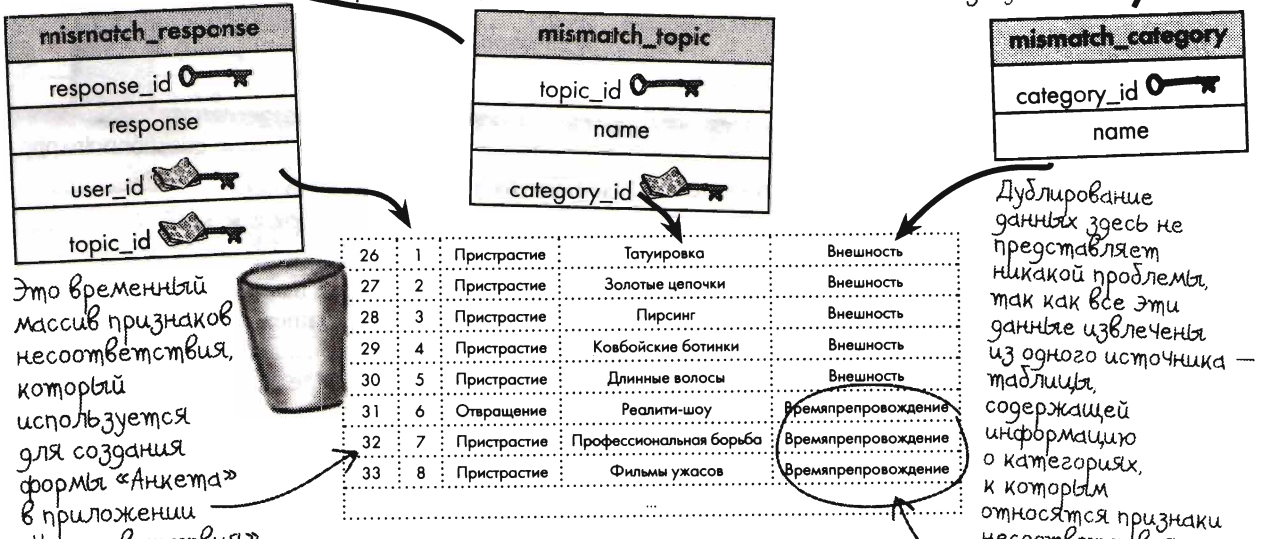
Увеличение количества таблиц в базе данных обычно приводит к усложнению и запутыванию запросов к ней.

```
// Извлечение данных из базы для создания формы
$query = "SELECT response_id, topic_id, response FROM mismatch_response " .
         "WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
  // Извлечение наименований признаков несоответствий
  // из таблицы mismatch_topic
  $query2 = "SELECT name, category_id FROM mismatch_topic " .
            "WHERE topic_id = '" . $row['topic_id'] . "'";
  $data2 = mysqli_query($dbc, $query2);
  if (mysqli_num_rows($data2) == 1) {
    $row2 = mysqli_fetch_array($data2);
    $row['topic_name'] = $row2['name'];

    $query3 = "SELECT name FROM mismatch_category " .
              "WHERE category_id = '" . $row2['category_id'] . "'";
    $data3 = mysqli_query($dbc, $query3);
    if (mysqli_num_rows($data3) == 1) {
      $row3 = mysqli_fetch_array($data3);
      $row['category_name'] = $row3['name'];
      array_push($responses, $row);
    }
  }
}
```

Мы извлекаем данные из трех различных таблиц, с помощью трех различных запросов.

Этот новый запрос использует значение ключа category_id для того, чтобы извлечь наименование категории признака несоответствия из таблицы mismatch_category.

Помните? Эта функция говорит нам о том, сколько записей возвращено в результате запроса.



Это временный массив признаков несоответствия, который используется для создания формы «Анкета» в приложении «Несоответствия».

Дублирование данных здесь не представляет никакой проблемы, так как все эти данные извлечены из одного источника — таблицы, содержащей информацию о категориях, к которым относятся признаки несоответствия. А в этой таблице они не дублируются.

Соединим наши ~~руки~~ таблицы

Ой! Можно ли что-нибудь сделать со всеми этими запросами в запросах? Решение заключается в SQL-операции, известной под названием **объединение** (`join`) и позволяющей извлекать результаты из нескольких таблиц с помощью одного запроса. Имеется множество видов объединений, но наиболее популярным является **внутреннее объединение** (`inner join`), при использовании которого записи извлекаются из двух таблиц на основании какого-либо условного выражения. В результате выполнения запроса с использованием внутреннего объединения извлекаются только те записи, которые отвечают этому условному выражению.

В результате выполнения единственного запроса с использованием внутреннего объединения извлекаются записи из нескольких таблиц одновременно.

Из двух различных таблиц будут извлечены идентификатор признака несоответствия и наименование категории, к которой этот признак относится.

```
SELECT mismatch_topic.topic_id, mismatch_category.name
  FROM mismatch_topic
 INNER JOIN mismatch_category
  ON (mismatch_topic.category_id = mismatch_category.category_id)
```

Таблица, содержащая информацию о категориях признаков несоответствия, была присоединена в запросе к таблице, содержащей информацию о признаках несоответствия, с использованием ключевых слов INNER JOIN.

Условие объединения таблиц заключается в том, что идентификаторы категорий признаков несоответствия должны соответствовать друг другу во всех записях обеих таблиц.

mismatch_topic

topic_id	name	category_id
1	Татуировка	1
2	Золотые цепочки	1
3	Пирсинг	1
4	Ковбойские ботинки	1
5	Длинные волосы	1
6	Реалити-шоу	2
7	Профессиональная борьба	2
8	Фильмы ужасов	2
9	Легкая музыка	2
10	Опера	2
11	Суши	3
...		

Эти колонки управляют объединением!

mismatch_category

category_id	name
1	Внешность
2	Времяпрепровождение
3	Еда
4	Люди
5	Организация досуга

1	Внешность
2	Внешность
3	Внешность
4	Внешность
5	Внешность
6	Времяпрепровождение
7	Времяпрепровождение
8	Времяпрепровождение
...	

Вторая колонка в результатах запроса содержит наименование категории, к которой относится признак несоответствия, извлеченное из таблицы mismatch_category так, чтобы его идентификатор соответствовал идентификатору категории в таблице mismatch_topic.

Первая колонка в результатах запроса содержит идентификатор признака несоответствия, извлеченный из таблицы, содержащей данные о признаках несоответствия.

В результатах запроса содержатся данные из обеих таблиц.

В результате выполнения запроса с использованием внутреннего объединения данные из двух таблиц успешно объединяются в одном результате.

Без использования объединения пришлось бы делать два отдельных запроса.

Соедините ~~точки~~ с помощью точек

Так как в объединении участвует более одной таблицы, очень важно давать ясно понять, на колонку какой таблицы дается ссылка в запросе. Говоря точнее, чтобы избежать путаницы, вы должны указывать наименование таблицы для каждой колонки, так как у таблиц часто встречаются одинаковые имена колонок, особенно если речь идет о ключах. Эта проблема решается простым добавлением имени таблицы перед именем колонки, при этом наименование таблицы отделяется от наименования колонки точкой. Например, рассмотрим ранее приведенный запрос, в результате выполнения которого создается набор идентификаторов признаков несоответствия и наименований категорий, к которым они относятся.

Использование условного обозначения «точка» позволяет определять, какой таблице принадлежит колонка, участвующая в объединении.

Это наименование таблицы.

Это точка!

Это наименование колонки в таблице, отделенное от наименования таблицы, которой она принадлежит, с помощью условного обозначения «точка». И точка!

```
SELECT mismatch_topic.topic_id, mismatch_category.name
  FROM mismatch_topic
  INNER JOIN mismatch_category
  ON (mismatch_topic.category_id = mismatch_category.category_id)
```

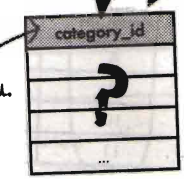
Еще одна ссылка «таблица/колонка», в которой используется условное обозначение «точка».

Это то место, где использование условного обозначения «точка» полностью устраняет всякую двусмысленность, являющуюся результатом того, что в двух разных таблицах имеются колонки с одинаковыми именами.

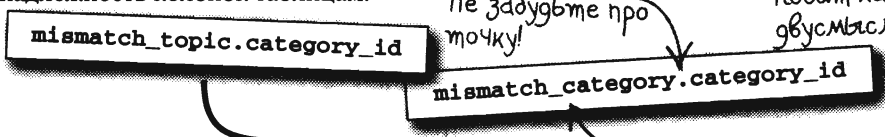
Не имея возможности указать наименование таблицы, которой принадлежит колонка, мы столкнулись бы в этом запросе с проблемой двусмысленности. Фактически выражение, следующее после ключевого слова `ON`, правильно понять было бы невозможно, потому что оно означало бы проверку на равенство значения колонки `category_id` самому себе, предположительно — в таблице `mismatch_topic`. Исходя из этих соображений, при построении запросов с использованием объединений необходимо в явном виде указывать принадлежность колонок таблицам.

Наименование колонки само по себе ничего не говорит о том, какой таблице она принадлежит.

Внутри выражения JOIN использование наименования такой колонки носит характер двусмысленности.



Не забудьте про точку!



mismatch_topic

topic_id	name	category_id
1	Татуировка	1
2	Золотые цепочки	1
3	Пирсинг	1
...	...	

Включение имени таблицы устраняет двусмысленность в запросе с использованием объединений.

mismatch_category

category_id	name
1	Внешность
2	Времяпрепровождение
3	Еда
...	...

В действительности мы можем делать еще больше, используя внутренние объединения

Внутреннее объединение не ограничивает использование в запросах других SQL-конструкций. Так как запрос с внутренними объединениями не перестает быть запросом, вы можете использовать в нем и другие допустимые в SQL конструкции для дальнейшего уточнения результата. Например, если вы хотите извлечь записи только с определенной информацией, вы можете включить обычное условное выражение WHERE, чтобы ограничить состав извлекаемых записей только теми из них, которые отвечают требованиям этого условного выражения.

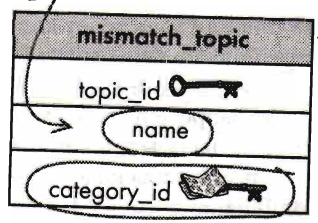
Внутреннее объединение комбинирует записи двух таблиц, используя в условных выражениях операторы сравнения.

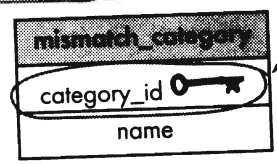
```
SELECT mismatch_topic.topic_id, mismatch_category.name
  FROM mismatch_topic
  INNER JOIN mismatch_category
  ON (mismatch_topic.category_id = mismatch_category.category_id)
  WHERE mismatch_topic.name = 'Фильмы ужасов'
```

Так что же будет в результатах выполнения этого запроса? Прежде всего не забывайте, что условное выражение WHERE служит для дальнейшего ограничения той части запроса, которая предшествует ему. Иначе говоря, это выражение вводит дальнейшие ограничения на перечень записей, извлекаемых в результате выполнения предыдущей части запроса. Повторим результат выполнения запроса без условного выражения WHERE:

Эта колонка ограничивает количество записей в результатах запроса как часть условного выражения WHERE.

Эти две колонки управляют объединением двух таблиц.





Идентификатор признака несоответствия извлечен из таблицы, содержащей данные о признаках несоответствия.

1	Внешность
2	Внешность
3	Внешность
4	Внешность
5	Внешность
6	Времяпрепровождение
7	Времяпрепровождение
8	Времяпрепровождение
...	

Наименование категории, к которой относится признак несоответствия, извлечено из таблицы mismatch_category.

Условное выражение WHERE оказывает эффект «вырезания» из всего перечня записей единственной записи с признаком несоответствия «Фильмы ужасов». Нам нужно посмотреть на таблицу mismatch_topic, чтобы увидеть, что это за запись.

mismatch_topic

topic_id	name	category_id
...		
7	Профессиональная борьба	2
8	Фильмы ужасов	2
9	Легкая музыка	2
...		

Эта запись отвечает требованиям условного выражения WHERE.

Результат выполнения первой части запроса (предшествующей условному выражению WHERE) урезан до этой единственной записи.

8	Времяпрепровождение

Условное выражение WHERE ограничило количество записей в результате запроса до одной.

не бывает глупых вопросов

В: Значит, условное выражение WHERE дает вам возможность ограничить результат объединенного запроса на основании записей одной из объединяемых таблиц?

О: Это так. Имейте в виду, что проверка выполнения требований условного выражения производится для оригинальной таблицы, а не для результатов объединенного запроса. Поэтому в предыдущем примере в результате выполнения запроса сначала извлекаются записи из двух таблиц с одинаковыми значениями идентификатора категории признака несоответствия (`category_id`), а затем выбираются только те записи, для которых колонка, содержащая наименование признака несоответствия, имеет определенное значение («Фильмы ужасов»). Поэтому при объединении таблиц рассматриваются значения колонок в двух таблицах, а при ограничении результата в соответствии с требованием условного выражения `WHERE` рассматриваются значения колонки, содержащей наименование категории признака несоответствия в одной таблице — `mismatch_topic`.

В: Может ли условное выражение WHERE в объединенном запросе основываться на данных таблицы mismatch_category?

О: Безусловно. Условное выражение `WHERE` может ограничивать результат, основываясь на данных любой из таблиц объединения. В качестве примера можно составить запрос с условным выражением `WHERE`, ограничивающим результат записями с определенным значением категории признаков несоответствия:

```
... WHERE mismatch_category.name = 'Времяпрепровождение'
```

Условное выражение `WHERE` ограничивает результат запроса только теми записями, для которых наименование категории признака несоответствия имеет значение «Времяпрепровождение». Условное выражение `WHERE` не влияет на то, как будет происходить объединение таблиц, а влияет только на состав записей, включаемых в результат запроса.

Упрощение выражения ON с помощью ключевого слова USING

Помните, цель использования выражения `INNER JOIN` состояла в том, чтобы упростить запутанные запросы приложения «Несоответствия»? Когда в объединении участвуют колонки с одинаковыми именами, мы можем еще больше упростить запрос с помощью выражения с ключевым словом `USING`. Выражение `USING` используется вместо выражения `ON` в объединенном запросе и требует указания имени колонки, используемой в качестве параметра объединения. Для применения этого выражения необходимо только, чтобы колонки в обеих объединяемых таблицах имели абсолютно одинаковые имена. В качестве примера рассмотрим тот же запрос:

Имена колонок в обеих таблицах одинаковые, только имена самих таблиц, разные.

```
SELECT mismatch_topic.topic_id, mismatch_category.name
  FROM mismatch_topic
  INNER JOIN mismatch_category
  ON (mismatch_topic.category_id = mismatch_category.category_id)
  WHERE mismatch_topic.name = 'Фильмы ужасов'
```

Так как в выражении `ON` используются одинаковые имена колонок (`category_id`), оно может быть заменено на выражение `USING`:

```
SELECT mismatch_topic.topic_id, mismatch_category.name
  FROM mismatch_topic
  INNER JOIN mismatch_category
  USING (category_id)
  WHERE mismatch_topic.name = 'Фильмы ужасов'
```

Все, что необходимо, — это наименование колонки, не нужен даже знак равенства (=).

Замените выражение с ключевым словом ON на выражение с ключевым словом USING для сокращения запроса, в котором объединяются таблицы на базе одинаковых имен колонок.

Для того чтобы использовать в объединенном запросе выражение с ключевым словом USING, имена колонок, на базе которых объединяются таблицы, должны быть одинаковыми.

Альтернативные имена для таблиц и колонок

Наш объединенный запрос становится все компактнее и компактнее. Давайте продвинемся в этом же направлении еще на один шаг. В SQL обращение к таблицам и колонкам осуществляется по тем наименованиям, которые для них определены в структуре базы данных. Но это может оказаться обременительным в больших запросах, в которых используются объединения нескольких таблиц. Такие громоздкие запросы очень неудобны для чтения и понимания. В этих случаях имеет смысл использовать альтернативные имена, которые являются временными именами, применяемыми для ссылок в запросе на таблицы и колонки.

Альтернативное имя дает вам возможность переименовать таблицу или колонку в запросе, чтобы в определенном смысле преобразовать его содержание, не меняя при этом его назначения (сделать короче, информативнее и т. п.).

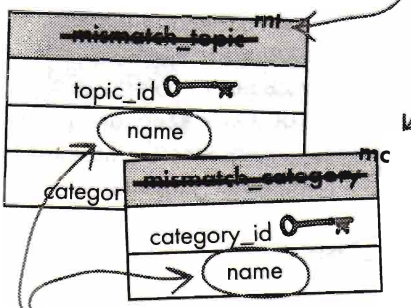
```
SELECT mt.topic_id, mc.name
  FROM mismatch_topic AS mt
  INNER JOIN mismatch_category AS mc
  USING (category_id)
  WHERE mt.name = 'Фильмы ужасов'
```

После замены наименований таблиц на альтернативные имена код становится менее громоздким и более удобным для чтения и понимания.

Ключевое слово AS в данном случае создает альтернативное имя mt для таблицы mismatch_topic.

Ссылка на таблицу mismatch_category осуществляется через ее альтернативное имя mc.

Любая ссылка на таблицу mismatch_topic может быть теперь сокращена до mt (альтернативного имени этой таблицы).



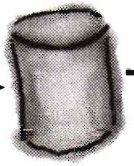
Выбор этих двух колонок с одинаковыми именами для включения в результат запроса... не слишком удачное решение!

Альтернативные имена удобны только для того, чтобы делать запросы более компактными? Нет, существуют ситуации, в которых они просто необходимы! В результате выполнения объединенного запроса, который оказался таким удобным в приложении «Несоответствия», извлекается как наименование признака несоответствия, так и наименование категории, к которой он относится. Но в таблицах `mismatch_topic` и `mismatch_category` используется одно и то же наименование колонок (`name`), в которых содержатся значения этих наименований. Это создает проблему, так как результат запроса становится неопределенным по отношению к наименованию колонок. Но мы можем устранить эту неопределенность, изменив наименования колонок на альтернативные, сделав при этом их более информативными (`topic_name` и `category_name` вместо простого `name`).

```
SELECT mt.name AS topic_name, mc.name AS category_name
  FROM mismatch_topic AS mt
  INNER JOIN mismatch_category AS mc
  USING (category_id)
  WHERE mt.topic_id = '11'
```

Наименование этой колонки заменено теперь на более информативное.

Наименования колонок в результате запроса теперь уникальны и информативны.



topic_name	category_name
Суши	Еда

После замены наименований колонок на альтернативные имена в результате запроса в качестве названий колонок будут выступать эти альтернативные имена, а не те их названия, которые указаны в структурах таблиц.

Объединения повышают эффективность

Объединенные запросы более эффективные и более компактные, чем вложенные запросы.

Итак, объединения позволяют делать запрос одновременно к нескольким таблицам, эффективно извлекать данные более чем из одного места и собирать их в едином результате запроса. Код приложения «Несоответствия», который применяется для создания массива признаков несоответствия, — отличный кандидат на использование объединенных запросов, так как он содержит три вложенных запроса к различным таблицам. Начнем с прежнего варианта кода:

```
// Извлечение данных из базы для создания формы
  $query = "SELECT response_id, topic_id, response FROM mismatch_response " .
                   "WHERE user_id = '" . $_SESSION['user_id'] . "'";
  $data = mysqli_query($dbc, $query);
  $responses = array();
  while ($row = mysqli_fetch_array($data)) {
    // Извлечение наименований признаков несоответствий
    // из таблицы mismatch_topic
    $query2 = "SELECT name, category_id FROM mismatch_topic " .
                     "WHERE topic_id = '" . $row['topic_id'] . "'";
    $data2 = mysqli_query($dbc, $query2);
    if (mysqli_num_rows($data2) == 1) {
      $row2 = mysqli_fetch_array($data2);
      $row['topic_name'] = $row2['name'];

      $query3 = "SELECT name FROM mismatch_category " .
                       "WHERE category_id = '" . $row2['category_id'] . "'";
      $data3 = mysqli_query($dbc, $query3);
      if (mysqli_num_rows($data3) == 1) {
        $row3 = mysqli_fetch_array($data3);
        $row['category_name'] = $row3['name'];
        array_push($responses, $row);
      }
    }
  }
```

В результате выполнения двух последних запросов извлекаются наименования признаков несоответствия и категорий, к которым они относятся: каждое наименование — отдельным запросом к соответствующей таблице.

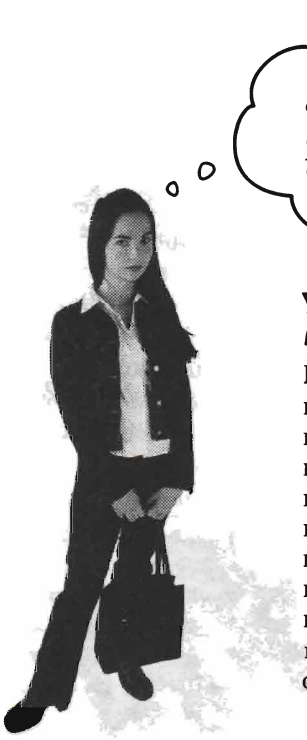
А это новая версия кода с использованием объединений:

```
// Извлечение данных из базы для создания формы
  $query = "SELECT response_id, topic_id, response FROM mismatch_response " .
                   "WHERE user_id = '" . $_SESSION['user_id'] . "'";
  $data = mysqli_query($dbc, $query);
  $responses = array();
  while ($row = mysqli_fetch_array($data)) {
    // Извлечение наименований признаков несоответствий
    // из таблицы mismatch_topic
    $query2 = "SELECT mt.name AS topic_name, mc.name AS category_name " .
      "FROM mismatch_topic AS mt " .
      "INNER JOIN mismatch_category AS mc USING (category_id) " .
      "WHERE mt.topic_id = " . $row['topic_id'] . "";
    $data2 = mysqli_query($dbc, $query2);
    if (mysqli_num_rows($data2) == 1) {
      $row2 = mysqli_fetch_array($data2);
      $row['topic_name'] = $row2['topic_name'];
      $row['category_name'] = $row3['category_name'];
      array_push($responses, $row);
    }
  }
```

С использованием объединения появляется возможность извлечь наименования признаков несоответствия и категорий, к которым они относятся, с помощью одного объединенного запроса.

Для упрощения запроса используются альтернативные имена.

Идентификаторы признаков несоответствия служат в качестве основы выбора записей для основного запроса, в то время как идентификаторы категорий, к которым относятся признаки несоответствия, управляют объединением.



У нас нет необходимости в двух запросах, по крайней мере, если мы используем весь потенциал объединений.

Можно использовать в объединении более двух таблиц, что как раз и требуется в коде приложения «Несоответствия», создающем массив признаков несоответствия. Нам необходим один запрос, в результате выполнения которого будут решены три задачи: извлечены все признаки несоответствия пользователя, получены наименования каждого признака несоответствия и, наконец, извлечены наименования категорий, к которым каждый из этих признаков несоответствия относится. В новом, улучшенном коде, приведенном на предыдущей странице, решены последние две из перечисленных задач: составлен запрос, включающий объединение таблиц `mismatch_topic` и `mismatch_category`. В идеале один запрос с двумя объединениями позволил бы убить всех трех зайцев.

Ниже приведен код, в результате выполнения которого с помощью одного запроса будут извлечены все необходимые данные признаков несоответствия благодаря грамотному использованию объединений. Проявите свои способности и напишите SQL-запрос, использующий объединение таблиц `mismatch_response`, `mismatch_topic` и `mismatch_category`.

```
// Извлечение данных из базы для создания формы
$query = ..........................................................................

 ..........................................................................
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
  array_push($responses, $row);
}
```

Ниже приведен код, в результате выполнения которого с помощью одного запроса будут извлечены все необходимые данные признаков несоответствия благодаря грамотному использованию объединений. Проявите свои способности и напишите SQL-запрос, использующий объединение таблиц `mismatch_response`, `mismatch_topic` и `mismatch_category`.

```
// Извлечение данных из базы для создания формы
$query = "SELECT mr.response_id, mr.topic_id, mr.response, " .
  "mt.name AS topic_name, mc.name AS category_name " .
  "FROM mismatch_response AS mr " .
  "INNER JOIN mismatch_topic AS mt USING (topic_id) " .
  "INNER JOIN mismatch_category AS mc USING (category_id) " .
  "WHERE mr.user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
  array_push($responses, $row);
}
```

Использование альтернативных имен позволяет упростить запрос и сделать его более легким для чтения.

Первое объединение добавляет таблицу с данными признаков несоответствия, позволяя извлекать из нее наименования признаков несоответствия на основании значений их идентификаторов

Второе объединение добавляет таблицу с данными категорий, к которым относятся признаки несоответствия, позволяя извлекать из нее наименования категорий на основании значений их идентификаторов.

Сохранение всех данных о признаках несоответствия в массив `$responses`.

не бывает глупых вопросов

В: Какие еще виды объединений существуют?

О: Из имеющихся типов внутренних объединений можно выделить объединения по эквивалентности (`equijoin`), объединения по неэквивалентности (`non-equijoin`) и естественное объединение (`natural join`). Первые два вида выполняют объединения, основываясь на равенстве или неравенстве значений соответственно. Вы уже видели примеры объединения, основанного на равенстве идентификаторов признаков несоответствия и идентификаторов категорий, к которым относятся признаки несоответствия. Так как выбираются только те записи, для которых эти значения равны (эквивалентны), этот класс объединения должен быть отнесен к классу объединений по эквивалентности.

Известно также естественное объединение, которое включает сравнение всех колонок двух таблиц с одинаковыми наименованиями. Фактически естественное объединение можно рассматривать как объединение по эквивалентности, но колонки, значения которых сравниваются, выбираются в нем автоматически. Такой автоматизм естественных объединений не всегда приемлем, так как результат объединения не полностью очевиден. Глядя на такой запрос, вы не видите, какие колонки участвуют в сравнении. Для этого вам необходимо обращаться к структуре таблиц.

В: Значит, все SQL-объединения — это просто варианты внутренних объединений?

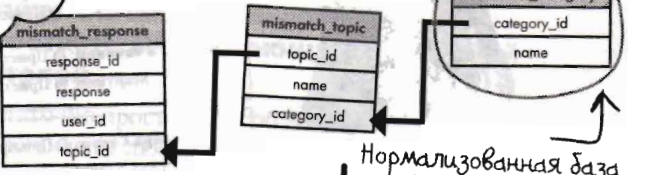
О: Нет, в вашем распоряжении имеется множество других типов объединений. Есть группа объединений с общим названием «внешние объединения» (`outer joins`), включающая разные виды объединений. К ней относятся левое внешнее объединение (`left outer join`), правое внешнее объединение (`right outer join`), полное внешнее объединение (`full outer join`) и редко используемое, но внушающее благоговейный трепет объединение по методу тройной двухсторонней спирали. Ну ладно, на самом деле такого объединения нет, а жаль! Идея, лежащая в основе внешних объединений, состоит в том, что в результат запроса включаются только те записи присоединяемой таблицы, которые не отвечают требованиям условного выражения `ON`. Отсюда следует, что всегда можно создать запрос с внешним объединением, и в результате его выполнения будут извлечены данные независимо от условий сравнения.

В зависимости от конкретных требований приложения внешние объединения могут оказаться такими же удобными, как и внутренние соединения. Чтобы узнать больше о различных видах объединений и их использовании, почитайте книгу Head First SQL.

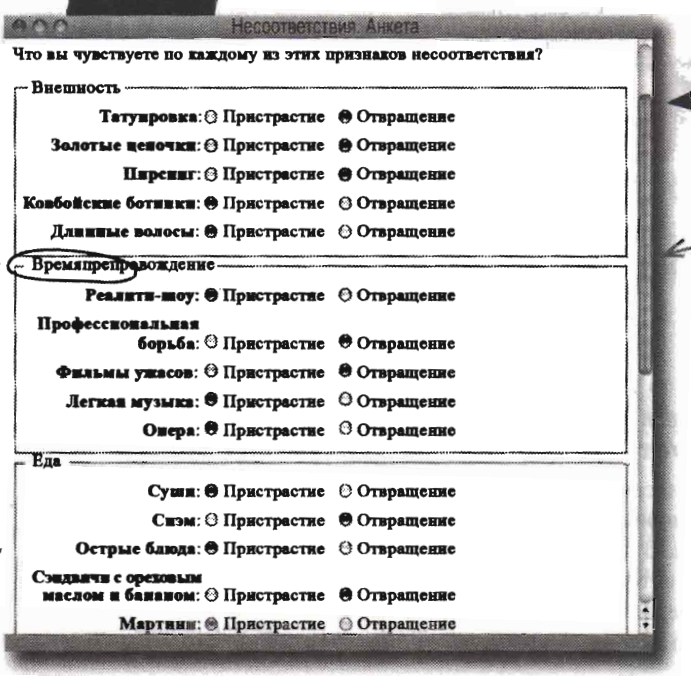
Тест-драйв

Исправьте сценарий «Анкета» так, чтобы данные по признакам несоответствия пользователей извлекались одним запросом.

Внесите в приложение «Несоответствия» изменения, обеспечивающие использование в нем единственного запроса для извлечения данных по признакам несоответствия. Загрузите новый сценарий на ваш веб-сервер и перейдите в браузере к форме «Анкета». Если все сделано без ошибок, вы... не должны увидеть никаких изменений. Но в глубине души вы знаете, что код сценария стал теперь гораздо лучше!

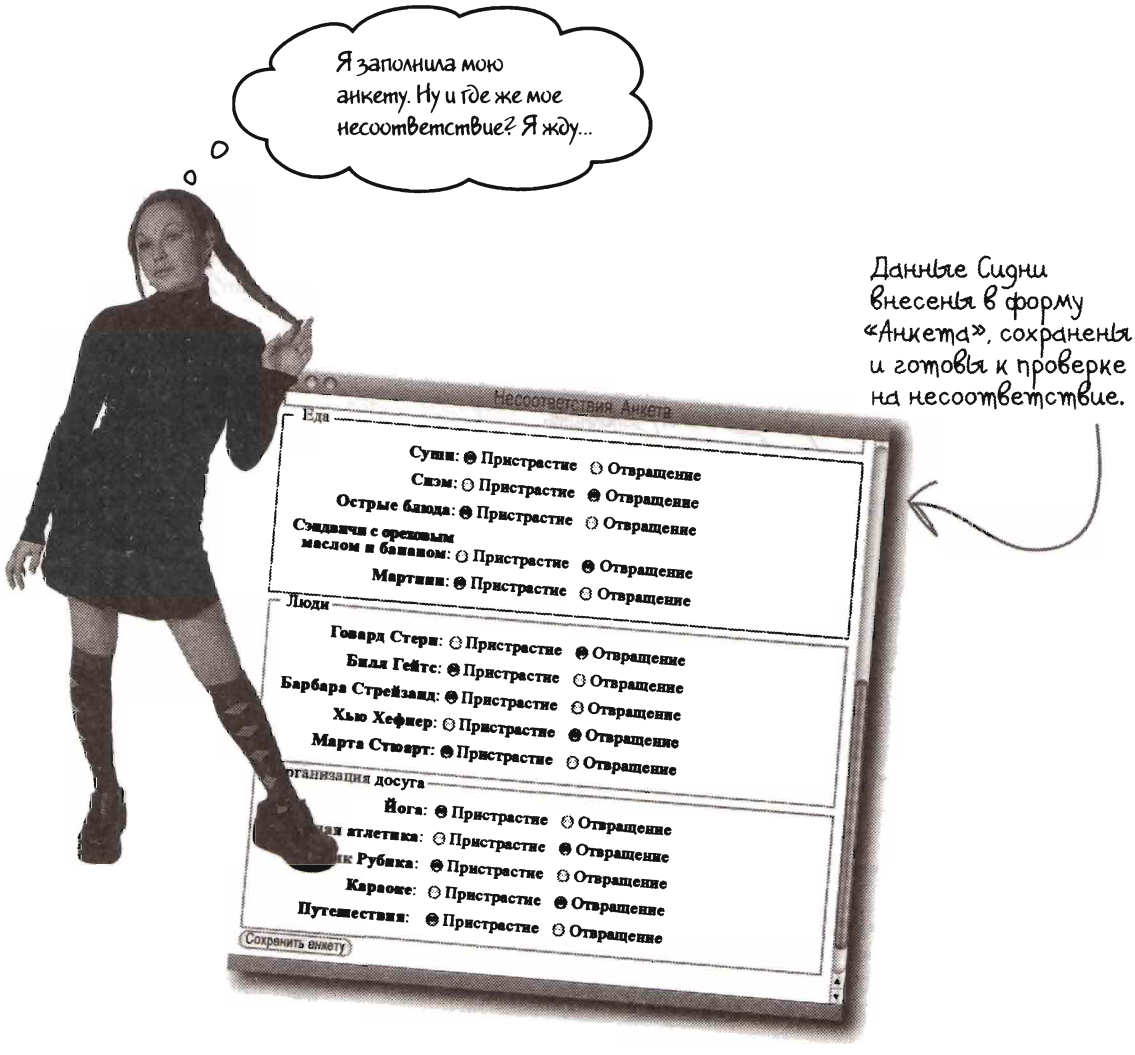


Нормализованная база данных приложения «Несоответствия» стала менее подвержена ошибкам, в том числе благодаря новой таблице, содержащей данные по категориям, к которым относятся признаки несоответствия.



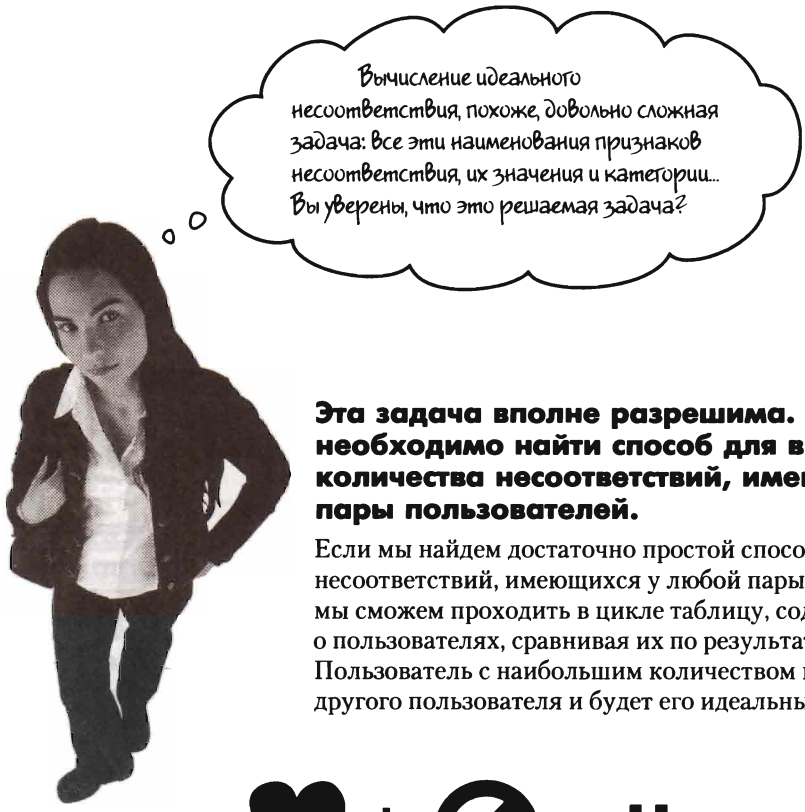
Форма «Анкета» динамически генерируется по результатам запроса с парой объединений к таблицам, содержащим данные о наименованиях признаков несоответствия, их значениях и категориях, к которым они относятся.

Теперь, когда эти данные больше не дублируются в базе, форма стала более последовательной, согласованной и понятной пользователям.



Все данные о признаках несоответствия сохранены приложением «Несоответствия», но оно пока никак не использует эти данные... не определяет несоответствующих.

Набор данных несоответствий — это еще не окончательное решение задачи, поставленной перед приложением «Несоответствия», — определения наилучшего несоответствия. В приложении «Несоответствия» еще отсутствует механизм отправки стрелы Купидона в базу данных для установки соединения между любящими сердцами. Для этого необходимо как-то проверить признаки несоответствия всех пользователей на соответствие идеальному несоответствию.



Эта задача вполне разрешима. Нам только необходимо найти способ для вычисления количества несоответствий, имеющихся у любой пары пользователей.

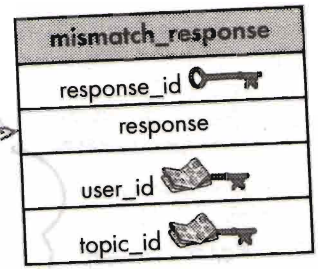
Если мы найдем достаточно простой способ вычислять количество несоответствий, имеющихся у любой пары пользователей, тогда мы сможем проходить в цикле таблицу, содержащую информацию о пользователях, сравнивая их по результатам этих вычислений. Пользователь с наибольшим количеством несоответствий для любого другого пользователя и будет его идеальным несоответствием!

Напишите, как бы вы вычисляли оценку несоответствия двух пользователей, используя данные, сохраненные в базе приложения «Несоответствия»:

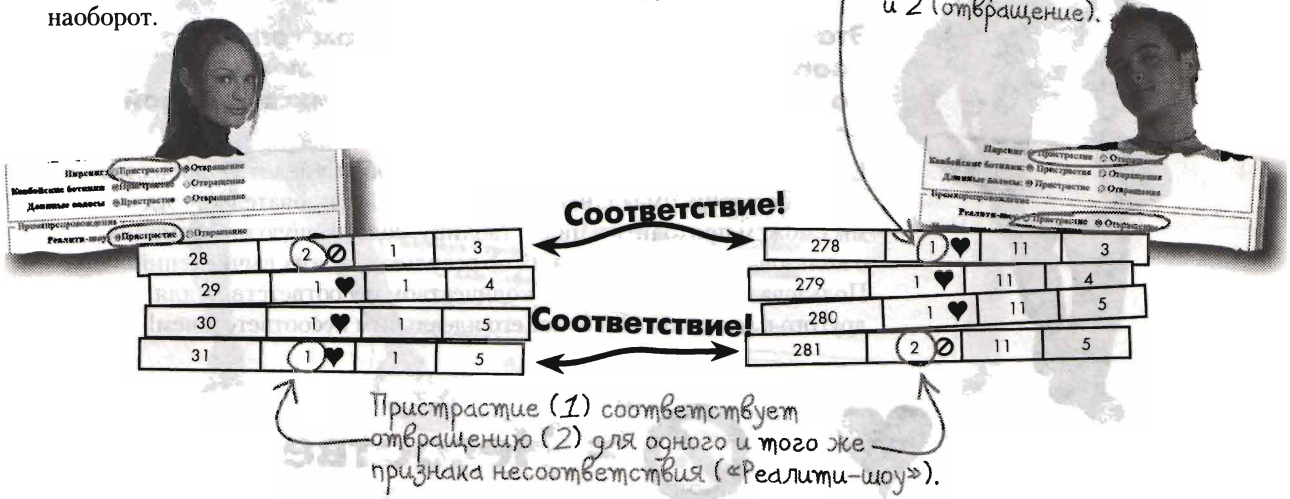
Любовь — это игра цифр

Если вы помните, признаки несоответствия сохраняются в таблице `mismatch_response` в виде чисел 0, 1 и 2, каждое из которых соответствует определенному значению признака несоответствия.

Неизвестно = 0 **Пристрастие = 1** **Отвращение = 2**



Эти данные используются для определения несоответствия между двумя пользователями, и нам необходимо найти записи двух пользователей с противоположными значениями одних и тех же признаков несоответствия. Иначе говоря, мы ищем пары записей в таблице `mismatch_response`, для которых значение колонки `response` должно быть равно 1 для одной и 2 для другой, или наоборот.



Но у нас все еще отсутствует удобный метод определения с помощью PHP-кода, когда имеет место несоответствие между одинаковыми признаками в записях двух пользователей. Конечно, можно было бы использовать две управляющие конструкции `if-else` для сравнения значений признаков несоответствия с числами 1 и 2, но решение может быть более элегантным и выразительным. Сложение двух несоответствующих признаков в любом сочетании всегда дает 3. Следовательно, для определения несоответствия между двумя признаками мы можем использовать очень простое уравнение.

Если признак несоответствия A + признак несоответствия B = = 3, имеется несоответствие!

Как видно, поиск наилучшей пары сводится к простой математике. Это решает конкретную задачу сравнения индивидуальных предпочтений, но не объясняет, как справиться с проблемой создания сценария «Мое несоответствие».

Пять этапов нахождения успешного несоответствия

Нахождение идеально несоответствующего кого-то — не просто сравнение записей, содержащих информацию о признаках несоответствия. Сценарий «Мое несоответствие» должен пройти несколько хорошо согласованных этапов для того, чтобы найти успешное несоответствие. Выполнение задач на всех этих этапах в конечном счете позволит пользователям удовлетворить свои желания, которые они испытывали, заполняя анкеты.

1. **Извлеките данные по признакам несоответствия из таблицы mismatcnh_response, не забыв объединить таблицы, содержащие данные по наименованиям признаков несоответствия и их категорий, с таблицей, содержащей значения этих признаков.**

2. **Инициализируйте переменные, в которых будут сохранены результаты поиска наилучшего несоответствия.**

3. **Пройдите в цикле таблицу, содержащую информацию о пользователях, сравнивая значения признаков несоответствия пользователя с такими же значениями других пользователей. Оценка определяется количеством противоположных значений признаков несоответствия, набранным пользователем по сравнению с другими пользователями.**

4. **После каждого прохода цикла проверьте значение текущей оценки несоответствия по сравнению с наилучшей оценкой. Если текущая оценка больше наилучшей, замените значение наилучшей оценки на значение текущей, не забыв сохранить также и значение признака несоответствия.**

5. **Убедившись, что наилучшее несоответствие найдено, запросите всю информацию о несоответствующем пользователе и покажите результат.**

Подготовка к поиску несоответствий

Операции по выполнению первого этапа нам знакомы, так как мы уже создавали объединенные запросы, подобные тем, которые нам сейчас понадобятся. Но необходимо сохранять значения признаков несоответствия пользователей для того, чтобы в дальнейшем (этап 3) была возможность сравнивать их с аналогичными значениями для других пользователей. Следующий код предназначен для создания массива `$user_responses`, в котором будут храниться значения признаков несоответствия пользователей, вошедших в приложение.

Этот запрос использует объединение, чтобы извлечь информацию о признаках несоответствия пользователя.

```
$query = "SELECT mr.response_id, mr.topic_id, mr.response, mt.name
AS topic_name " .
  "FROM mismatch_response AS mr " .
  "INNER JOIN mismatch_topic AS mt " .
  "USING (topic_id) " .
  "WHERE mr.user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$user_responses = array();
while ($row = mysqli_fetch_array($data)) {
  array_push($user_responses, $row);
}
```

Цикл while используется для того, чтобы в результате запроса пройти все записи и создать массив, содержащий информацию о признаках несоответствия пользователя.

Когда обработка в цикле while закончится, массив будет содержать информацию обо всех признаках несоответствия пользователя.

СДЕЛАНО

1. **Извлеките данные по признакам несоответствия из таблицы mismatcnh_response, не забыв объединить таблицы, содержащие данные по наименованиям признаков несоответствия и их категорий, с таблицей, содержащей значения этих признаков.**

Операции по выполнению второго этапа сценария «Мое несоответствие» включают создание переменных, в которых будут содержаться результаты поиска несоответствия. Эти переменные будут использоваться во всем сценарии «Мое несоответствие» в процессе поиска наилучшего несоответствия:

```
$mismatch_score = 0;
$mismatch_user_id = -1;
$mismatch_topics = array();
```

Это идентификатор пользователя, который проверяется на соответствие потенциальному статусу «Мое несоответствие»... В конце поиска эта переменная содержит идентификатор пользователя с наилучшей оценкой несоответствия.

Эта переменная содержит оценку несоответствия между двумя пользователями: чем выше оценка, тем лучше несоответствие.

Если значение этой переменной после окончания поиска осталось равным -1, это говорит нам о том, что ни один из остальных пользователей не заполнил анкету, что, конечно, очень маловероятно.

Этот массив содержит значения признаков несоответствия между двумя пользователями.

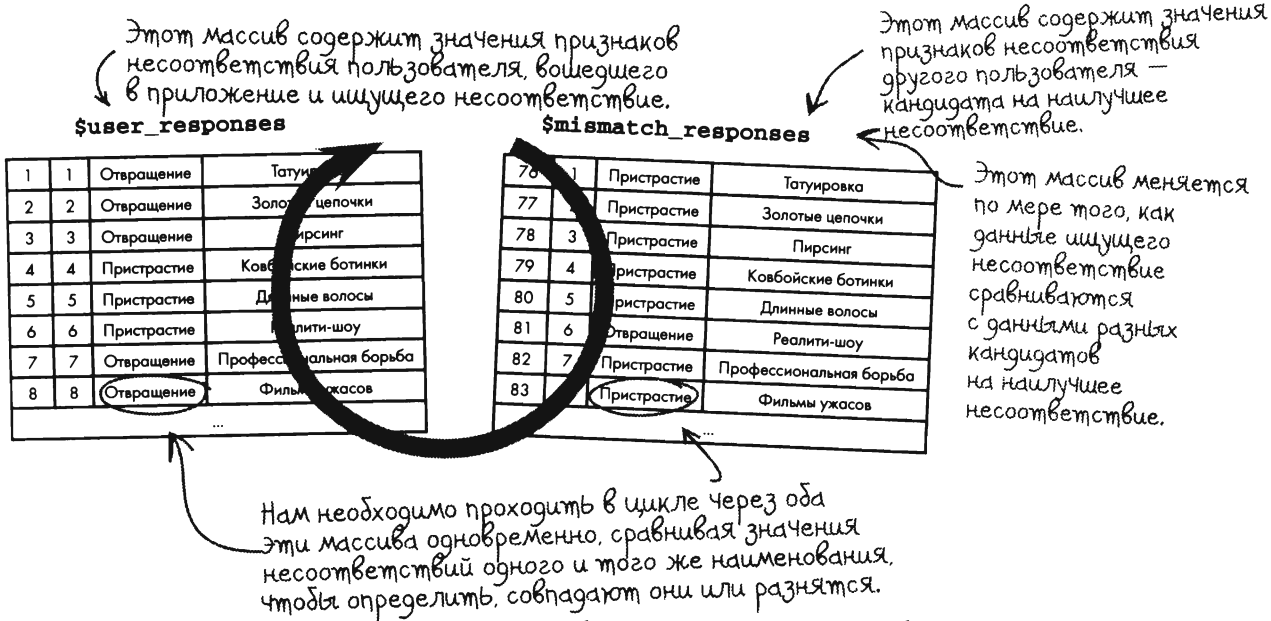
СДЕЛАНО

2. **Инициализируйте переменные, в которых будут сохранены результаты поиска наилучшего несоответствия.**

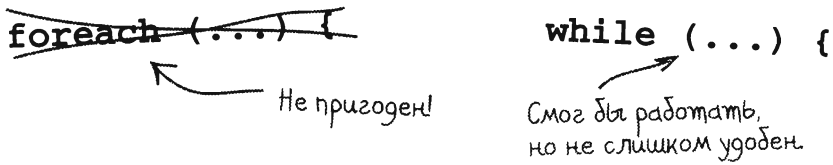
Испытание пользователей на несоответствие

Следующий этап сценария «Мое несоответствие» предусматривает прохождение в цикле по данным каждого пользователя и сравнение значений его признаков несоответствия со значениями признаков несоответствия пользователя, вошедшего в приложение. Иначе говоря, мы берем запись пользователя, вошедшего в приложение, то есть **ищущего несоответствие** (например, Сидни), и, проходя в цикле через записи всех пользователей, сравниваем значения его признаков несоответствия со значениями признаков несоответствия этих пользователей, то есть **кандидатов на наилучшее несоответствие**. Мы ищем для проверяемого пользователя кандидата с наивысшей оценкой несоответствия.

С чего начать? Что скажете по поводу прохождения в цикле элементов массива `$user_responses` (значений признаков несоответствия)? Внутри цикла мы будем сравнивать значение каждого элемента со значениями элементов другого массива, который содержит значения признаков несоответствия кандидатов. Мы назовем этот массив `$mismatch_responses`.



Сложность задачи заключается в том, что нам необходим цикл, в котором мы будем проходить по элементам двух массивов одновременно, сравнивая подобные элементы друг с другом. Цикл `foreach` не подходит, так как он может проходить по элементам только одного массива, а нам нужно проходить по элементам двух массивов **одновременно**. Цикл `while` подошел бы, но нам пришлось бы создать переменную, счетчик цикла, и вручную увеличивать ее значение при каждом проходе цикла. В идеале нам нужен цикл, автоматически подсчитывающий количество проходов и сохраняющий эти значения в переменной, которую мы сможем использовать для доступа к элементам каждого из массивов.



Все, что нам необходимо, — это цикл for

PHP предлагает еще один тип цикла с функциональностью, которая необходима нам для сравнения признаков несоответствия в сценарии «Мое несоответствие». Это цикл `for`, и он очень удобен для повторения какого-либо фрагмента кода известное количество раз. Например, цикл `for` **удобен** для операций счета, таких как обратный счет до нуля или прямой счет до какого-то значения. Ниже приведена структура цикла `for`, из которой хорошо видно, как можно построить цикл для прохождения по элементам массива, используя счетчик цикла (`$i`).

Инициализация счетчика цикла, то есть присвоение этой переменной какого-либо значения перед началом цикла.

Инициализация
Начало цикла. Значение счетчика цикла устанавливается равным 0.

Условное выражение
Следующий проход цикла будет выполнен только в том случае, если условное выражение имеет значение `true`, то есть если значение переменной `$i` меньше, чем количество элементов массива с данными по признакам несоответствия.

Обновление
Обновление значения счетчика цикла увеличением его на единицу.

```
for ($i = 0; $i < count($user_responses); $i++) {
  ...
}
```

Любой код, помещенный между фигурными скобками, будет выполняться при каждом прохождении цикла.

Функция `count()` возвращает количество элементов массива, что используется в условном выражении цикла.

Обновление значения счетчика цикла прибавлением единицы. `$i++` — это то же самое, что `$i = $i + 1`.

Этап 3 сценария «Мое несоответствие» включает сравнение двух пользователей при прохождении в цикле всех их данных по признакам несоответствия и вычисление оценки, основанной на количестве встреченных между ними несоответствий. Используя представленные ниже данные, закончите цикл `for`, в котором вычисляются эти оценки.

`$user_responses`
Массив, содержащий данные по признакам несоответствия пользователя.

`$mismatch_responses`
Массив, содержащий данные по признакам несоответствия пользователя — кандидата на наилучшее несоответствие.

`$score`
Оценка несоответствия, которая будет вычисляться в цикле.

```
for ($i = 0; $i < count($user_responses); $i++) {

  if (..... + ..... == .....) {

    .....
    array_push($topics, $user_responses[$i]['topic_name']);
  }
}
```

не бывает глупых вопросов

В: Почему для вычисления оценки вместо цикла for просто не использовать цикл foreach?

О: Хотя в цикле `foreach` мы легко сможем пройти все элементы массива, содержащего данные по признакам несоответствия, этот цикл не предоставляет нам индекс элемента, который рассматривается в текущем проходе. Данный индекс очень важен потому, что он используется для доступа к элементам обоих массивов: как массива, содержащего данные по признакам несоответствия пользователя, вошедшего в приложение, так и массива данных кандидата на наилучшее несоответствие. В цикле, который нам необходим, мы могли бы обойтись без знания индекса для одного из массивов, но не для обоих. Поэтому нам необходим обычный цикл `for`, в котором, используя его переменную — счетчик цикла — как индекс, мы можем получить доступ к элементам обоих массивов.

В: С какой целью мы сохраняем данные по признакам несоответствия в своем собственном массиве?

О: Массив данных по признакам несоответствия необходим для того, чтобы дать пользователю понять, как он соотносится по каждому из признаков несоответствия с пользователем, который является потенциальным идеальным несоответствием. Недостаточно просто сообщить пользователю, кто именно является его идеальным несоответствием, лучше предоставить ему данные, по каким конкретным признакам они не соответствуют друг другу. Это даст пользователю более полный результат, на основании которого он сможет понять, почему этот конкретный человек так идеально ему не соответствует.

В: Относительно 5-го этапа сценария «Мое несоответствие»… Как это может быть, что для пользователя не будет найдено наилучшего несоответствия?

О: Хотя это и маловероятно, вы можете представить себе сценарий, при котором в системе имеет учетную запись только один пользователь. В этом случае не будет никого, кто бы мог ему не соответствовать.

Этап 3 сценария «Мое несоответствие» включает сравнение двух пользователей при прохождении в цикле всех их данных по признакам несоответствия и вычисление оценки, основанной на количестве встреченных между ними несоответствий. Используя представленные ниже данные, закончите цикл `for`, в котором вычисляются эти оценки.

$user_responses

1	1	Пристрастие	Татуировка
2	2	Пристрастие	Золотые цепочки
3	3	Пристрастие	Пирсинг
4	4	Пристрастие	Ковбойские ботинки
5	5	Пристрастие	Длинные волосы
6	6	Отвращение	Реалити-шоу
7	7	Пристрастие	Профессиональная борьба
8	8	Пристрастие	Фильмы ужасов
...			

$mismatch_responses

76	1	Отвращение	Татуировка
77	2	Отвращение	Золотые цепочки
78	3	Отвращение	Пирсинг
79	4	Пристрастие	Ковбойские ботинки
80	5	Пристрастие	Длинные волосы
81	6	Пристрастие	Реалити-шоу
82	7	Отвращение	Профессиональная борьба
83	8	Отвращение	Фильмы ужасов
...			

$score

Помните? Количество значений признаков несоответствия, введенных пользователем, точно равно количеству их наименований, так как мы создавали форму ввода данных «Анкета» исходя из этих данных.

Переменная $score может принимать значения от нуля (ни одного несоответствия) до общего количества наименований несоответствий (полное несоответствие).

Оценка увеличивается на единицу после каждого найденного несоответствия.

```
for ($i = 0; $i < count($user_responses); $i++) {
  if ($user_responses[$i]['response'] + $mismatch_responses[$i]['response'] == 3 ) {
    $score += 1;
    array_push($topics, $user_responses[$i]['topic_name']);
  }
}
```

Счетчик цикла используется для получения доступа к каждому элементу массива, содержащего данные по признакам несоответствия пользователя, вошедшего в приложение.

В массив добавляется наименование каждого признака несоответствия, чтобы пользователь смог их увидеть, когда будет смотреть результаты поиска несоответствий.

Несоответствие возникает тогда, когда у одного пользователя признак несоответствия имеет значение 1 («Пристрастие»), а у другого 2 («Отвращение»), поэтому сложение этих двух значений при несоответствии всегда даст 3.

СДЕЛАНО

3 **Пройдите в цикле таблицу, содержащую информацию о пользователях, сравнивая значения признаков несоответствия пользователя с такими же значениями других пользователей. Оценка определяется количеством противоположных значений признаков несоответствия, набранным пользователем по сравнению с другими пользователями.**

Завершение приложения «Несоответствия»

Новый цикл, в котором происходит вычисление оценок несоответствий, является частью большого сценария (mymismatch.php), задача которого — найти для пользователя идеальное несоответствие в базе данных приложения и вывести эти данные на экран монитора.

Этот сценарий находит для пользователя его идеальное несоответствие!

mymismatch.php

```
// Поиск несоответствий производится, только если пользователь
// уже внес в анкету свои значения признаков несоответствия
$query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
if (mysqli_num_rows($data) != 0) {
  // Вначале извлечение значений признаков несоответствия из таблицы,
  // содержащей информацию о признаках несоответствия пользователей
1 // (для получения наименований признаков несоответствия используется объединение — JOIN)
  $query = "SELECT mr.response_id, mr.topic_id, mr.response, mt.name AS topic_name " .
    "FROM mismatch_response AS mr " .
    "INNER JOIN mismatch_topic AS mt USING (topic_id) " .
    "WHERE mr.user_id = '" . $_SESSION['user_id'] . "'";
  $data = mysqli_query($dbc, $query);
  $user_responses = array();
  while ($row = mysqli_fetch_array($data)) {
    array_push($user_responses, $row);
  }
// Инициализация переменных, в которых будут сохранены результаты поиска
2 $mismatch_user_id = -1;
  $mismatch_topics = array();
  $mismatch_score = 0;
```

Несоответствие может быть найдено только для того пользователя, который заполнил анкету.

Здесь используется уже знакомое нам объединение, чтобы извлечь наименование признака несоответствия при извлечении информации о несоответствиях пользователя.

В массиве $user_responses сохраняются все данные по признакам несоответствия пользователя.

Эти переменные отслеживают параметры процесса поиска несоответствия.

Подождите, это еще не все, переверните страницу!

```
    // Проход в цикле записей в таблице, содержащей информацию о пользователях,
    // и сравнение значений признаков несоответствия пользователя с такими же значениями других пользователей
    $query = "SELECT user_id FROM mismatch_user WHERE user_id != '" . $_SESSION['user_id'] . "'";
    $data = mysqli_query($dbc, $query);

    while ($row = mysqli_fetch_array($data)) {

      // Извлечение значений признаков несоответствий для пользователя (кандидата на наилучшее несоответствие)
      $query2 = "SELECT response_id, topic_id, response FROM mismatch_response" .
                " WHERE user_id = " . $row['user_id'] . "";
      $data2 = mysqli_query($dbc, $query2);

      $mismatch_responses = array();

      while ($row2 = mysqli_fetch_array($data2)) {
        array_push($mismatch_responses, $row2);
      }

    // Сравнение значений признаков несоответствия и вычисление оценки несоответствия
      $score = 0;

      $topics = array();

      for ($i = 0; $i < count($user_responses); $i++) {

      if ($user_responses[$i]['response'] + $mismatch_responses[$i]['response'] == 3) { $score += 1;
        array_push($topics, $user_responses[$i]['topic_name']);
      }

      }

      // Оценка несоответствия текущего пользователя
      // сравнивается с наилучшей оценкой на данный момент
      if ($score > $mismatch_score) {

        // Найдено лучшее несоответствие, поэтому переменные,
        // отслеживающие параметры процесса поиска, обновляются.

        $mismatch_score = $score;

        $mismatch_user_id = $row['user_id'];

        $mismatch_topics = array_slice($topics, 0);

      }

    }

...
```

3

4

СДЕЛАНО

В результате выполнения этого запроса будут извлечены записи всех пользователей, за исключением того, для которого ищется несоответствие.

В результате выполнения этого запроса для поиска наилучшего несоответствия извлекается информация о несоответствиях, занесенных пользователями в анкету.

Фигурные скобки ограничивают блок кода цикла while.

Это цикл for, в котором вычисляется оценка несоответствия для поиска кандидата на наилучшее несоответствие.

Переменная текстового типа '2' приводится к типу int (2), для того чтобы ее можно было прибавлять и сравнивать с другими переменными типа int.

Если у текущего пользователя обнаружено лучшее несоответствие, чем найденное до сих пор, то присваиваем значение лучшего несоответствия этому пользователю.

Эта функция извлекает группу элементов массива. В данном случае мы используем ее для того, чтобы просто скопировать массив $topics в массив $mismatch_topics.

Мы все еще внутри блока кода первой управляющей конструкции if, которая началась на предыдущей странице, а код все еще не закончился...

```
// Проверка того, найдено ли несоответствие
if ($mismatch_user_id != -1) {
  $query = "SELECT username, first_name, last_name, city, state, picture FROM mismatch_user"
         " WHERE user_id = '$mismatch_user_id'";
  $data = mysqli_query($dbc, $query);
  if (mysqli_num_rows($data) == 1) {
    // Запись пользователя с наилучшим несоответствием найдена в таблице,
    // вывод информации об этом пользователе
    $row = mysqli_fetch_array($data);
    echo '<table><tr><td class="label">';
    if (!empty($row['first_name']) && !empty($row['last_name'])) {
      echo $row['first_name'] . ' ' . $row['last_name'] . '<br />';
    }
    if (!empty($row['city']) && !empty($row['state'])) {
      echo $row['city'] . ', ' . $row['state'] . '<br />';
    }
    echo '</td><td>';
    if (!empty($row['picture'])) {
      echo '<img src="' . MM_UPLOADPATH . $row['picture'] . '" alt="Фотография пользователя" /><br />';
    }
    echo '</td></tr></table>';
    // Вывод значений признаков несоответствия
    echo '<h4>Вы не соответствуете по следующим ' . count($mismatch_topics) . '
признакам:</h4>';
    foreach ($mismatch_topics as $topic) {
      echo $topic . '<br />';
    }
    // Вывод гиперссылки на профиль пользователя с наилучшим несоответствием
    echo '<h4>' . 'Просмотр профиля <a href=viewprofile.php?user_id=' . $mismatch_user_id . '>'
$row['first_name'] . ' </a>' . '</h4>';
  }
  }
}
else {
  echo '<p>Вы должны <a href="questionnaire.php">заполнить анкету, </a> ' .
    'прежде чем для вас может быть найдено несоответствие.</p>';
}
```

Прежде чем выводить результат поиска, нужно убедиться, что наилучшее несоответствие действительно найдено.

Запрос информации о кандидате на наилучшее несоответствие.

Вывод имени пользователя.

Вывод города и штата пользователя.

Не забудьте сгенерировать тег <img> для вывода фотографии пользователя.

Очень важно показать, по каким из признаков наблюдается несоответствие.

В конце мы предоставляем гиперссылку на профиль пользователя с наилучшим несоответствием, чтобы пользователь, вошедший в приложение, мог узнать о нем больше.

СДЕЛАНО

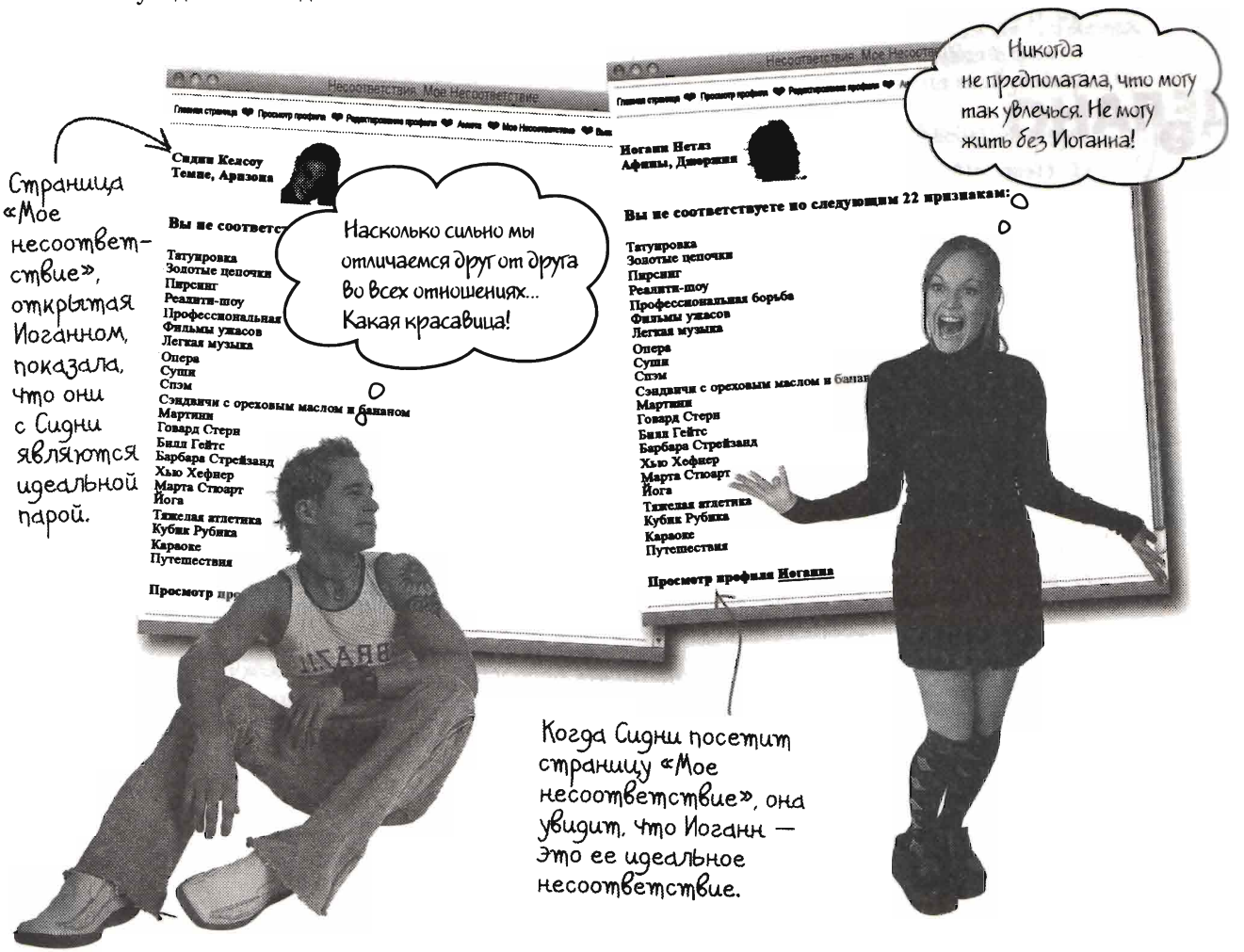
5

Тест-драйв

Найдите свое идеальное несоответствие.

Внесите в приложение «Несоответствия» изменения, обеспечивающие использование в нем нового сценария «Мое несоответствие» (или загрузите приложение с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Это требует создания нового сценария `mymismatch.php`, а также добавление строки «Мое несоответствие» в сценарий `navmenu.php`, чтобы у пользователя была возможность получить доступ к странице «Мое несоответствие».

Загрузите все сценарии на ваш веб-сервер и откройте главную страницу (`index.php`) приложения «Несоответствия» в браузере. Войдите в приложение, заполните форму «Анкета» и щелкните кнопкой мыши по гиперссылке «Мое несоответствие», чтобы увидеть ваше идеальное несоответствие.

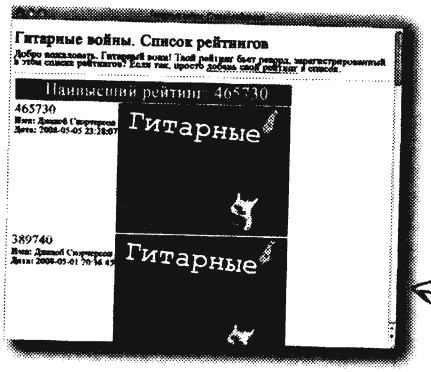


Структура базы данных и магниты

Помните приложение «Гитарные войны», с момента создания которого прошла вечность? Ваша задача сейчас — рассмотреть базу данных этого приложения, которой можно было бы помочь в вопросах нормализации, что позволит значительно ее усовершенствовать. Используйте магниты, расположенные внизу, чтобы воплотить в жизнь эти идеи, и не забудьте также про первичные и внешние ключи.

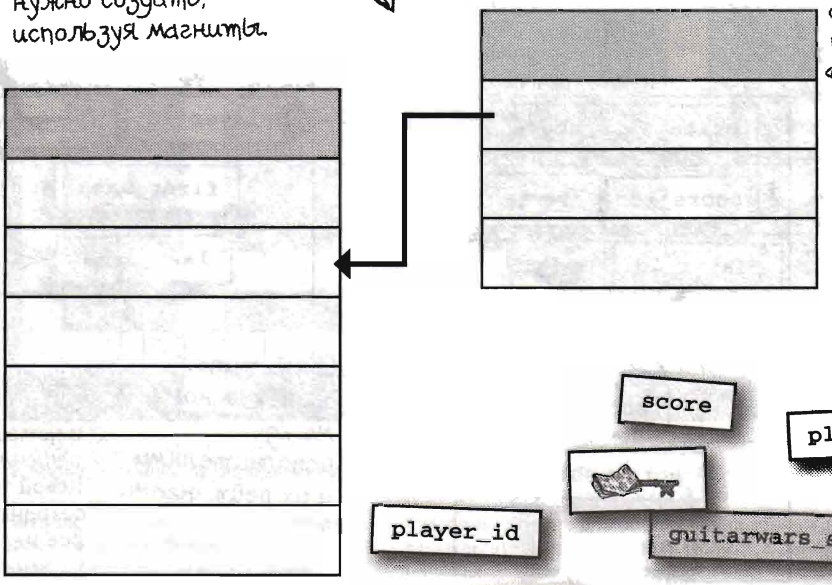
Это первоначальная база данных приложения «Гитарные войны», в которой сохраняются рейтинги, введенные пользователями.

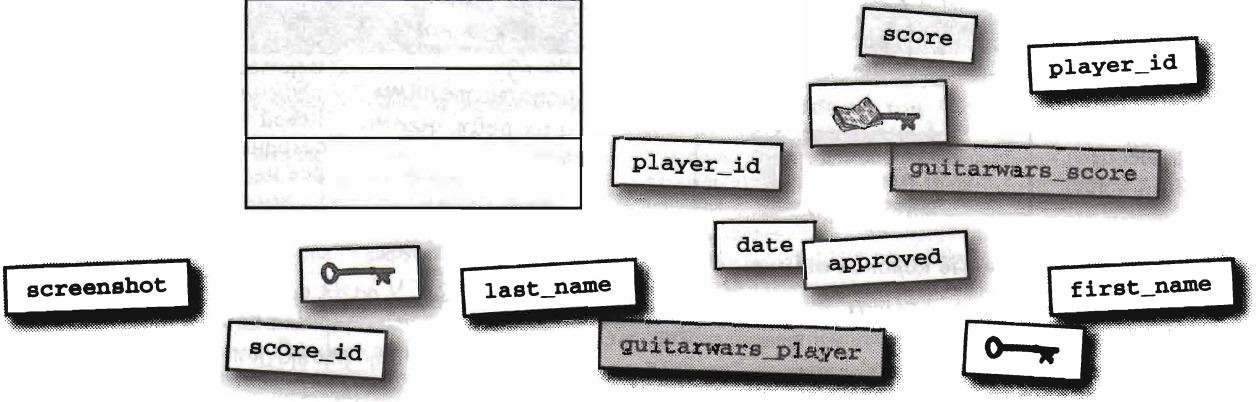
guitarwars
date
name
score
screenshot
approved



Вывод рейтингов на главной странице приложения «Гитарные войны» управляется базой данных.

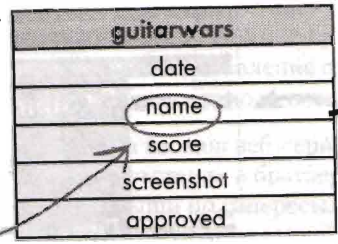
Это новая, улучшенная схема, которую вам нужно создать, используя магниты.





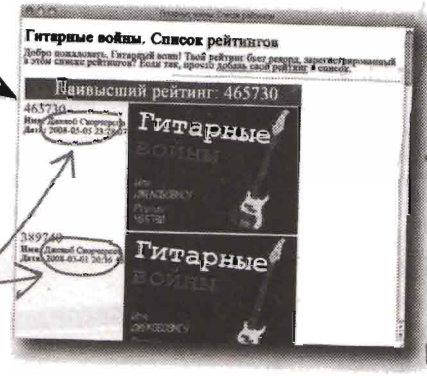
Структура базы данных и магниты. Решение

Помните приложение «Гитарные войны», с момента создания которого прошла вечность? Ваша задача сейчас — рассмотреть базу данных этого приложения, которой можно было бы помочь в вопросах нормализации, что позволит значительно ее усовершенствовать. Используйте магниты, расположенные внизу, чтобы воплотить в жизнь эти идеи, и не забудьте также про первичные и внешние ключи.



У таблицы нет первичного ключа, который является очень важной частью нормализованной базы данных.

Так как один и тот же пользователь может отправить несколько рейтингов, данные в колонке name могут дублироваться... Это плохо.

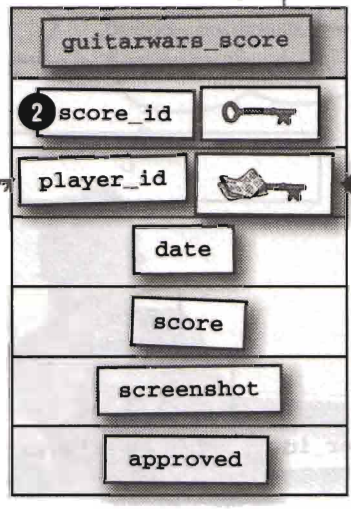


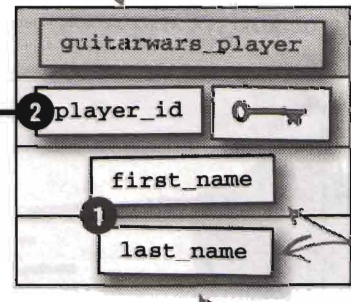
Вывод рейтингов на главной странице приложения «Гитарные войны» управляется базой данных.

Имя каждого пользователя теперь разбито на имя и фамилию для лучшей атомизации, и эти данные сохраняются теперь только один раз независимо от того, сколько раз пользователь отправлял свои рейтинги.

Новая колонка score_id используется в качестве первичного ключа, так необходимого для нормализованной базы данных.

Таблицы получают новые наименования, так как теперь они служат для сохранения более конкретных данных.





Ссылка на таблицу, содержащую рейтинги пользователей, производится через новый внешний ключ.

Связь типа один-к-одному между исполнителями и их рейтингами.

Задача атомизации имен пользователей и устранения дублирования решена путем создания новой таблицы, в которой сохранены эти имена. Все необходимые связи устанавливаются через ключи.

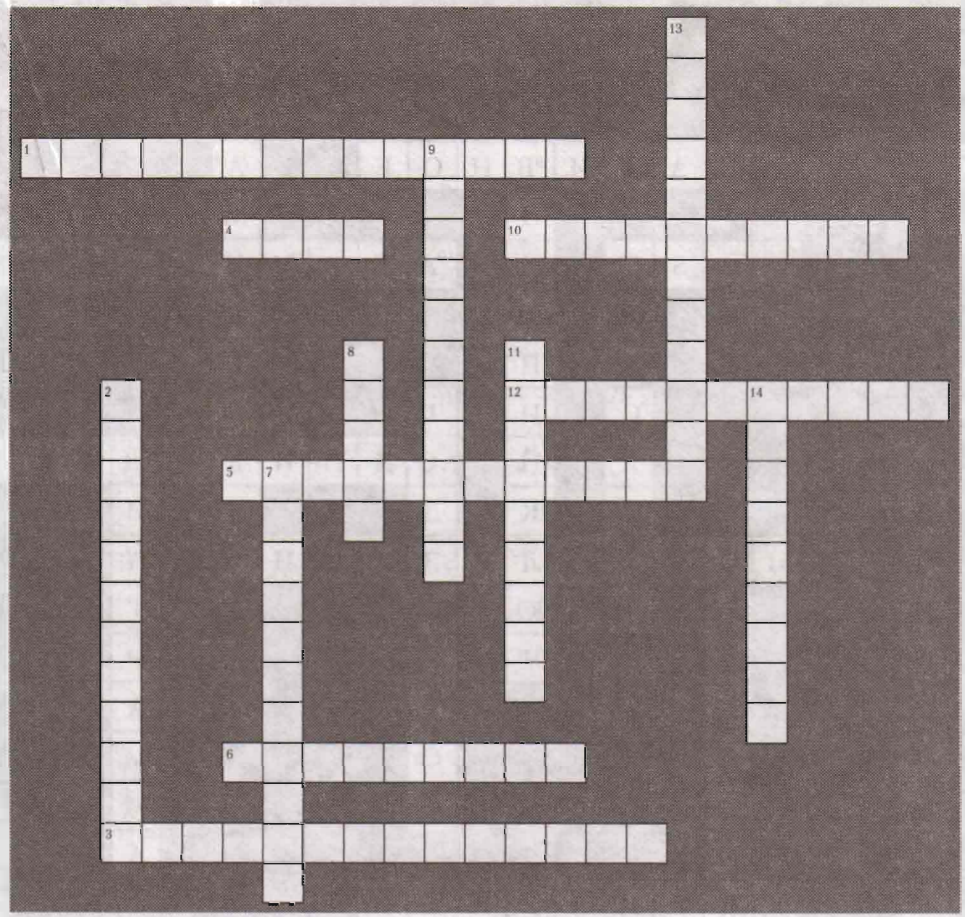
1. **Убедитесь, что данные колонок атомарны.**
2. **Создайте для каждой таблицы первичный ключ.**
3. **Убедитесь, что неключевые колонки не зависят друг от друга.**

Это те же правила еще раз — только лишь для того, чтобы убедиться, что вы их не забыли!

3 У базы данных приложения «Гитарные войны» отсутствуют проблемы.

Кроссворд PHP и MySQL

Расстраиваетесь, что ваше собственное идеальное несоответствие все еще не найдено? Отвлекитесь от этих грустных мыслей, решая этот кроссворд.



По горизонтали

1. Временное имя, используемое для ссылки на определенную информацию в запросе.

3. Тип связи, при которой множество записей одной таблицы связано со множеством записей другой таблицы.

4. Это позволяет вам конвертировать данные одних типов в данные других типов.

5. Процесс устранения избыточности и других структурных проблем в базе данных.

6. Вы можете сократить некоторые управляющие конструкции if-else с помощью этого удобного небольшого оператора.

10. Это не имеет отношения к атомной энергии. Это просто изменение размеров данных до минимальных, при которых они сохраняют свой смысл в приложении.

12. Тип связи, при котором одна запись одной таблицы связана с одной записью другой таблицы.

По вертикали

2. Тип связи, при котором одна запись одной таблицы связана со множеством записей другой таблицы.

7. Используйте это для объединения данных, полученных в результате запроса к одной таблице, с данными из другой таблицы.

8. Представление всех объектов вашей базы данных, таких как таблицы и колонки, и их взаимных связей.

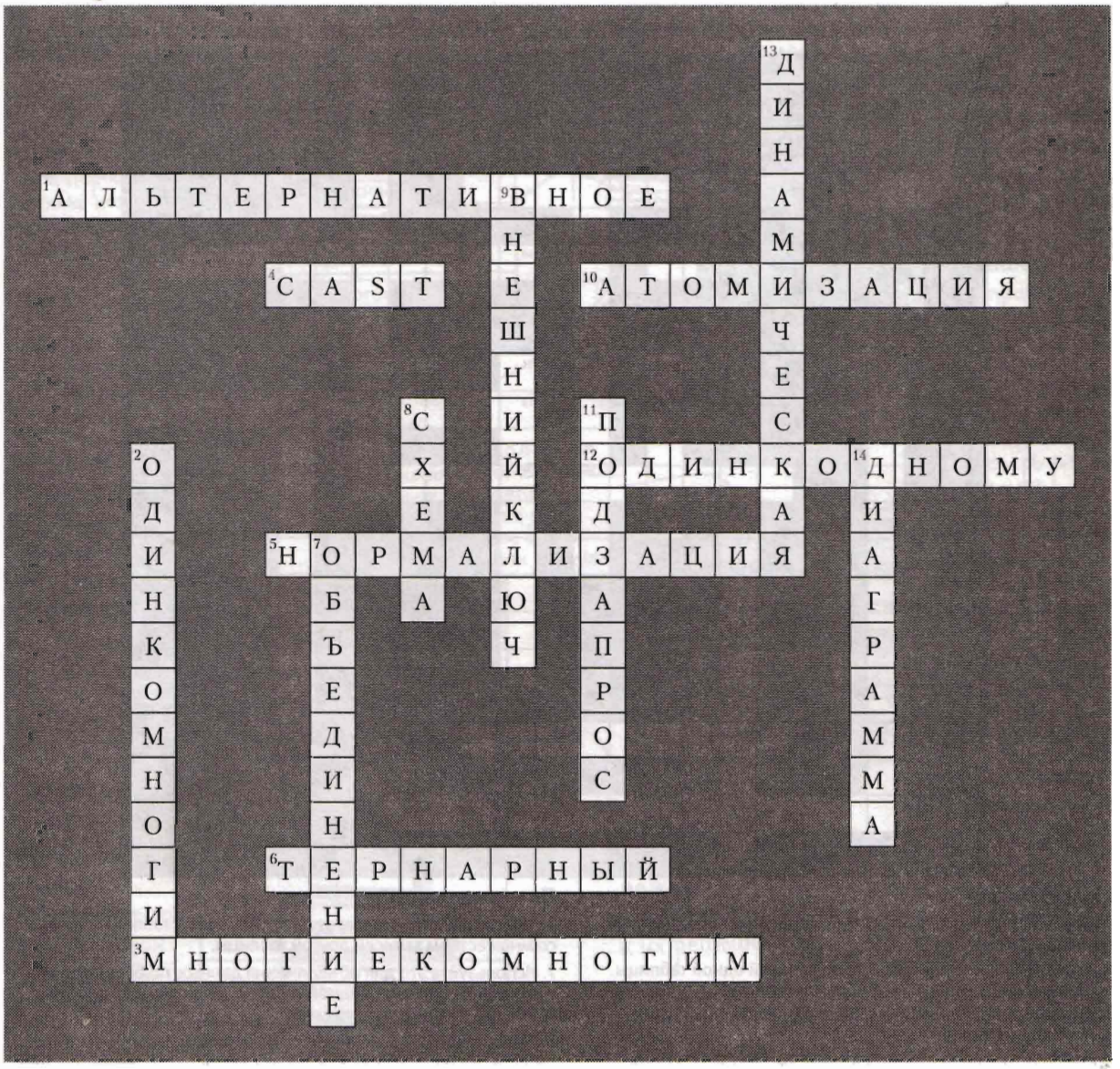
9. Колонка в таблице, ссылающаяся на первичный ключ в другой таблице.

11. Объединения помогают избавиться от этих вещей.

13. Когда форма генерируется исходя из данных таблицы, она считается _____.

14. То, что может оказать большую помощь при составлении структуры базы данных.

Кроссворд PHP и MySQL. Решение



Ваш инструментарий PHP и MySQL

В этой главе вы познакомились еще с несколькими приемами работы с базой данных MySQL, не говоря уже о некоторых новых познаниях в PHP. Давайте сделаем резюме.

Нормализация

Это процесс изменения структуры базы данных для устранения дублирования данных и оптимизации размещения данных и связей между ними. Основная цель нормализации — добиться структуры базы данных, обеспечивающей высокий уровень устойчивости и надежности в условиях увеличения объема сохраняемых в ней данных.

Схема базы данных и диаграммы

Схема базы данных — это представление ее структуры (таблиц, колонок и т. д.) и связей между ними. Диаграмма — это визуальное представление вашей базы данных, включая детали, описывающие колонки, ответственные за связи между таблицами.

Внешний ключ

Колонка в таблице, предназначенная для связи ее с другой таблицей. Значения внешнего ключа дочерней таблицы обычно соответствуют значениям первичного ключа главной таблицы, чем достигается эффективная связь записей этих двух таблиц.

for (...)

Цикл, идеально подходящий для обработки группы данных с определенным количеством элементов в ней. При создании цикла инициализируется специальная переменная (счетчик), задается условное выражение и определяется, как должны изменяться значения счетчика при прохождении цикла.

? :

Этот тернарный оператор является PHP-конструкцией, которая выполняет функцию компактной управляющей конструкции if-else. Он очень удобен для простого выбора между двумя значениями на основании значения (true или false) условного выражения.

Внутреннее объединение (INNER JOIN)

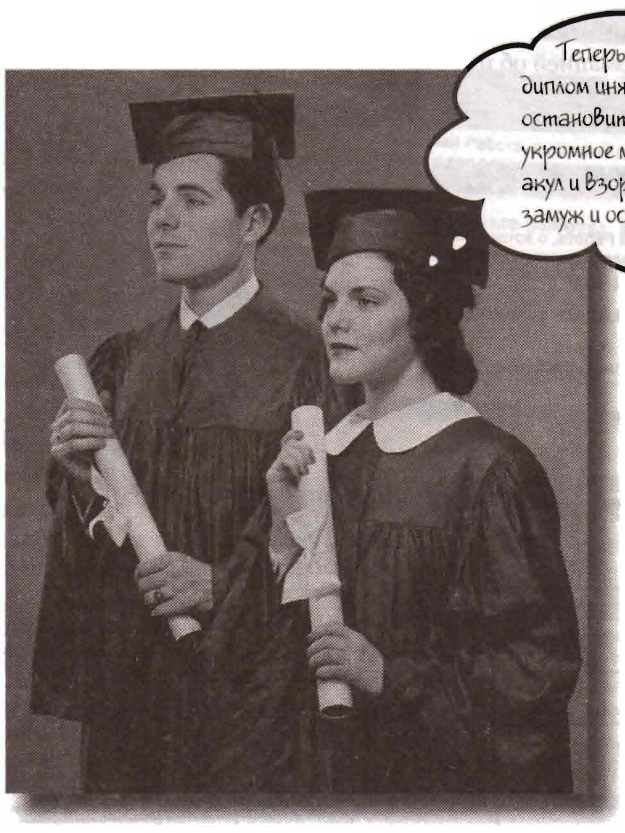
Объединяет данные двух таблиц, извлеченные из соответствующих записей. В отличие от обычного запроса объединенный запрос позволяет извлекать данные из более чем одной таблицы, что исключительно важно для баз данных, включающих множество таблиц.

AS *имя*

SQL-выражение, устанавливающее альтернативное имя для данных внутри запроса. Альтернативное имя часто используется для упрощения запроса путем замены длинных имен таблиц, или колонок на короткие. Оно также может использоваться для замены имен колонок, если они не слишком информативны в результате запроса.

9 текстовые строки и пользовательские функции

Функции облегчают жизнь

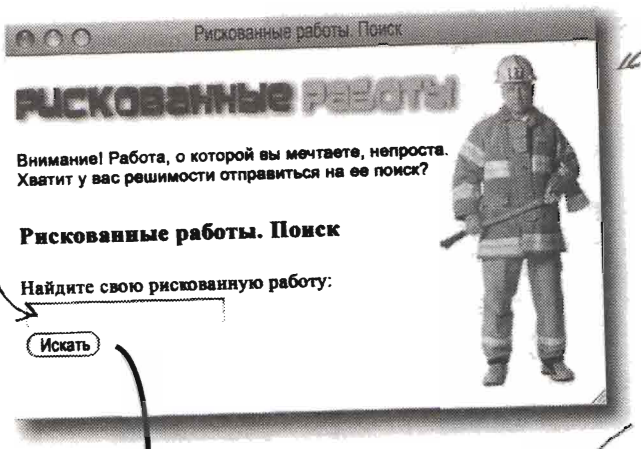


Функции переводят ваше приложение на совершенно новый уровень. Вы уже использовали встроенные PHP-функции для решения определенных задач. Наступило время рассмотреть еще несколько исключительно полезных встроенных функций. А затем вы научитесь создавать свои собственные пользовательские функции для того, чтобы продвинуться до такого уровня, что даже сложно себе представить, что это вообще возможно. Ладно, может, не до такого, на котором вы могли бы разводить лазерных акул, но, по крайней мере, пользовательские функции помогут оптимизировать ваш код и сделать его пригодным для многократного использования.

Хорошую рискованную работу не так легко найти

Задача недавно созданной в Интернете компании `RiskyJobs.biz` заключается в том, чтобы помогать различным компаниям в поиске людей, способных выполнять рискованные работы. Бизнес-модель достаточно проста: за каждую рискованную работу, для которой мы можем представить кандидата, мы получаем комиссионное вознаграждение. Чем больше найденный нами работник соответствует характеру работы, тем выше ставка этих комиссионных.

Компании «Рискованные работы» необходима помощь в улучшении функциональности поисковой системы их сайта. К настоящему времени уже имеется база данных с информацией о рискованных работах, ждущих своей очереди быть обнаруженными людьми, способными взяться за их исполнение. Давайте посмотрим на поисковую форму приложения «Рискованные работы» и базу данных с информацией об имеющихся работах.



Эта простая форма поиска вызывает сценарий, который ищет нужную запись в таблице riskyjobs.

Форма поиска рискованных работ инициирует запрос на поиск в таблице riskyjobs нужной работы.

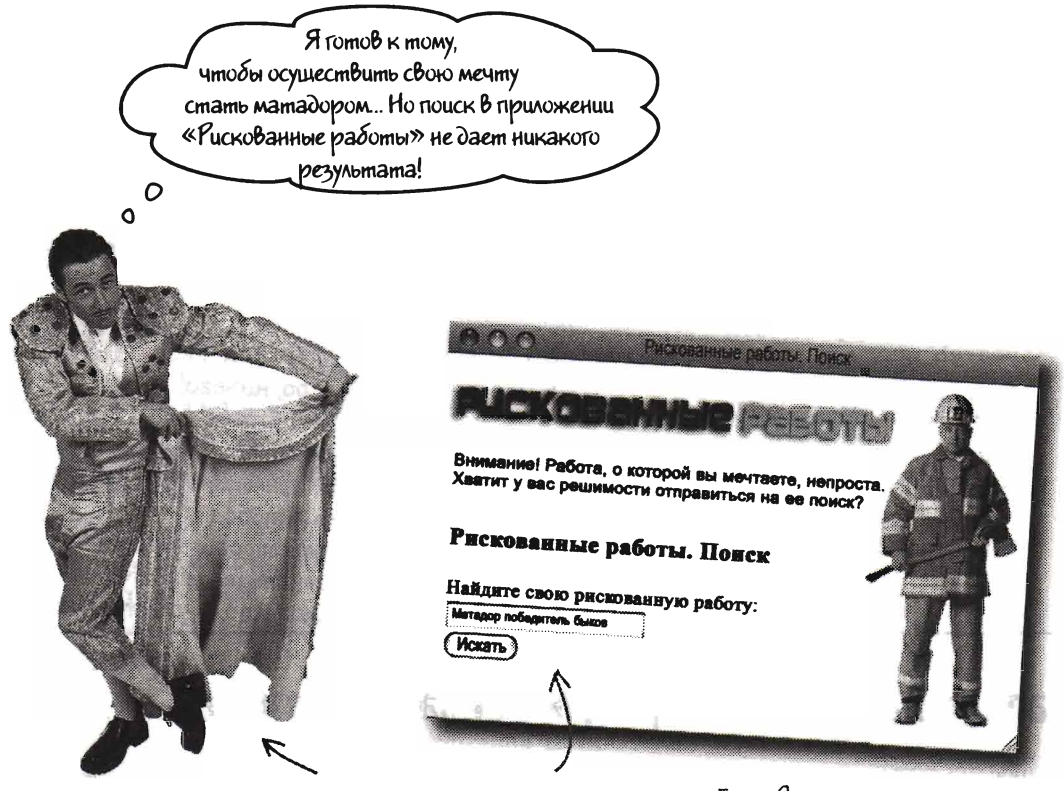
В таблице riskyjobs содержатся наименования работ и их описания вместе с датой опубликования информации о каждой работе.

riskyjobs

job_id	title	description	city	state	zip	company	date_posted
1	Матадор	Беспокойная молочная ферма...	Рутланд	Вермонт	05701	Без ума от молочного хозяйства ...	2008-03-11 10:51:24
2	Папарацци	Студия по фотографированию...	Беверли Хилз	Калифорния	90210	ООО «Звездоискатели»	2008-03-24 10:51:24
3	Дрессировщик акул	Дрессировка акул, чтобы...	Орландо	Флорида	32801	Корпорация «Акульи наживки»	2008-04-28 03:12:45
4	Пожарный	Город Дейтавиль...	Дейтавиль	Огайо	45490	Город Дейтавиль	2008-05-22 12:34:17
5	Измеритель электрического напряжения	Вы будете в...	Дюрхам	Северная Каролина	27701	ООО «Шоковые системы»	2008-06-28 11:16:30
6	Крокодилий дантист	Вы любите животных...	Эверглейдс Сити	Флорида	34139	Алчные рептилии	2008-07-14 10:51:24
7	Ходящий по заварному крему	Нам необходимы люди...	Альбукерке	Нью-Мексико	87101	Пирожковые технологии	2008-07-24 10:54:05
8	Электромонтер по ремонту механических быков	Хэнкс Хэнки Тонк...	Хобокен	Нью-Джерси	07030	Хэнкс Хэнки Тонк	2008-07-27 11:22:28
...							

Каждая запись уникально идентифицирована с помощью первичного ключа.

Покажите результат поиска!



Эрнесто, бесстрашный победитель быков, выглядит растерянным, потому что его поиск в приложении «Рискованные работы» не дает никакого результата.

Время поработать

После того как данные формы будут переданы на сервер для обработки, строка поиска сохранится в переменной `$user_search`, которая добавляется к следующему SQL-запросу на поиск. Напишите, сколько записей будет найдено в таблице `riskyjobs` в результате выполнения поиска по этому запросу.

```
$search_query = "SELECT job_id, title, state, description FROM riskyjobs " .
  "WHERE title = '$user_search'";
$result = mysqli_query($dbc, $search_query);
```

Напишите ваш ответ здесь!

Решение задачи

После того как данные формы будут переданы на сервер для обработки, строка поиска сохранится в переменной `$user_search`, которая добавляется к следующему SQL-запросу на поиск. Напишите, сколько записей будет найдено в таблице `riskyjobs` в результате выполнения поиска по этому запросу.

Если в условном выражении WHERE применен оператор тождества (=), это означает, что две эти строки должны точно соответствовать друг другу (быть тождественно равными).

Эта переменная содержит то, что было введено в поле ввода данных формы поиска.

```
$search_query = "SELECT job_id, title, state, description FROM riskyjobs " .
  "WHERE title = '$user_search'";
$result = mysqli_query($dbc, $search_query);
```

Ноль, зеро, ничего! Проблема в том, что условия сравниваемых данных при выполнении нашего запроса слишком строги: точное соответствие тексту, введенному пользователем.

..0.....

Запрос на поиск не прощает никакой ошибки

Запрос `SELECT` в сценарии «Рискованные работы» составлен в слишком жесткой форме, требующей полного соответствия сравниваемых строк для достижения успеха в поиске. Это является проблемой для успешного поиска, и поэтому пользователям необходимо предоставить возможность вводить данные для поиска, которые необязательно точно соответствуют данным, имеющимся в базе.

Регистр букв условия поиска не имеет значения, так как условное выражение WHERE в MySQL по умолчанию не учитывает регистра клавиатуры.

Давайте вернемся назад, к форме поиска, которую заполнил Эрнесто, в результате чего был составлен запрос на поиск в таблице `riskyjobs` записи со значением колонки `title` «Матадор победитель быков»:

```
SELECT job_id, title, description FROM riskyjobs
WHERE title = 'Матадор победитель быков'
```

Оператор = требует точного соответствия при сравнении значений двух строк.

Видите, в чем проблема? При выполнении этого запроса из таблицы будут выбираться только те записи, для которых колонка `title` содержит буквальную строку текста «Матадор победитель быков». Строка «Матадор» запросу не соответствует, точно так же, как не соответствуют строки «Пожарный» или «Электромонтер по ремонту механических быков». Может, это и хорошо для последних двух строк, но все-таки поиск не производится так, как хотелось бы. И причина не в смешении регистров (поиск в MySQL по умолчанию не учитывает регистра клавиатуры). Дело в том, что, раз мы применяем в условном выражении `WHERE` знак = для сравнения, строка, указанная в запросе для поиска, должна точно соответствовать строке, имеющейся в базе данных.

SQL-запросы могут стать более гибкими при использовании в них оператора LIKE

Что нам в действительности необходимо, так это способ поиска в базе данных строк, у которых соответствует критериям поиска только какая-то их часть, а не все они целиком. SQL предоставляет такую возможность путем использования оператора `LIKE` (подобный), который позволяет более гибко производить операции сравнения в условном выражении `WHERE`. Вы можете рассматривать оператор `LIKE` как значительно смягченный вариант оператора `=`. Посмотрите на следующий запрос, использующий оператор `LIKE` для извлечения записей, в которых слово «канат» встречается в любом месте строки, содержащейся в колонке `title`.

```
SELECT job_id, title, description FROM riskyjobs
WHERE title LIKE '%канат%'
```

Оператор LIKE позволяет вам находить частичное соответствие в случае, когда проверяемые строки полностью не соответствуют друг другу.

Символы % являются групповыми: они соответствуют любым символам в начале и в конце слова.

Использование оператора `LIKE` позволяет значительно проще найти соответствие, особенно если речь идет о поиске соответствия части слова или фразы. Посмотрите на примеры строк, которые соответствуют критериям условного выражения приведенного выше запроса:

Испытатель канатов

Отважный канатоходец

Профессиональный боксер

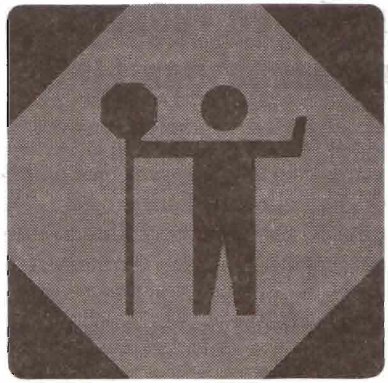
Чокнутые биты

В SQL используется и другой групповой символ, который также применяется вместе с оператором `LIKE`. Это символ подчеркивания (_). Он представляет один любой символ. Рассмотрите следующее выражение с оператором `LIKE`:

```
LIKE '____канат%'
```

Это выражение имеет следующее содержание: «Найти строку "канат" с двумя любыми символами, предшествующими ей, и любым количеством любых символов, следующих за ней». Это будет соответствовать фразе «я канатоходец», но не «отважный канатоходец».

Использование оператора `LIKE` предполагает также использование в поисковой строке групповых символов, которые являются в ней заменителями обычных символов. В SQL групповой символ — знак процента (`%`) — является заменителем группы любых символов любой длины, включая нулевую. Размещение этих групповых символов перед и после поисковой строки, как показано в вышеприведенном запросе, говорит SQL выбрать все строки, в которых где-либо встречается искомая строка независимо от того, сколько символов имеется до и после нее.



Стоп! Задержитесь на минуту, чтобы познакомиться поближе с базой данных «Рискованные работы»... И попробуйте сделать несколько поисков.

Загрузите файл `riskyjobs.sql` для приложения «Рискованные работы» с сайта, расположенного по адресу `www.headfirstlabs.com/books/hfphp`. Этот файл содержит SQL-запросы для создания таблицы `riskyjobs` и занесения в нее данных для примеров.

После того как вы выполните запросы из файла `riskyjobs.sql` в какой-либо инструментальной программе MySQL, попробуйте сделать несколько запросов, имитирующих поиски работы. Ниже показаны примеры, с которых вы можете начать.

```
SELECT * FROM riskyjobs
```

В результате выполнения этого запроса будут извлечены все колонки всех записей из таблицы riskyjobs.

```
SELECT job_id, title, description FROM riskyjobs
WHERE title = 'Матадор — победитель быков'
```

В результате выполнения этого запроса из таблицы riskyjobs будут извлечены идентификаторы работ, наименования и описания для записей, название которых «Матадор — победитель быков» точно.

```
SELECT job_id, title, description FROM riskyjobs
WHERE description LIKE '%животные%'
```

В этом запросе используется оператор LIKE для извлечения записей, у которых в колонке description встречается слово «животные».

Загрузите это!

Весь код для приложения «Рискованные работы» доступен для загрузки на сайте «Лаборатория "Очертя голову"» по адресу:

www.headfirstlabs.com/books/hfphp

Оператор LIKE и магниты

Ниже разбросаны несколько выражений с оператором `LIKE`. Можете ли вы, используя магниты, показать соответствие этих выражений и их результатов? Какие магниты не будут соответствовать ни одному из приведенных здесь выражений?

Некоторым выражениям могут соответствовать несколько результатов одновременно.

`LIKE '%л'`

`LIKE '% для%'`

`LIKE 'ор%'`

`LIKE '%та%'`

`LIKE '%ло_%'`

`LIKE '%ли'`

`LIKE '%Змей заклинатель%'`

Человек-снаряд

Талисман

Тестер корма для животных

Клоун

Матадор

Политик

Прыгун в воду со скалы

Манекен для крэш-тестов

Кошачий пастух

Заклинатель змей

Кантователь коров

Искатель акул

Оператор LIKE и магниты. Решение

Ниже разбросаны несколько выражений с оператором LIKE. Можете ли вы, используя магниты, показать соответствие этих выражений и их результатов? Какие магниты не будут соответствовать ни одному из приведенных здесь выражений?

Этому выражению соответствуют все строки, оканчивающиеся на букву «л».

```
LIKE '%л'
```

Искатель акул

Прыгун в воду со скал

Должны быть пробелы перед и после слова «для».

```
LIKE '% для%'
```

Тестер корма для животных

Манекен для крэш-тестов

В SQL-запросах по умолчанию не учитывается регистр клавиатуры, поэтому слово, начинающееся как со строчной буквы «т», так и с прописной, будет соответствовать этому выражению.

```
LIKE 'та%'
```

Танцор

Талисман

```
LIKE '%ор %'
```

Этому выражению соответствуют все строки, оканчивающиеся на буквы «ор».

Танцор

Матадор

Только два символа после букв «ло».

```
LIKE '%ло_'
```

Клоун

Этому выражению соответствует все строки, включающие буквы «ли».

```
LIKE '%ли%'
```

Политик

Талисман

Заклинатель змей

Эти фразы не соответствуют ни одному выражению.

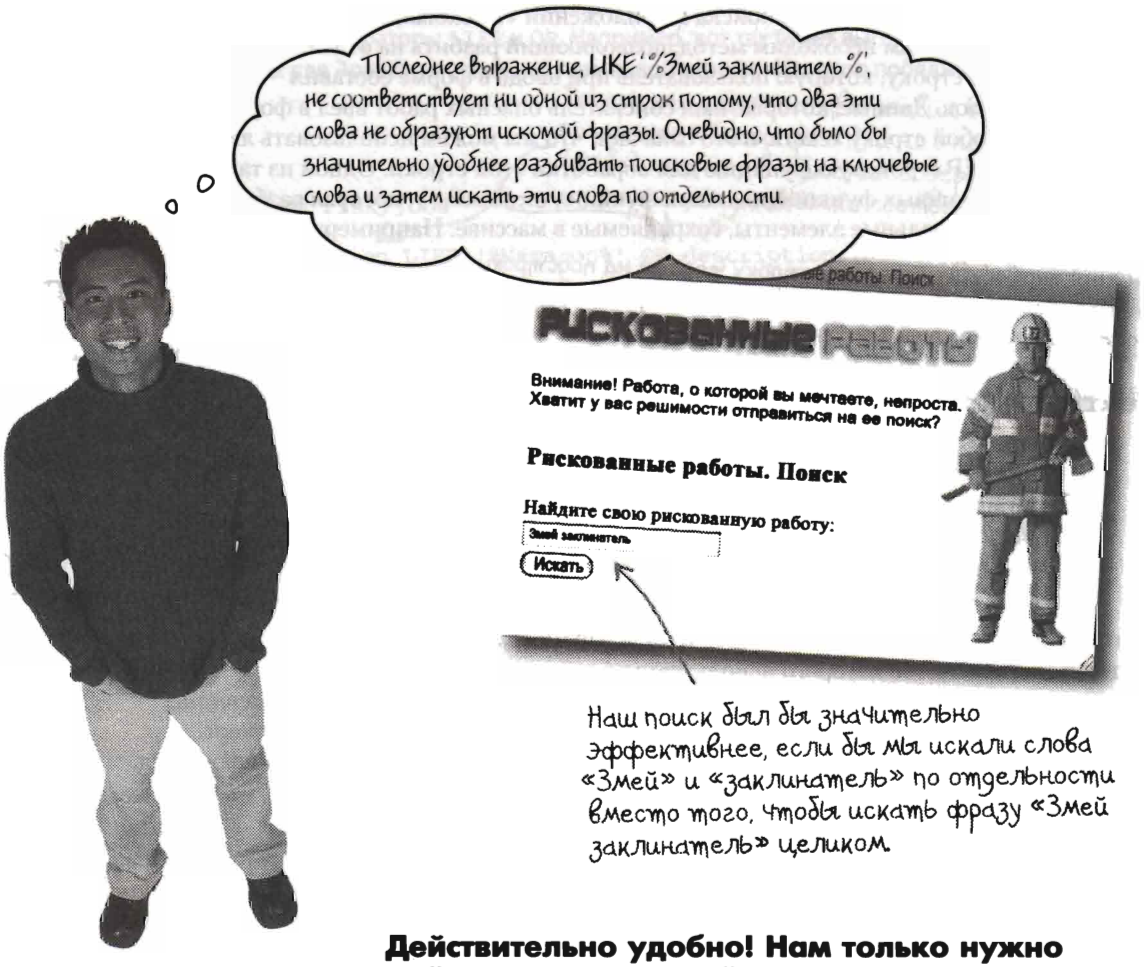
Человек-снаряд

Прыгун в воду со скал

Кошачий пастух

Этому выражению не соответствует ни одна строка.

```
LIKE '%Змей заклинатель%'
```



Действительно удобно! Нам только нужно найти метод, который позволит в процессе поиска определять соответствие отдельным словам, а не всей фразе целиком.

Использование для определения соответствия фразы, которую пользователь ввел в поле ввода данных поисковой формы, не всегда приводит к благоприятному результату. Поиск был бы значительно эффективнее, если бы мы проверяли соответствие отдельным словам поисковой фразы, а не всей фразе целиком. Но как мы сможем это сделать? Мы могли бы сохранить все слова поисковой фразы в массиве и затем подправить запрос `SELECT` так, чтобы поиск производился по отдельным словам.

Разбейте строку на отдельные слова

Для того чтобы сделать функцию поиска в приложении «Рискованные работы» более эффективной, нам необходим метод, позволяющий разбить на отдельные слова поисковую строку, которую пользователь при вводе в форме составил из нескольких слов. Данные, которые наш соискатель опасных работ ввел в форму, представляют собой строку текста, и это означает, что мы можем использовать любую встроенную в PHP строковую функцию для обработки этой строки. Одной из таких исключительно мощных функций является функция `explode()`, которая разбивает строку текста на отдельные элементы, сохраняемые в массиве. Например:

Функция explode() разбивает строку на массив подстрок.

Функция explode() разрезает строку текста на подстроки, основываясь на значении разделителя, который отделяет подстроки друг от друга внутри строки.

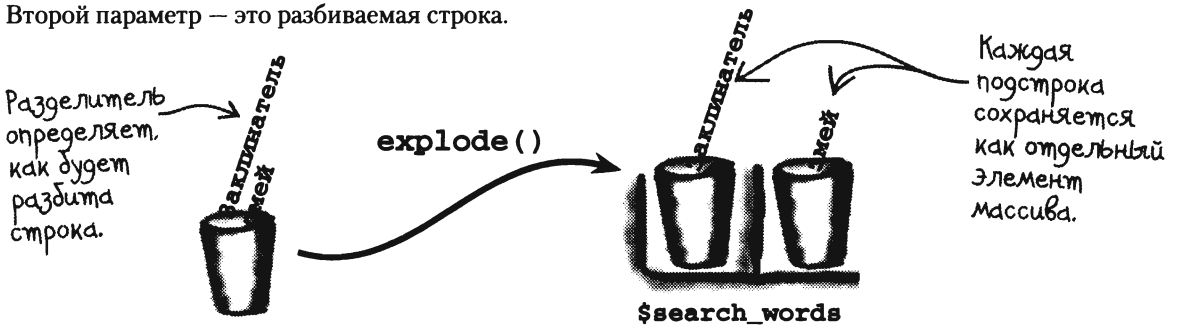
Этот параметр — строка текста, которую мы хотим разбить на подстроки.

```
$search_words = explode(' ', 'Змей заклинатель');
```

Переменная $search_words теперь содержит массив поисковых подстрок, которые мы передадим SQL-запросу.

Этот параметр сообщает функции explode(), что в качестве разделителя подстрок внутри строки в этом случае используется символ пробела. В качестве разделителя может быть использован как один символ, так и несколько.

Функции `explode()` необходимо передать два параметра. Первым является **разделитель**, в качестве которого используется символ или группа символов, указывающие, **в каких местах** должна быть разбита строка. Мы используем в качестве разделителя символ пробела, и это означает, что строка будет разбита на подстроки во всех местах, где встречается пробел. Сам разделитель в состав подстрок не включается. Второй параметр — это разбиваемая строка.



Использование массива поисковых подстрок в приложении «Рискованные работы» требует включения в него дополнительного кода, прежде чем мы сможем начать запрашивать данные в базе «Рискованные работы». Теперь, если кто-нибудь введет фразу «Заклинатель змей» в поле ввода формы поиска, в результате выполнения этого кода строка будет разбита на два слова, и каждое из них будет сохранено в массиве `$search_words`.

```
$user_search = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
```

Каждое слово строки, содержащейся в переменной $user_search, функция explode() сохраняет в массиве $search_words.

УПРАЖНЕНИЕ

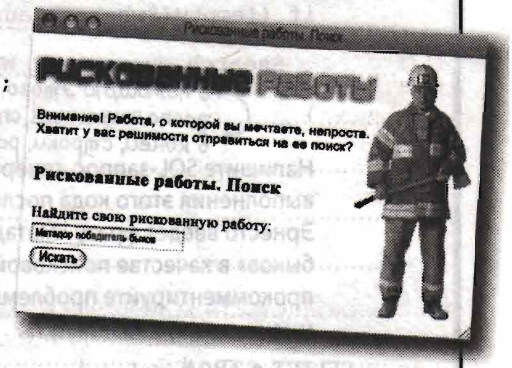
Для того чтобы включить в процесс поиска отдельные слова, полученные в результате разбиения поисковой строки на подстроки, мы должны добавить их к запросу SELECT, используя операторы LIKE и OR. Например, вот так будет выглядеть запрос для поиска работы для Эрнесто с помощью поисковой строки «Матадор победитель быков»:

Теперь мы ищем соответствия в колонке «описание» (description), а не «наименование» (title), так как в описании содержится больше информации для поиска соответствия.

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%Матадор%' OR description LIKE '%победитель%' OR
  description LIKE '%быков%'
```

А теперь предположим, что мы использовали следующий PHP-код в попытке включить в этот запрос данные, введенные пользователем в форму поиска приложения «Рискованные работы»:

```
$search_query = "SELECT * FROM riskyjobs";
$where_clause = '';
$user_search  = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
foreach ($search_words as $word) {
  $where_clause .= " description LIKE '%$word%' OR ";
}

if (!empty($where_clause)) {
  $search_query .= " WHERE $where_clause";
}
```



Напишите SQL-запрос, генерируемый в результате выполнения этого кода после того, как Эрнесто введет фразу «Матадор победитель быков» в качестве поисковой строки, а затем прокомментируйте проблемы, которые, по вашему мнению, могут возникнуть.

.....

.....

.....

Для того чтобы включить в процесс поиска отдельные слова, полученные в результате разбиения поисковой строки на подстроки, мы должны добавить их к запросу SELECT, используя операторы LIKE и OR. Например, вот так будет выглядеть запрос для поиска работы для Эрнесто с помощью поисковой строки «Матадор победитель быков»:

Теперь мы ищем соответствия в колонке «описание» (description), а не «наименование» (title), так как в описании содержится больше информации для поиска соответствия.

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%Матадор%' OR description LIKE '%победитель%' OR
  description LIKE '%быков%'
```

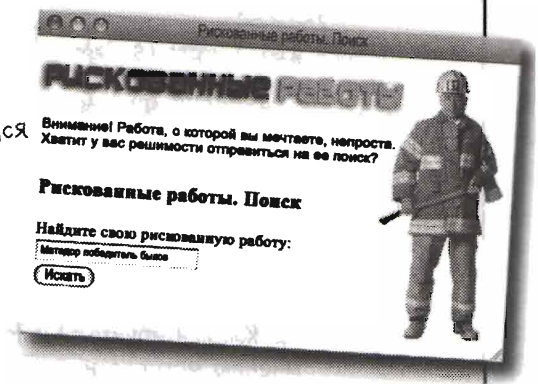
А теперь предположим, что мы использовали следующий PHP-код в попытке включить в этот запрос данные, введенные пользователем в форму поиска приложения «Рискованные работы»:

```
$search_query = "SELECT * FROM riskyjobs";
$where_clause = '';
$user_search  = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
foreach ($search_words as $word) {
  $where_clause .= " description LIKE '%$word%' OR ";
}
if (!empty($where_clause)) {
  $search_query .= " WHERE $where_clause";
}
```

Каждое выражение с оператором LIKE заканчивается логическим оператором OR для того, чтобы соединить его со следующим аналогичным выражением, что работает замечательно, кроме последнего выражения.

Убедитесь в том, что условное выражение WHERE непустое, прежде чем добавлять его к поисковому запросу.

С помощью этого оператора строка, расположенная справа от него, добавляется в конец строки, расположенной слева.



Напишите SQL-запрос, генерируемый в результате выполнения этого кода после того, как Эрнесто введет фразу «Матадор победитель быков» в качестве поисковой строки, а затем прокомментируйте проблемы, которые, по вашему мнению, могут возникнуть.

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%Матадор%' OR description LIKE '%победитель%' OR
  description LIKE '%быков%' OR
```

В конце запроса появился логический оператор OR, что приведет к ошибке при выполнении запроса!

Функция implode() создает строку из подстрок

Все, что нам необходимо, — это вставить логические операторы OR между выражениями с операторами LIKE в нашем условном выражении WHERE, **но только не в его конце**. Так как же это может быть сделано? Что если при каждом добавлении в цикле нового выражения с оператором LIKE проверять, не достигнут ли конец условного выражения, и добавлять логический оператор OR только в том случае, если конец еще не достигнут. Это будет работать, но выглядит довольно некрасиво. Значительно более выразительное решение заключается в использовании функции implode(), которая выполняет задачу, обратную той, что выполняет функция explode(). Функция implode() создает строку из массива подстрок, переданного ей в качестве аргумента.

```
$where_clause = implode(' OR ', $where_list);
```

Это разделитель, который вставляется между подстроками, когда они объединяются в одну общую строку.

Функция implode() возвращает одну строку.

Это должен быть массив подстрок, которые вы хотите объединить в одну общую строку.

Но как это поможет нам решить проблему висящего в конце запроса логического оператора OR? Дело в том, что функция позволяет вам указать разделитель, который будет вставлен между подстроками в процессе создания из них окончательной строки. Если мы используем в качестве разделителя ' OR ', то сможем создать строку условного выражения WHERE, в котором логические операторы OR будут только между выражениями с оператором LIKE.

Время поработать

Перепишите PHP-код, ответственный за генерацию запроса поиска рискованной работы, чтобы решить проблему висящего в конце запроса логического оператора OR с помощью функции implode().

Решение задачи

Перепишите PHP-код, ответственный за генерацию запроса поиска рискованной работы, чтобы решить проблему висящего в конце запроса логического оператора OR с помощью функции `implode()`.

```
$search_query = "SELECT * FROM riskyjobs";
$where_list = array();
$user_search = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
foreach ($search_words as $word) {
  $where_list[] = "description LIKE '%$word%'";
}
$where_clause = implode(' OR ', $where_list);

if (!empty($where_clause)) {
  $search_query .= " WHERE $where_clause";
}
```

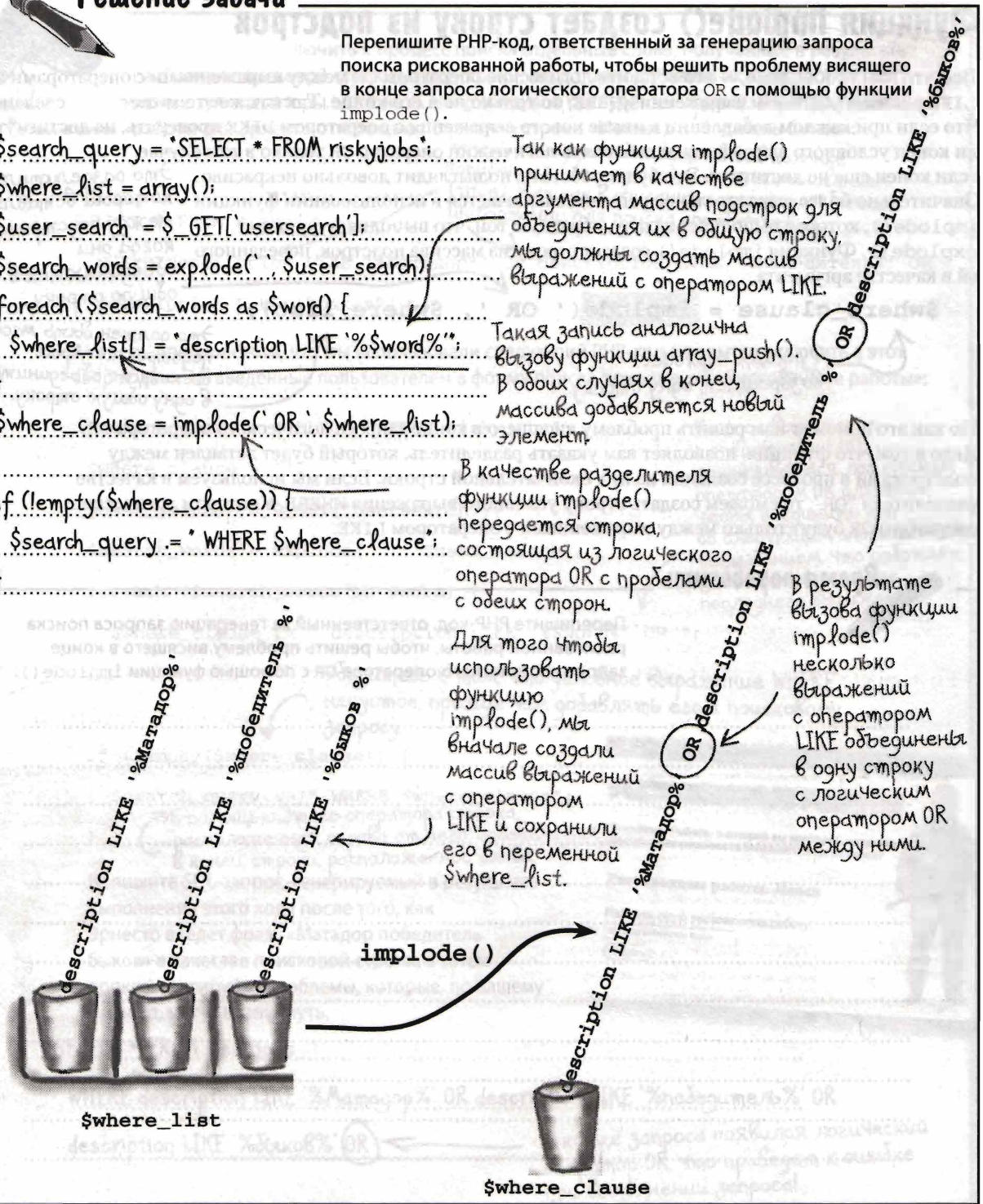
Так как функция implode() принимает в качестве аргумента массив подстрок для объединения их в общую строку, мы должны создать массив выражений с оператором LIKE.

Такая запись аналогична вызову функции array_push(). В обоих случаях в конец массива добавляется новый элемент.

В качестве разделителя функции implode() передается строка, состоящая из логического оператора OR с пробелами с обеих сторон.

Для того чтобы использовать функцию implode(), мы вначале создали массив выражений с оператором LIKE и сохранили его в переменной $where_list.

В результате вызова функции implode() несколько выражений с оператором LIKE объединены в одну строку с логическим оператором OR между ними.

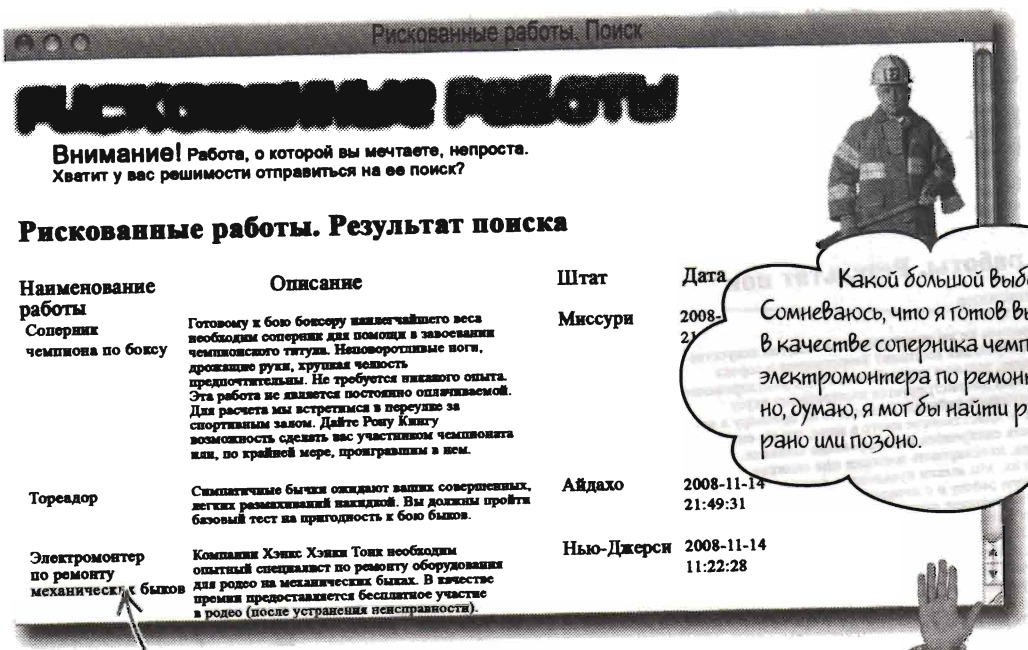


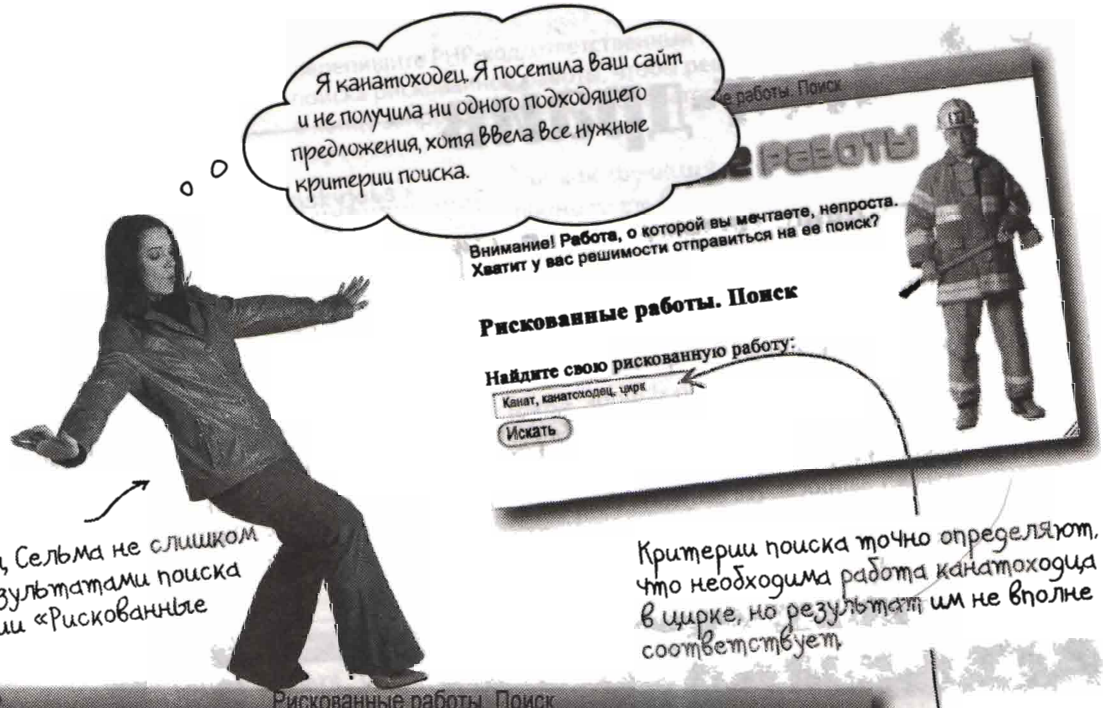
Тест-драйв

Испытайте систему поиска приложения «Рискованные работы».

Загрузите приложение «Рискованные работы» с сайта по адресу `www.headfirstlabs.com/books/hfphp`. Сценарий `search.php` содержит код, генерирующий запрос, который мы только что разбирали, и используется для обработки данных, введенных в форму поиска `search.html`.

Загрузите сценарий и остальные файлы приложения «Рискованные работы» на ваш веб-сервер и откройте форму поиска `search.html` в браузере. Попробуйте задать несколько различных вариантов критериев поиска, чтобы увидеть, как код сценария создает запросы. Не забудьте попробовать поисковую строку «Матадор победитель быков», которая является хорошим критерием поиска для тестирования нового кода с использованием функции `implode()`.





Канатоходец Сельма не слишком довольна результатами поиска в приложении «Рискованные работы».

Критерии поиска точно определяют, что необходима работа канатоходца в цирке, но результат им не вполне соответствует.

Рискованные работы. Поиск

РИСКОВАННЫЕ РАБОТЫ

Внимание! Работа, о которой вы мечтаете, непроста. Хватит у вас решимости отправиться на ее поиск?

Рискованные работы. Результат поиска

Наименование работы	Описание	Штат	Дата
Мастер по жонглированию кошками	Являетесь ли вы специалистом в забытом искусстве жонглирования кошками? Запрещенное в сорока странах, только в цирке Джима Рунза жонглирование кошками демонстрируется изощренному вкусу современной публики. Приходите на премьеру в наш цирк — единственное место в мире, где вы сможете увидеть синхронное жонглирование кошками. Это правда, жонглировать кошками еще сложнее, чем пасти их. Мы можем предложить вам как одну, так и другую работу и с нетерпением ожидаем того момента, когда вы сможете присоединиться к нашей команде. Будьте готовы, пожалуйста, к тому, чтобы быть подвергнутыми нескольким тщательным испытаниям ваших способностей иметь дело с представителями кошачьих. Только самые лучшие будут допущены к участию в программе «Мастер по жонглированию кошками».	Аризона	2008-11-14 21:13:35
Испытатель канатов	Если для вас нет лучшего времяпрепровождения, чем часами висеть на канате на большой высоте, тогда, скорее всего, эта работа для вас. Каждый из наших испытателей проходит через жесткий тест из 43 пунктов. При этом на завершающем этапе ему предлагается провисеть в течение значительного времени. Это можете быть вы! Мы предоставляем защитную сетку, но вы должны иметь свой собственный шлем и перчатки. Здесь, на предприятии «Купол», Монтана, мы можем предложить вам отличные условия для работы, начиная с того, что вы можете приносить сюда своих питомцев, и заканчивая строгой одеждой по пятницам. Нам также необходимо 3 справки, включающие заверенную информацию о максимальном времени, проведенном вами в подвешенном состоянии, и количестве падений. Мы больше, чем просто цирк.	Монтана	2008-11-14 21:17:16

Неправильно введены критерии поиска или работа для канатоходцев действительно отсутствует?

Время поработать

Напишите SQL-запрос, который будет создан после того, как Сельма введет данные «канат, канатоходец, цирк» в строку ввода данных поисковой формы. Прокомментируйте проблемы, которые могут возникнуть при выполнении этого запроса.

.....

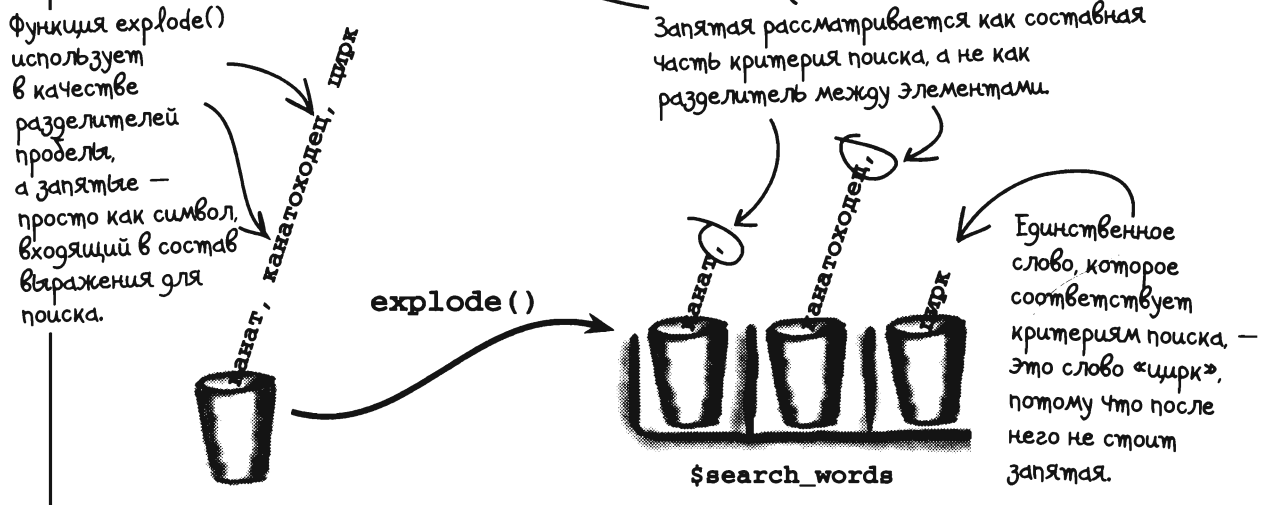
.....

.....

Решение задачи

Напишите SQL-запрос, который будет создан после того, как Сельма введет данные «канат, канатоходец, цирк» в строку ввода данных поисковой формы. Прокомментируйте проблемы, которые могут возникнуть при выполнении этого запроса.

```
SELECT * FROM riskyjobs
WHERE description LIKE '%канат,%' OR description LIKE %канатоходец,%' OR
description LIKE '%цирк%'
```



Не вижу никаких проблем. Нужно просто вызвать функцию explode() дважды. Первый раз — чтобы избавиться от пробелов, второй — от запятых.

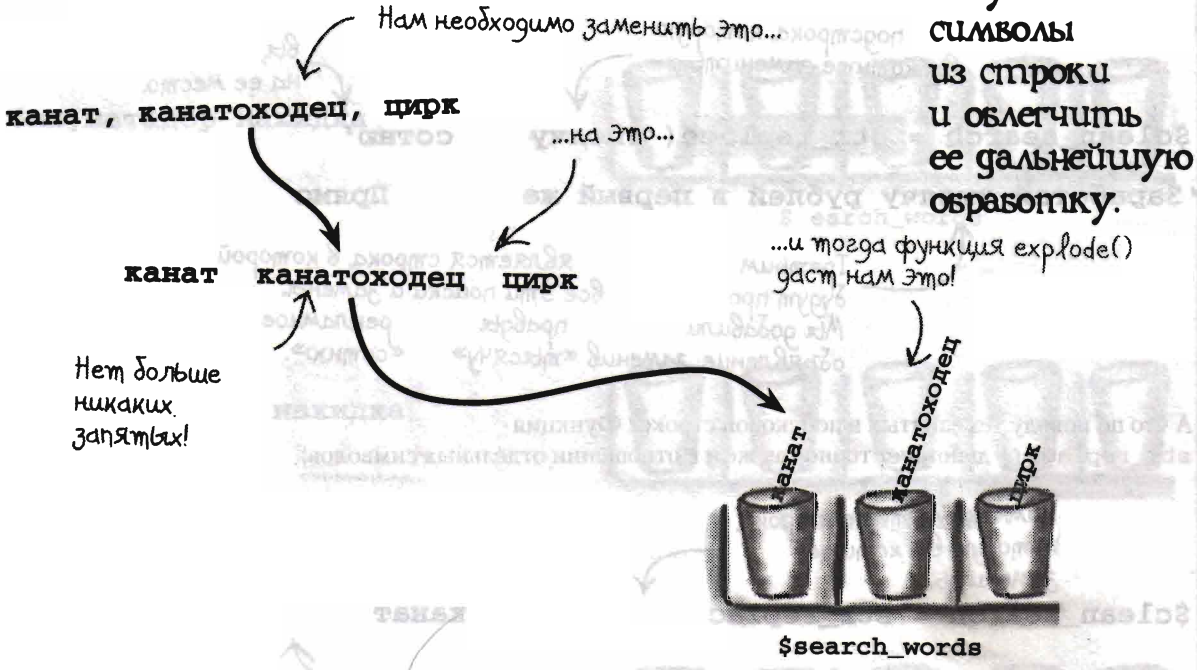
Функция explode() позволяет вам разбить единую строку на подстроки, но в данном случае мы уже имеем дело с подстроками.

В результате первого вызова функции `explode()` образуется множество строк, сохраненных в массиве, то есть мы уже не имеем дело с единой строкой, которую можно было бы обрабатывать так же, как и первоначальную. Попытка вызывать функцию `explode()` для каждой из подстрок, скорее всего, просто приведет к появлению дополнительных проблем. Вместо того чтобы пытаться решать проблемы путем многократного вызова функции `explode()`, нам необходимо произвести **предварительную обработку** поисковой строки так, чтобы в ней остался только **один разделитель**, прежде чем вызывать функцию `explode()`. Тогда эта функция сделает то, для чего она предназначена, — разобьет строку на подстроки в соответствии с заданным разделителем.

Предварительная обработка поисковой строки

Мы хотим передать строку функции `explode()` так, чтобы она разбила ее на подстроки без проблем за один раз. Как нам это осуществить? Нужно сделать так, чтобы функция `explode()` имела дело только с одним разделителем, таким как символ пробела. Это означает, что мы должны так **предварительно обработать** поисковую строку, чтобы каждая подстрока отделялась от своих соседей с помощью только одного символа пробела, даже если пользователь ввел еще и запятую.

Предварительная обработка позволяет нам удалить ненужные символы из строки и облегчить ее дальнейшую обработку.



Не бывает глупых вопросов

В: Можем ли мы использовать более одного символа в качестве разделителя при разбиении строки на подстроки с помощью функции expolode()?

О: Да, вы можете указать несколько символов, которые будут играть роль разделителя, но это не то же самое, что указывать разные разделители, и не решает нашей проблемы. Если бы мы вызвали функцию `expolode()` в таком виде:

```
expolode(', ' $user_search),
```

чтобы разбить нашу строку на подстроки, она бы успешно решила задачу в случае ввода пользователем поисковой строки «канат, канатоходец, цирк». Но она не стала бы разбивать строку «канат канатоходец цирк» и оставила бы нас с исходным вариантом строки. Ничего хорошего.

В: Можем ли мы просто удалить запятые вместо того, чтобы превращать их в символы пробелов?

О: Это будет работать только при условии, что пользователь будет разделять свои подстроки с помощью как запятых, так и символов пробела, на что мы рассчитывать не можем. Если мы удалим запятые, то рискуем превратить вполне вероятный вариант строки «канат,канатоходец,цирк» в «канаткanатоходеццирк», чему, скорее всего, в базе данных приложения «Рискованные работы» соответствия не найдется.

Замените ненужные для поиска символы

Предварительная обработка поисковой строки в приложении «Рискованные работы» очень похожа на процесс поиска и замены в текстовом редакторе. В нашем случае необходимо найти все запятые и заменить их на символы пробела. PHP-функция `str_replace()` позволяет вам провести именно эту операцию. Ее необходимо вызвать, передав три аргумента: текст, который необходимо найти, текст, на который его нужно заменить, и строку, в которой нужно произвести все эти поиски и замены. Ниже приведен пример использования этой функции:

Это подстрока, которую вы хотите заменить...

...а это подстрока, которую вы хотите вставить на ее место.

```
$clean_search = str_replace('тысячу', 'сотню',
'Заработай тысячу рублей в первый же день. Прямо сейчас!');
```

Третьим аргументом является строка, в которой будут произведены все эти поиски и замены. Мы добавили немного правды в это рекламное объявление, заменив «тысячу» на «сотню».

А что по поводу тех запятых в поисковой строке? Функция `str_replace()` действует точно так же и в отношении отдельных символов:

Помните? Это подстрока, которую вы хотите заменить...

...а это подстрока, которую вы хотите вставить на ее место.

```
$clean_search = str_replace(',', ' ', 'канат, канатоходец, цирк');
```

Везде, где в этой строке встречается запятая, она будет заменена пробелом.

После вызова этой функции переменной `$clean_search` будет присвоено значение `канат канатоходец цирк`.

Вы не замечаете ничего подозрительного в результатах вызова функции `str_replace()`? Как вы думаете, замена запятых на символы пробелов будет работать так, как мы этого хотим?

УПРАЖНЕНИЕ

Исходя из кода, приведенного ниже, покажите, как будет выглядеть массив `$search_words` для каждой из поисковых строк. Покажите значение каждого из соответствующих элементов массива и зачеркните лишние элементы, если массив окажется короче.

```
$clean_search = str_replace(',', ' ', $user_search);
$search_words = explode(' ', $clean_search);
```

бык,матадор накидка

$search_words

3 пробела!

бык матадор накидка

$search_words

бык , матадор накидка

$search_words

2 пробела!

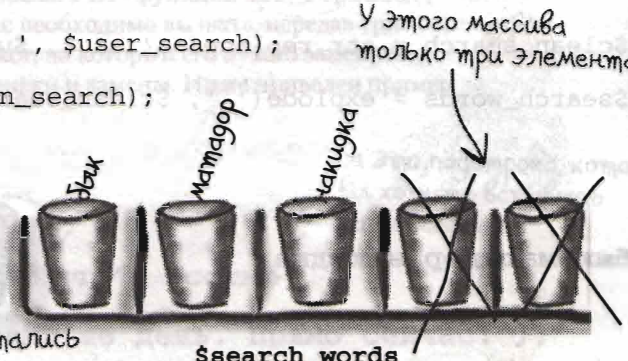
бык,матадор, накидка

$search_words

Исходя из кода, приведенного ниже, покажите, как будет выглядеть массив `$search_words` для каждой из поисковых строк. Покажите значение каждого из соответствующих элементов массива и зачеркните лишние элементы, если массив окажется короче.

```
$clean_search = str_replace(',', ' ', $user_search);
$search_words = explode(' ', $clean_search);
```

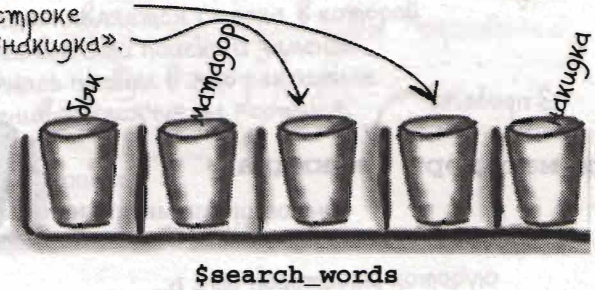
бык,матадор накидка



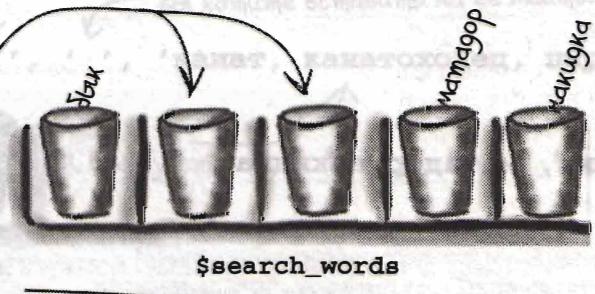
бык матадор накидка

Эти два элемента массива остались пустыми из-за тех двух лишних символов пробела в поисковой строке между словами «матадор» и «накидка».

3 символа пробела!



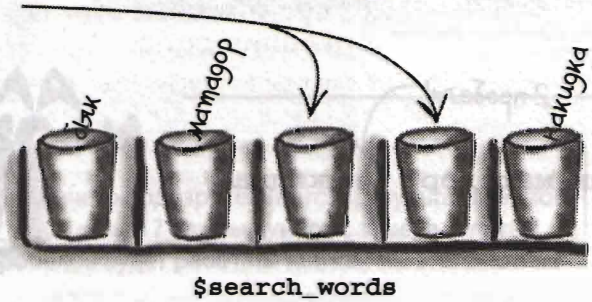
бык , матадор накидка



бык,матадор, накидка

Опять два пустых элемента в массиве из-за того, что запятая заменена символом пробела.

2 символа пробела!

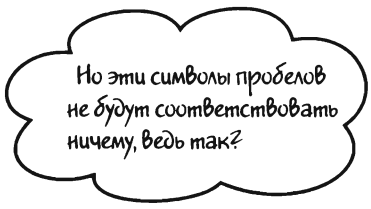


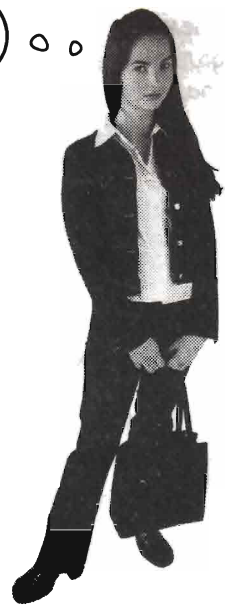
К сожалению, нет. В результате предварительной обработки мы избавились от ненужных символов, но не получили в конечном счете массива, содержащего элементы, пригодные для поиска.

Помните, наша цель состояла в том, чтобы получить строку, в которой каждая подстрока отделена от своих соседей одним и тем же разделителем — символом пробела? Взгляните еще раз на то, что получилось в последних трех случаях на предыдущей странице. Некоторые из элементов массива `$search_words` пусты. Если мы попытаемся создать условное выражение для поискового запроса с использованием такого массива, содержащего пустые элементы, мы получим что-то вроде этого:

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%бык%' OR
  description LIKE '%матадор%' OR
  description LIKE '% %' OR
  description LIKE '% %' OR
  description LIKE '%накидка%'
```

Эти символы пробела будут соответствовать каждому символу пробела в описании работы. Мы стоим перед серьезной проблемой.





Нет! Их наличие в условном выражении приведет к тому, что этому выражению будут соответствовать практически все описания работ.

Если в описании работы встречается хоть один символ пробела, что практически неизбежно, такое описание будет соответствовать условиям поиска, и запись, его содержащая, попадет в результат запроса. Таким образом, из таблицы `riskyjobs` будут извлечены практически все записи. Нам необходимо избавиться от этих пустых элементов массива перед тем, как создавать SQL-запрос, чтобы сделать сценарий поиска полезным.

Запросу необходимо условие поиска с действительными критериями

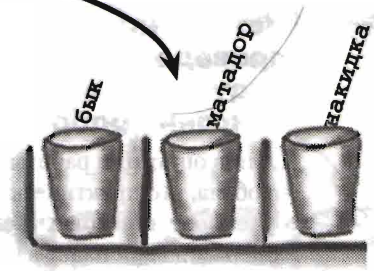
Положительная сторона этой проблемы заключается в том, что не составляет большого труда избавиться от пустых элементов массива, перед тем как создавать массив. Нам необходимо создать новый массив, в котором будут содержаться только действительные критерии поиска. Поэтому мы скопируем все непустые элементы первого массива во второй массив, который затем и используем для создания запроса `SELECT`.

Для того чтобы создать новый массив, мы можем пройти в цикле `foreach` все элементы оригинального массива и, используя управляющую конструкцию `if`, найти все непустые его элементы. Как только такой элемент будет найден, он должен быть скопирован в новый массив. Вот так примерно должен выглядеть этот процесс:

Это оригинальный массив, который наряду с элементами с действительными критериями поиска содержит также и пустые элементы.

Эти два пустых элемента массива должны быть удалены!

Нам необходимо добавить в сценарий код, с помощью которого будет создаваться новый массив, содержащий только непустые критерии поиска.



Новый массив короче, потому что он содержит только элементы с действительными критериями поиска и не включает никаких пустых элементов!

Копирование непустых элементов массива в новый массив

А теперь давайте посмотрим на код, с помощью которого непустые элементы из нашего массива `$search_words` будут скопированы в массив `$final_search_words`.

```
$search_query = "SELECT * FROM riskyjobs";

// Извлечение критериев поиска в массив
$clean_search = str_replace(',', ' ', $user_search);
$search_words = explode(' ', $clean_search);
$final_search_words = array();
if (count($search_words) > 0) {
  foreach ($search_words as $word) {
    if (!empty($word)) {
      $final_search_words[] = $word;
    }
  }
}
```

Здесь ничего нового: заменяем запятые символами пробела с помощью функции str_replace().

Проходим в цикле по каждому элементу массива $search_words. Каждый непустой элемент добавляем в массив $final_search_words.

После проверки того, что в массиве `$search_words` есть хотя бы один критерий поиска, проходим в цикле `foreach` все элементы массива в поиске непустых элементов. Как только такой элемент будет найден, с помощью оператора `[]` он добавляется в конец массива `$final_search_words`. Вот так создается новый массив.

Что же дальше? Затем мы создаем запрос `SELECT` точно так же, как и прежде, с той лишь разницей, что на этот раз мы используем массив `$final_search_words` вместо массива `$search_words`:

```
// Создание условного выражения с использованием всех
критериев поиска
if (count($final_search_words) > 0) {
  foreach($final_search_words as $word) {
    $where_list[] = "description LIKE '%$word%'";
  }
}
$where_clause = implode(' OR ', $where_list);

// Add the keyword WHERE clause to the search query
if (!empty($where_clause)) {
  $search_query .= " WHERE $where_clause";
}
```

Этот код точно такой же, как тот, который мы использовали ранее, но на этот раз в нем используется массив $final_search_words, в котором нет пустых элементов.

В результате выполнения этого кода мы получаем поисковый запрос, в котором больше нет пустых элементов. Ниже показан новый запрос для поиска записей с описанием работы, соответствующим поисковой строке «бык, матадор, накидка»:

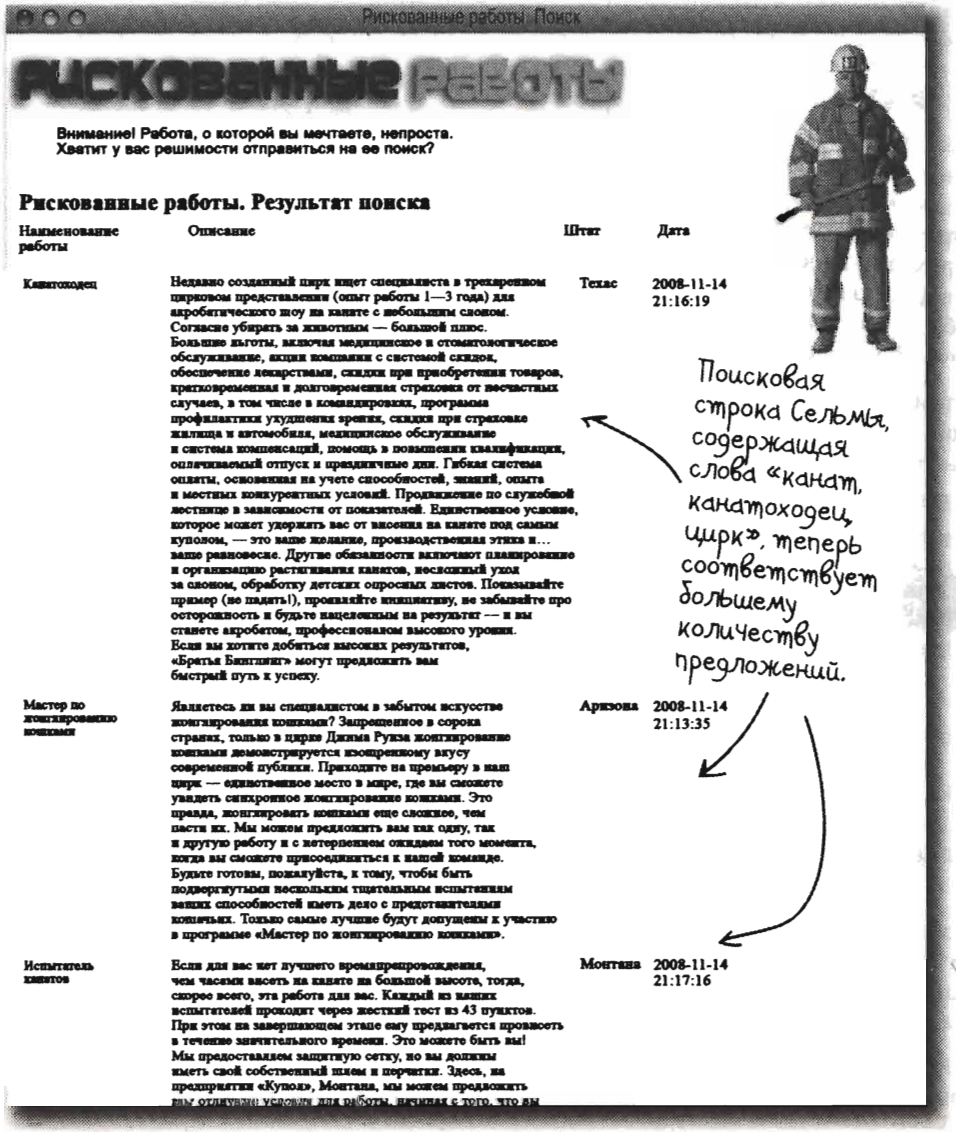
```
SELECT * FROM riskyjobs
   WHERE description LIKE '%бык%' OR
   description LIKE '%матадор%' OR
   description LIKE '%накидка%'
```

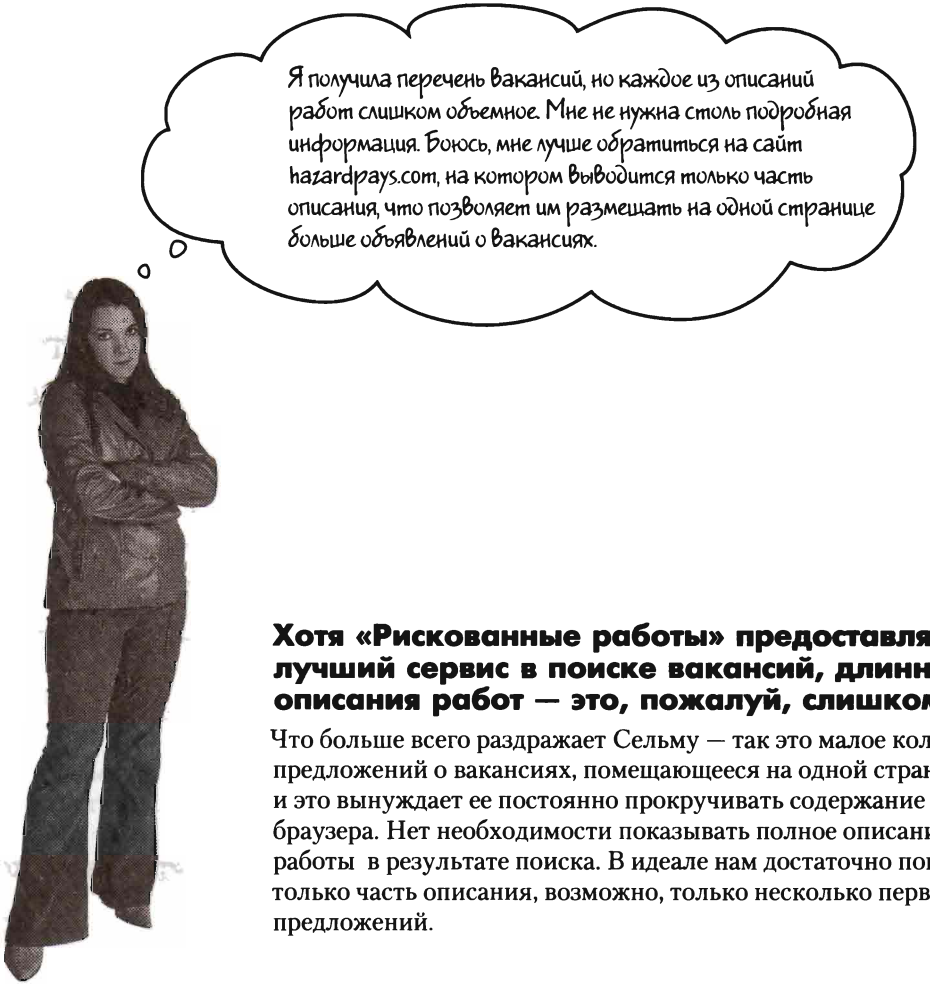
Даст этот код поиска нашим пользователям то, чего они ждут от него?

Тест-драйв

Обновите сценарий поиска так, чтобы в нем использовалась предварительная обработка поисковой строки, введенной пользователем.

Обновите сценарий `search.php` так, чтобы в нем использовались функции `explode()` и `implode()` для предварительной обработки поисковой строки, введенной пользователем, и создавался более надежный запрос SELECT. Затем загрузите сценарий на ваш веб-сервер и попробуйте сделать несколько вариантов поиска.





Хотя «Рискованные работы» предоставляют лучший сервис в поиске вакансий, длинные описания работ — это, пожалуй, слишком.

Что больше всего раздражает Сельму — так это малое количество предложений о вакансиях, помещающееся на одной странице, и это вынуждает ее постоянно прокручивать содержание в окне браузера. Нет необходимости показывать полное описание каждой работы в результате поиска. В идеале нам достаточно показать только часть описания, возможно, только несколько первых предложений.

Напишите, как, по вашему мнению, мы могли бы сократить описания работ, чтобы они не занимали так много места в результатах поиска:

Иногда вам достаточно только части строки

Так как объем описания работ в базе данных приложения «Рискованные работы» различен для различных работ, при этом некоторые из описаний слишком длинны, мы могли бы сделать результаты поиска удобнее для просмотра, укоротив эти описания до приемлемого размера. А чтобы не вводить пользователя в заблуждение по поводу истинной длины этих описаний, мы могли бы поставить многоточие в конце каждого из них, давая пользователю понять, что он видит только часть описания работы.

PHP-функция substr() позволяет вам извлечь из строки ее часть.

Функция PHP `substr()` предназначена для извлечения из строки ее части. Вы передаете этой функции оригинальную строку и два целых числа. Первое число обозначает позицию, с которой вы хотите начать извлечение текста, второе — это длина извлекаемой подстроки в символах. Ниже приведен синтаксис:

```
substr($string, start, length)
```

Это оригинальная строка, из которой мы хотим извлечь подстроку.

Это обозначает позицию в строке, с которой должно начаться извлечение подстроки...

...а это означает количество извлекаемых символов.

Когда речь идет о функции `substr()`, вы можете рассматривать строку как массив, в котором каждый элемент содержит по одному символу строки. Рассмотрим следующую строку:

```
$job_desc = 'Являетесь ли вы специалистом в забытом искусстве жонглирования кошками? ';
```

Так же как в обычном массиве, где на каждый элемент указывает его индекс, каждый символ строки имеет свой индекс, начиная с нуля и заканчивая номером последнего символа в строке.

```
Являетесь ли вы специалистом в забытом искусстве жонглирования кошками?
0 1 2 3 4 5 6 7 8 9...                                          ... 50 51 52
```

Мы можем использовать эти индексы в функции `substr()` для того, чтобы извлечь часть строки:

Извлечь 2 символа, начиная с позиции 13.

```
substr($job_desc, 4, 3)   → вы
```

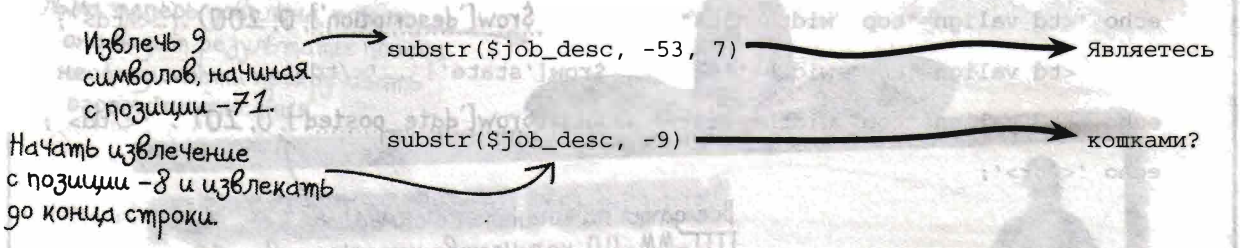
Начать извлечение с позиции 67 и, так как мы не передали значение третьего аргумента, извлекать до конца строки.

```
substr($job_desc, 49)     → ами?
substr($job_desc, 0, 3)   → Являетесь
substr($job_desc, 0, 9)   → Являетесь ли вы
```

Извлечение подстрок с другого конца

Возможности функции `substr()` не ограничиваются извлечением подстрок с начала строки. Вы можете также извлекать символы, начиная с конца строки. Извлечение происходит слева направо, только для указания начального пункта извлечения вы используете отрицательные значения индекса.

Ниже показаны примеры:



Время поработать

Ниже приведен PHP-код, с помощью которого генерируется таблица HTML с результатами поиска опасных работ. Добавьте пропущенный код, задачей которого является ограничение описания работ первой сотней символов, а также отображение в колонке `Дата` только года, месяца и числа.

```
echo '<table border="0" cellpadding="2">';
echo '<td>Наименование работы</td><td>Описание</td><td>Штат</td><td>Дата</td>';
while ($row = mysqli_fetch_array($result)) {
  echo '<tr class="results">';
  echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
  echo '<td valign="top" width="50%">' . ..... . '</td>';
  echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
  echo '<td valign="top" width="20%">' . ..... . '</td>';
  echo '</tr>';
}
echo '</table>';
```

Решение задачи

Ниже приведен PHP-код, с помощью которого генерируется таблица HTML с результатами поиска опасных работ. Добавьте пропущенный код, задачей которого является ограничение описания работ первой сотней символов, а также отображение в колонке Дата только года, месяца и числа.

```
echo '<table border="0" cellpadding="2">';

echo '<td>Наименование работы</td><td>Описание</td><td>Штат</td><td>Дата</td>';

while ($row = mysqli_fetch_array($result)) {
  echo '<tr class="results">';
  echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
  echo '<td valign="top" width="50%">' . substr($row['description'], 0, 100) . '...</td>';
  echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
  echo '<td valign="top" width="20%">' . substr($row['date_posted'], 0, 10) . '</td>';
  echo '</tr>';
}
echo '</table>';
```

Добавляем многоточие в конце строки, чтобы дать понять, что это только часть описания.

Все даты начинаются с символов ГГГГ-ММ-ДД, количество которых равно 10.

Чокнутые биты

Есть возможность ограничить длину строки описания работы путем использования вместо PHP-функции `substr()` функции MySQL под названием `SUBSTRING()`, которая принимает такие же аргументы, что и функция `substr()`. Единственная разница заключается в том, что индекс первого символа строки для этой функции имеет значение 1, а не 0, как для PHP-версии. Поэтому извлечение первых 100 символов описания работы будет выглядеть в этом случае так:

```
SELECT SUBSTRING(job_description, 1, 100)
    FROM riscyjobs;
```

Преимущество использования PHP-версии заключается в том, что в этом случае нам доступны как полные варианты описания работ, так и сокращенные. При использовании MySQL-версии нам будут доступны только сокращенные варианты описания работ, и в случае возникновения необходимости посмотреть полный вариант нам придется делать дополнительный запрос к базе данных.

Не бывает глупых вопросов

В: Может ли функция substr() обрабатывать числовые значения?

О: Нет, она имеет дело только со строчными переменными. Но если вы сохраните в базе данных число в виде `CHAR`, `VARCHAR` или `TEXT`, то при извлечении этого числа из базы данных PHP будет рассматривать его как строку, и в таком случае вы сможете обработать его с помощью функции `substr()`.

В: Что будет, если передать функции substr() в качестве длины подстроки величину большую, чем длина всей строки? Вернет ли она всю строку с символами пробелов в конце в таком количестве, чтобы ее длина соответствовала переданному значению?

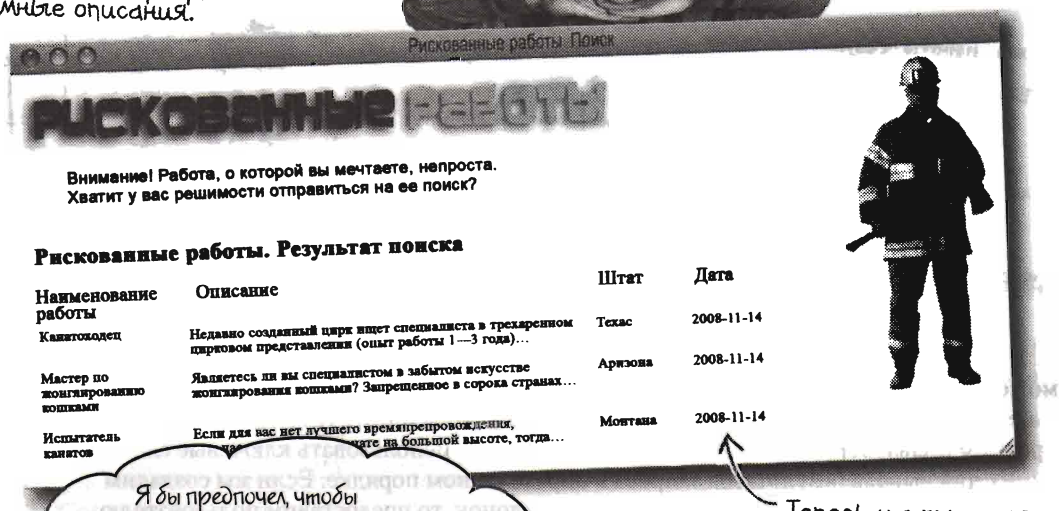
О: Она вернет всю строку. Но никаких пробелов в конце добавлено не будет. Например, в результате выполнения следующего кода будет получена строка «собака»: `substr('собака', 0, 10)`.

Тест-драйв

Внесите изменения в сценарий поиска так, чтобы уменьшить размеры строк, содержащих описание работы, и дату регистрации объявления.

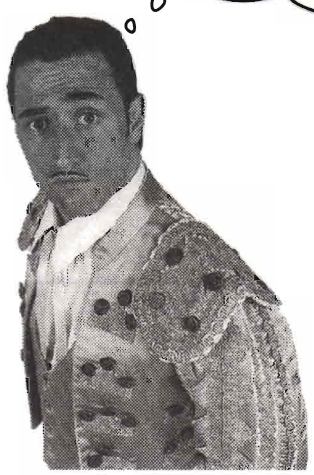
Обновите сценарий `search.php` так, чтобы в нем использовалась функция `substr()` для уменьшения длины выводимых в результате поиска строк, содержащих описание работы и дату регистрации объявления. Затем загрузите сценарий на ваш веб-сервер и попробуйте сделать несколько вариантов поисков работы.

Сельма теперь довольна, потому что она видит результаты поиска без необходимости прокручивать эти огромные описания.



Я бы предпочел, чтобы результаты были отсортированы по дате регистрации или по штатам. Я был бы не против найти работу матадора в Вермонте.

Теперь и дату стало намного легче воспринимать, так как показана только дата, а не дата и время.

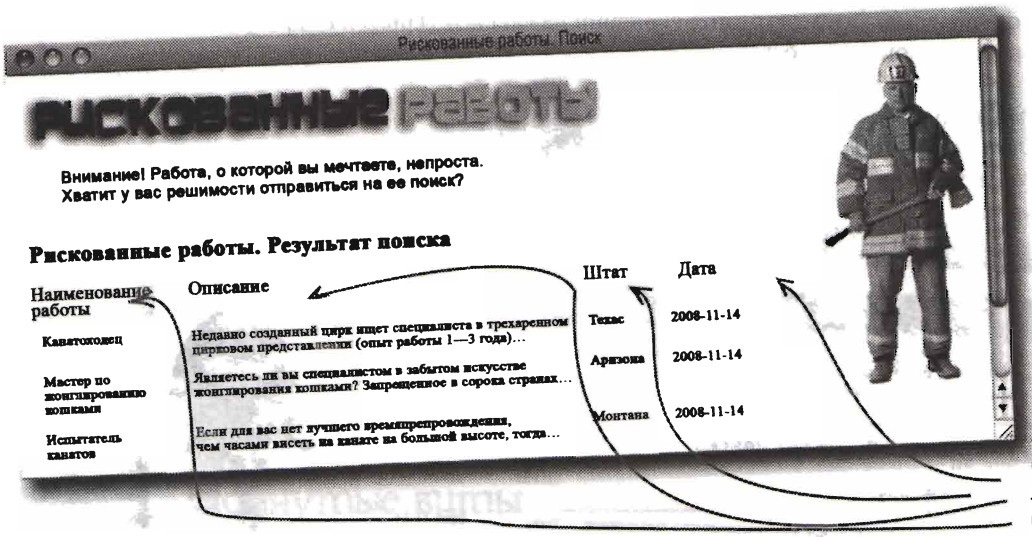


СИЛА МЫСЛИ

Как мы теперь должны изменить запрос и расположение элементов на странице, чтобы появилась возможность сортировать данные по дате регистрации объявления, штату или наименованию работы?

Несколько запросов позволят нам сортировать результаты нашего поиска

Для того чтобы позволить пользователям сортировать результаты своих поисков, нам необходим способ, дающий им возможность определить, по какому параметру они хотят отсортировать результаты поиска. Может, форма... или кнопка? Мы можем использовать HTML, чтобы превратить заголовок каждой колонки таблицы результатов поиска в гиперссылку. Пользователь может щелкнуть кнопкой мыши на гиперссылке, чтобы указать, какую из них использовать для сортировки результатов запроса.



Пользователь может разобраться с предложениями более тщательно, отсортировав результаты поиска по любой из этих колонок. Мы можем превратить эти наименования в гиперссылки, чтобы предоставить пользователю возможность щелкать кнопкой мыши на любом из них для выбора желаемого параметра сортировки.

Мы можем использовать эти гиперссылки для перезагрузки того же самого сценария поиска, но с запросом, в результате выполнения которого данные будут отсортированы в выбранном пользователем порядке. Мы уже знаем, как использовать ключевые слова `ORDER BY` для вывода результатов запроса в определенном порядке. Если мы создадим различные запросы с сортировкой по каждой из колонок, то предоставим пользователю возможность сортировать результаты поиска по наименованию работ, описанию, штату или располагать записи в хронологическом порядке.

Вот SQL-запрос с сортировкой результатов запроса в алфавитном порядке описаний работ:

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%Матадор%' OR
  description LIKE '%победитель%' OR description LIKE '%быков%'
  ORDER BY description
```

Результаты запроса будут отсортированы по описанию работ в алфавитном убывающем порядке.

Время поработать

Напишите три различных запроса, в результате выполнения которых данные по поиску рискованных работ будут отсортированы по наименованию работ, штату и дате регистрации. Пользователь ввел в поле ввода поиска строку «мойщик, окно, небоскреб».

Как могли бы мы переписать эти запросы, если бы захотели отсортировать наименования работ и штаты в обратном порядке? И как поместить в начало самые последние зарегистрированные объявления?

Решение задачи

Напишите три различных запроса, в результате выполнения которых данные по поиску рискованных работ будут отсортированы по наименованию работ, штату и дате регистрации. Пользователь ввел в поле ввода поиска строку «мойщик, окно, небоскреб».

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%мойщик%' OR description LIKE '%окно%' OR
  description LIKE '%небоскреб%'
  ORDER BY job_title
```

Для выражения ORDER BY порядком по умолчанию является нисходящий (ASCending) порядок. Такая запись аналогична записи: ODRER BY job_title ASC.

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%мойщик%' OR description LIKE '%окно%' OR
  description LIKE '%небоскреб%'
  ORDER BY state
```

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%мойщик%' OR description LIKE '%окно%' OR
  description LIKE '%небоскреб%'
  ORDER BY date_posted
```

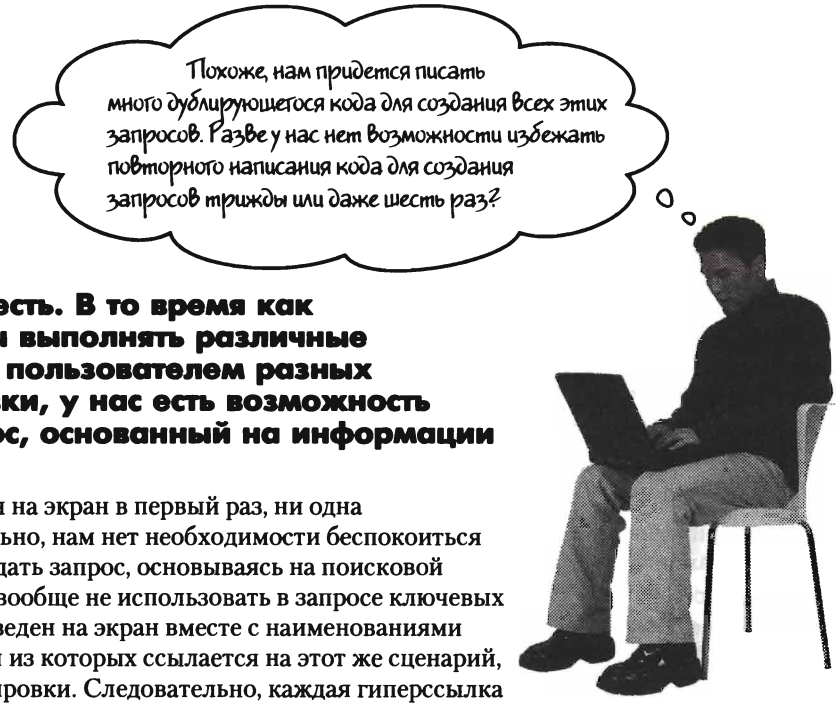
Как могли бы мы переписать эти запросы, если бы захотели отсортировать наименования работ и штаты в обратном порядке? И как поместить в начало самые последние зарегистрированные объявления?

Мы могли бы использовать такие запросы, например, если результаты уже выведены в нисходящем (по умолчанию) порядке для определенной колонки, но пользователь щелкает опять на той же гиперссылке, давая понять, что он хочет изменить нисходящий порядок на восходящий.

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%мойщик%' OR description LIKE '%окно%' OR
  description LIKE '%небоскреб%'
  ORDER BY job_title DESC
```

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%мойщик%' OR description LIKE '%окно%' OR
  description LIKE '%небоскреб%'
  ORDER BY state DESC
```

```
SELECT * FROM riskyjobs
  WHERE description LIKE '%мойщик%' OR description LIKE '%окно%' OR
  description LIKE '%небоскреб%'
  ORDER BY date_posted DESC
```



Такая возможность у нас есть. В то время как мы действительно должны выполнять различные запросы в ответ на выбор пользователем разных гиперссылок для сортировки, у нас есть возможность создать конкретный запрос, основанный на информации этой гиперссылки.

Когда результаты запроса выводятся на экран в первый раз, ни одна гиперссылка не выбрана, следовательно, нам нет необходимости беспокоиться о сортировке. Мы можем просто создать запрос, основываясь на поисковой строке, введенной пользователем, и вообще не использовать в запросе ключевых слов `ORDER BY`. Результат будет выведен на экран вместе с наименованиями колонок в виде гиперссылок, каждая из которых ссылается на этот же сценарий, но с различными параметрами сортировки. Следовательно, каждая гиперссылка должна состоять из URL с критериями поиска и параметром, который определяет порядок сортировки результата запроса.

Что бы нам действительно помогло в этом деле, так это создание наших собственных пользовательских функций, которые, получив информацию о том, в каком порядке должны быть отсортированы записи в результате запроса, возвращали бы нам готовую строку запроса с соответствующим условным выражением `WHERE` и порядком сортировки `ORDER BY`. Наша новая пользовательская функция, используя значение параметра `sort`, определяет порядок сортировки записей в результате запроса. Ниже приведены этапы, которые должна пройти информация в функции в процессе обработки.

1. Предварительная обработка поисковой строки и сохранение критериев поиска в массиве.
2. Выборочное использование значения параметра `sort`, который говорит о том, в каком порядке должны быть отсортированы записи в результате запроса.
3. Удаление пустых критериев поиска.
4. Создание условного выражения `WHERE`, содержащего все критерии поиска.
5. Если параметр `sort` имеет какое-либо значение — добавление к запросу выражения порядка сортировки `ORDER BY`.
6. Передача в основной сценарий полностью созданной строки запроса.

Может показаться, что все это потребует много работы. Но большая часть кода у нас уже написана. Нам только остается передать этот код нашей пользовательской функции. Но прежде чем приступать к этому, давайте посмотрим, как создаются пользовательские функции...

Функции дают вам возможность использовать код повторно

Функция — это блок кода, отделенный от основного кода, и вы можете использовать его в вашем сценарии в любой необходимый момент. До сих пор вы использовали встроенные PHP-функции. `Explode()`, `substr()` и `mysqli_query()` являются примерами функций, которые заранее определены в интерпретаторе PHP и могут быть вызваны из любого сценария.

Пользовательские функции позволяют вам дать имя блоку кода для того, чтобы можно было выполнять его многократно.

Но у вас также имеется возможность определить свои собственные пользовательские функции для задач, решение которых не предусмотрено функциями, встроенными в интерпретатор языка. Создав пользовательскую функцию, вы можете использовать ее код столько раз, сколько вам это нужно, без необходимости каждый раз переписывать один и тот же код. Вместо этого вы просто вызываете функцию по ее имени, когда у вас возникает необходимость выполнить записанный в ней код.

Ниже приведен пример пользовательской функции с именем `replace_commas()`, в задачу которой входит заменить в строке все запятые на символы пробелов:

Коду каждой пользовательской функции предшествует ключевое слово function.

Это имя функции, которое вы выбираете сами. Старайтесь делать его настолько информативным, насколько это возможно.

Вслед за именем функции располагается пара круглых скобок, внутри которых вы перечисляете имена аргументов, отделенных друг от друга запятыми. В данном случае используется только один аргумент.

```
function replace_commas($str) {
  $new_str = str_replace(',', ' ', $str);
  return $new_str;
}
```

Фигурными скобками ограничивается блок кода функции аналогично тому, как это делается в циклах или управляющих конструкциях if.

Функция может возвращать какое-либо значение коду, ее вызвавшему. В данном случае функция replace_commas() возвращает измененную строку.

Когда вам необходимо использовать функцию, просто вызовите ее по имени, передав в круглых скобках необходимое значение, которое ожидают ее аргументы. Если функция разработана так, что она возвращает какое-либо значение, вы можете присвоить его любой переменной, как показано в примере:

В качестве аргумента функции передается строка «канат, канатоходец, цирк».

```
$clean_search = replace_commas('tightrope, walker, circus');
```

Функция возвращает новую строку, в которой все запятые заменены символами пробелов.

Создание запроса с использованием пользовательской функции

У нас уже есть большая часть кода, необходимого для пользовательской функции, которая должна создавать запрос для поиска рискованных работ. Все, что нам осталось, — это только передать этот код нашей пользовательской функции. Вот так выглядит пользовательская функция `build_query()`:

Мы передаем функции в качестве аргумента массив $user_search, который мы создали, используя данные, введенные пользователем в поисковую форму.

```
function build_query($user_search) {
  $search_query = "SELECT * FROM riskyjobs";

  // Извлечение критериев поиска в массив
  $clean_search = str_replace(',', ' ', $user_search);
  $search_words = explode(' ', $clean_search);
  $final_search_words = array();
  if (count($search_words) > 0) {
    foreach ($search_words as $word) {
      if (!empty($word)) {
        $final_search_words[] = $word;
      }
    }
  }

  // Создание условного выражения WHERE с использованием всех
критериев поиска  $where_list = array();
  if (count($final_search_words) > 0) {
    foreach($final_search_words as $word) {
      $where_list[] = "description LIKE '%$word%'";
    }
  }
  $where_clause = implode(' OR ', $where_list);

    // Добавление условного выражения WHERE к поисковому запросу
  if (!empty($where_clause)) {
    $search_query .= " WHERE $where_clause";
  }

  return $search_query;
}
```

Внутри функции ничего нового.

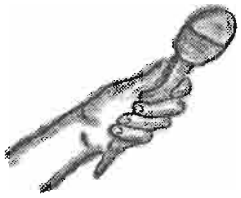
Вот фактически все новшества. Здесь мы возвращаем новую строку запроса для того, чтобы код, вызвавший функцию, мог использовать этот запрос.

Функция `build_query()` возвращает полную строку SQL-запроса, основанную на поисковой строке, переданной ей в виде значения аргумента `$user_search`. Для того чтобы использовать функцию, мы просто вызываем ее, передавая в качестве аргумента поисковую строку, введенную пользователем, и затем присваиваем переменной `$search_query` возвращаемое функцией значение:

```
$search_query = build_query($user_search);
```

Переменной $search_query присваивается возвращаемое функцией значение, в этом случае — наш новый поисковый запрос.

Значение этой переменной соответствует данным, введенным пользователем в поисковую форму.



Пользовательская функция: близкий взгляд

Интервью этой недели:
Пользовательские функции: насколько они пользовательские?

Корреспондент редакции Head First: Послушайте, нас всех тут интересует один вопрос: в конце концов, что плохого в дублировании кода? Я хочу сказать, что в действительности все это очень просто. Вы копируете нужный фрагмент кода в буфер и затем вставляете его в нужное место. Вот и все.

Пользовательская функция: О, не начинайте про дублирование кода. Подобные сценарии выглядят просто уродливо, а пытаться читать и понимать их — просто мука. Все это уже плохо само по себе, но существует множество еще очень важных причин избегать дублирования кода.

Корреспондент редакции Head First: Да?

Пользовательская функция: Хорошо, представьте себе, что вам нужно что-то изменить в вашем коде. Такая необходимость возникает довольно часто.

Корреспондент редакции Head First: Ну и что? Все постоянно меняется. Вы просто открываете в редакторе текст своего сценария и вносите необходимые изменения.

Пользовательская функция: А что если эти изменения затрагивают дублирующийся код? И встречаются в пяти, а то и в десяти местах вашего приложения?

Корреспондент редакции Head First: Я не вижу никаких проблем. Вы просто находите эти места и исправляете код в них. И готово.

Пользовательская функция: Отлично. Но что, если вы пропустите одно такое место? Хоть вы и программист, но всего лишь человек. Если вы допустите такую оплошность, вам, возможно, придется здорово попотеть в попытках понять, что же случилось с вашим приложением.

Корреспондент редакции Head First: Пожалуй, я соглашусь, что такое может случиться, но в чем заключается ваша помощь?

Пользовательская функция: Ах, так это же ваше счастье, что вы имеете дело со мной! Если проблемный код находится в функции, вам необходимо изменить его только в функции. Всего лишь однажды. Раз, два — и готово.

Корреспондент редакции Head First: Должен вам заметить, что все это звучит прекрасно. Но я все еще не понимаю, почему я должен сворачивать с прямого пути и пользоваться вашими услугами. Я хочу сказать, что вы достаточно ограничены в своих возможностях, так? Вы ведь можете работать только со строчными переменными.

Пользовательская функция: Стоп! Подождите секундочку, ковбой! Я могу работать с любыми типами данных, которые вы мне передадите. Пока код внутри меня будет написан с соблюдением всех правил, диктуемых языком, я могу работать с любыми типами данных, которые вам необходимы. Я использовала массив в том последнем примере, черт побери. Я бы сказала, что работа с этими массивами — дьявольски мудреная штука.

Корреспондент редакции Head First: Но вы возвратили строку.

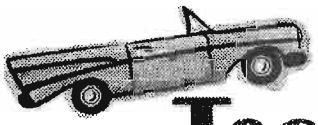
Пользовательская функция: Я могу возвратить все, что вы захотите. Все зависит от того, насколько корректно вы будете использовать мои возможности.

Корреспондент редакции Head First: Это другая сторона дела. Вы слишком требовательны. Вы требуете каких-нибудь данных при своем вызове.

Пользовательская функция: Где вы набираетесь подобных глупостей? Вы можете вызывать меня без каких-либо аргументов. Не пишите в этом случае никаких переменных в круглых скобках, следующих за моим именем, когда создаете меня. Хотя мне трудно представить причины, по которым у вас не возникло бы необходимости передавать мне какие-либо данные, так же как и получать их от меня.

Корреспондент редакции Head First: Наше время заканчивается. Спасибо за интервью.

Пользовательская функция: Не стоит благодарности.



Тест-драйв

Обновите сценарий поиска так, чтобы в нем использовалась функция build_query().

Создайте в сценарии `search.php` функцию `build_query()`, заменив первоначальный код вызовом новой функции `implode()`. Загрузите сценарий на ваш веб-сервер и попробуйте сделать несколько вариантов поиска в браузере, чтобы убедиться, что все в порядке.

Эта новая пользовательская функция build_query(), конечно, хороша, но она пока не сортирует результаты поиска. Могли бы мы передать функции еще один аргумент, чтобы она могла решить и эту задачу?

Вполне разрешимая задача. Мы можем передать функции build_query() два параметра вместо одного.

Мы уже передаем функции аргумент `$user_search`, который содержит критерии поиска, введенные пользователем. Теперь нам необходим еще один аргумент, `$sort`, который определит вид сортировки результатов поиска. Новый аргумент `$sort` должен управлять созданием возвращаемой функцией строки запроса в шести вариантах, которые мы обсуждали: с сортировкой по наименованию работ, штату и дате регистрации объявления в нисходящем и восходящем порядке.

Мы могли бы присвоить значение выражения порядка сортировки (`ORDER BY`) переменой `$sort`. Или использовать числа от 1 до 6 для представления каждого из порядка сортировки, как показано ниже:

```
$sort == 1 ➲ ORDER BY job_title
$sort == 2 ➲ ORDER BY job_title DESC
$sort == 3 ➲ ORDER BY state
$sort == 4 ➲ ORDER BY state DESC
$sort == 5 ➲ ORDER BY date_posted
$sort == 6 ➲ ORDER BY date_posted DESC
```

Нет серьезных оснований для сортировки по описанию работы, так как расположение этой информации в алфавитном порядке мало что дает.

Мы выбрали эти числа и их значения совершенно случайно. Не существует жестких правил по их выбору, кроме соответствия.

Но разве использование чисел не затрудняет чтение и понимание кода? Без информативных комментариев — затрудняет, но существуют важные основания использовать в этой ситуации числа. Если мы будем использовать значения выражений порядка сортировок (`ORDER BY`) в виде строк, они появятся в URL сценария как его составная часть. Это откроет широкий доступ к наименованиям колонок таблицы `riskyjobs`, что является весьма нежелательным из соображений безопасности.

Хорошо, я поняла, как работает этот новый аргумент $sort, но как мы определим, какое число должно быть передано нашей функции? Неужели пользователь должен говорить нам об этом?

Да, пользователь должен определить, в каком порядке будут отсортированы результаты поиска, точно так же, как он указывает сами критерии поиска.

Хорошо то, что мы уже определились с методом реализации: мы намерены представить наименования колонок результата поиска в виде гиперссылок. Когда пользователь щелкнет кнопкой мыши на каком-либо наименовании колонки, например «Штат», мы передадим функции `build_query()` число, которому соответствует строка запроса с сортировкой по наименованию штатов.

Нам необходимо перезагрузить страницу после того, как пользователь щелкнет кнопкой мыши на каком-либо наименовании колонки, поэтому мы создаем форму, ссылающуюся на себя.

Но нашему сценарию все еще необходимо получить от гиперссылки значение порядка сортировки. Мы можем обеспечить это при создании гиперссылки путем добавления аргумента `$sort` к ее URL:

Результат поиска создается в виде HTML-таблицы, вот почему здесь теги <td>.

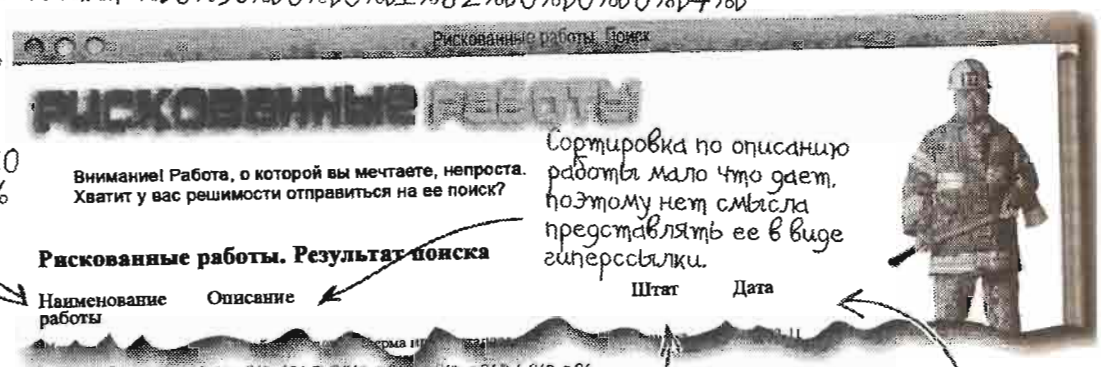
```
$sort_links .= '<td><a href="' . $SERVER['PHP_SELF'] .
    '?usersearch=' . $user_search . '&sort=3">Штат</a></td>';
```

Нашей функции build_query() необходимы введенные пользователем критерии поиска, которые мы и передаем в составе URL.

Мы передаем информацию о желаемом порядке сортировки результатов поиска. Так как это гиперссылка, соответствующая колонке «Штат», параметр sort равен 3.

При создании страницы результата поиска каждое наименование колонки (кроме колонки «Описание работы») представлено в виде гиперссылки со своим собственным URL, включающим аргумент `sort` со значением, которое определяет порядок сортировки, выбранный пользователем.

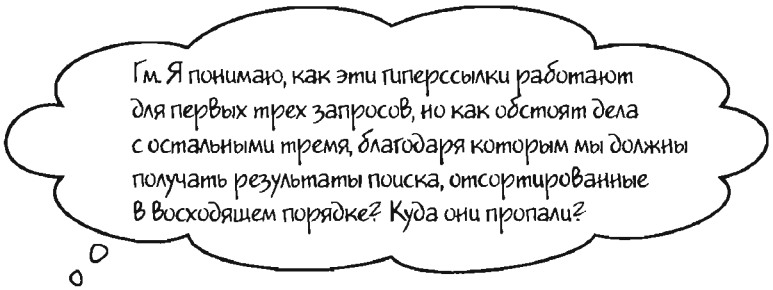
«<a href="search.php?usersearch=%D0%9C%D0%B0%D1%82%D0%B0%D0%B4%D0%BE%D1%80%20%D0%BF%D0%BE%D0%B1%D0%B5%D0%B4%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%20%D0%B1%D1%8B%D0%BA%D0%BE%D0%B2&sor t=1

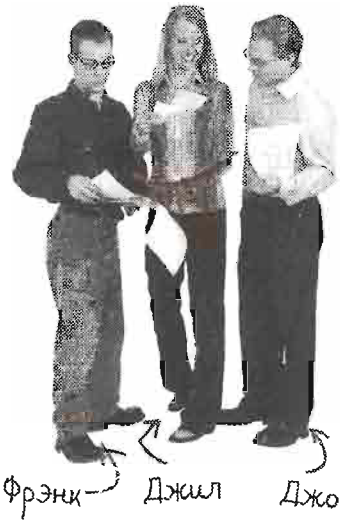


Сортировка по описанию работы мало что дает, поэтому нет смысла представлять ее в виде гиперссылки.

«<a href="search.php?usersearch=%D0%9C%D0%B0%D1%82%D0%B0%D0%B4%D0%BE%D1%80%20%D0%BF%D0%BE%D0%B1%D0%B5%D0%B4%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%20%D0%B1%D1%8B%D0%BA%D0%BE%D0%B2&sor t=3">

«<a href="search.php?usersearch=%D0%9C%D0%B0%D1%82%D0%B0%D0%B4%D0%BE%D1%80%20%D0%BF%D0%BE%D0%B1%D0%B5%D0%B4%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%20%D0%B1%D1%8B%D0%BA%D0%BE%D0%B2&sor t=5">





Джо: Одна и та же гиперссылка должна позволять пользователю задавать как нисходящий порядок сортировки, так и восходящий.

Джил: Это так. Каждый раз, когда пользователь будет щелкать на одной и той же ссылке, порядок сортировки должен меняться на обратный.

Фрэнк: Разве это не значит, что мы должны теперь как-то помнить порядок сортировки для колонки, на гиперссылке которой пользователь щелкает кнопкой мыши, потому что на этот раз записи должны будут отсортироваться в порядке, обратном текущему?

Джо: Не пойму, о чем ты говоришь.

Фрэнк: Ну смотри. Гиперссылка не должна каждый раз определять один и тот же порядок сортировки. Например, если ты щелкнул кнопкой мыши на гиперссылке «Наименование работы», она должна включать в URL данные для сортировки наименований работ в нисходящем алфавитном порядке. Но при повторном щелчке на этой же гиперссылке ее URL должен содержать данные для сортировки тех же наименований работ, но теперь уже в восходящем алфавитном порядке.

Джил: Все правильно. Но имей в виду, что каждый вид сортировки определяется передаваемым в составе URL числом, исходя из величины которого сценарий выбирает вид сортировки, а раз мы создаем этот URL, то можем задавать, какое именно число включить в его состав.

Джо: Я понимаю. Значит, мы должны составить наш код так, чтобы состав URL гиперссылки учитывал текущее состояние порядка сортировки, правильно?

Фрэнк: О, я понял! Разве мы не решали уже подобную задачу с помощью нескольких управляющих конструкций `if`? Я имею в виду их отменную способность принимать решения, что скажете?

Джо: Это, конечно, будет работать, но мы говорим о принятии нескольких решений, основанных на значении одной и той же переменной, содержащей тип сортировки. Было бы значительно лучше, если бы мы у нас была возможность воспользоваться более удобным методом, чем несколько вложенных управляющих конструкций `if-else`.

Джил: Очень правильное замечание, и это отличная возможность попробовать новую управляющую конструкцию, о которой я недавно услышала. Управляющая конструкция `switch` позволяет вам принимать одно из множества (значительно больше двух) решений, основываясь на значении одной переменной.

Фрэнк: Это звучит обнадеживающе. Давайте попробуем.

Джо: Согласен. Что угодно, только не переусложненные `if-else` конструкции, у меня от них голова болит.

Джил: У меня тоже. Я думаю, управляющая конструкция `switch` — это то, что нам нужно...

Управляющая конструкция SWITCH выбирает нужное решение из значительно большего набора возможных вариантов, чем IF

Управляющая конструкция `switch` предлагает эффективный способ выполнять определенный блок кода в зависимости от значения переменной этой конструкции. Это что-то, что потребовало бы задействовать небольшую армию управляющих конструкций `if-else`, особенно в ситуациях с выбором из множества возможных вариантов.

Управляющая конструкция SWITCH состоит из серии меток CASE, каждая из которых обозначает свой блок кода. Блоку кода какой метки будет передано управление, зависит от значения переменной этой конструкции.

Вместо того чтобы составлять множество вложенных управляющих конструкций `if-else` для проверки всех возможных вариантов, вы создаете управляющую конструкцию `switch`, в состав которой входит множество меток `case`. Каждая из них включает одно из возможных значений переменной конструкции `switch` и блок кода. Если в конце блока кода помещено выражение `break`, управление передается строке кода, следующей за управляющей конструкцией `switch`, при этом все остальные метки игнорируются. Таким образом, завершение блока кода выражением `break` гарантирует, что при любом значении переменной управляющей конструкции `switch` будет выполнен только один блок кода, а какой именно — зависит от значения этой переменной.

Давайте посмотрим на пример использования управляющей конструкции `switch`:

```
switch ($benefit_code) {
case 1:
  $benefits = 'Многосторонняя медицинская страховка на 10 дней заболевания';
  break;
case 2:
  $benefits = 'Только вскрытие после смерти. Остался один оплаченный месяц';
  break;
case 3:
case 4:
  $benefits = 'Всего хорошего!';
  break;
default:
  $benefits = 'Ничего.';
}

echo 'Мы предлагаем четыре обширных пакета льгот';
echo 'Пакет, выбранный вами: ' . $benefits;
```

Это переменная управляющей конструкции switch, ее значение определяет, какой из блоков кода будет выполнен.

Выражение break говорит интерпретатору PHP передать управление первой строке кода после управляющей конструкции switch.

Этот блок кода будет выполнен, только если значение переменной $benefit_code равно 1.

Если вы хотите, чтобы один и тот же блок кода выполнялся для двух и более значений переменной, просто поместите его после всех перечисленных меток.

Если переменная имеет любое значение, кроме 1, 2, 3 или 4, будет выполнен этот блок кода.

На самом деле только три пакета, так как 3 и 4 — метки одного и того же пакета: за меткой 3 не следует никакого блока кода, включая break.

В приложении «Рискованные работы» имеется функция под названием `generate_sort_links()`, которая дает пользователям возможность сортировать результаты поиска щелчком кнопки мыши на заголовке колонки. К сожалению, часть очень важного кода утрачена. Восстановите код этой функции. И не забудьте числа, которым соответствуют виды сортировок: 1 = наименования работ в нисходящем алфавитном порядке, 2 = наименования работ в восходящем алфавитном порядке, 3 = наименования штатов в нисходящем алфавитном порядке, 4 = наименования штатов в восходящем алфавитном порядке, 5 = даты регистрации объявлений в нисходящем порядке, 6 = даты регистрации объявлений в восходящем порядке.

```
..... generate_sort_links($user_search, $sort) {
  $sort_links = '';

  .....($sort) {
  case 1:
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Дата</a></td>';
    .....
  case 3:
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Дата</a></td>';
    .....
  case 5:
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Дата</a></td>';
    .....
  .....
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= .....">Дата</a></td>';
  }

  return .....;
}
```

Это код создания заголовков по умолчанию, такие заголовки появляются, когда пользователь не выбрал никакого метода сортировки.

авершенная функция generate_sort_links()

В приложении «Рискованные работы» имеется функция под названием `generate_sort_links()`, которая дает пользователям возможность сортировать результаты поиска щелчком кнопки мыши на заголовке колонки. К сожалению, часть очень важного кода утрачена. Восстановите код этой функции. И не забудьте числа, которым соответствуют виды сортировок: 1 = наименования работ в нисходящем алфавитном порядке, 2 = наименования работ в восходящем алфавитном порядке, 3 = наименования штатов в нисходящем алфавитном порядке, 4 = наименования штатов в восходящем алфавитном порядке, 5 = даты регистрации объявлений в нисходящем порядке, 6 = даты регистрации объявлений в восходящем порядке.

```
function generate_sort_links($user_search, $sort) {
  $sort_links = '';

  switch ($sort) {
  case 1:
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 2 ">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 3 ">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 5 ">Дата</a></td>';
    break;
  case 3:
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 1 ">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 4 ">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 3 ">Дата</a></td>';
    break;
  case 5:
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 1 ">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 3 ">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 6 ">Дата</a></td>';
    break;
  default:
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 1 ">Наименование работы</a></td><td>Описание</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 3 ">Штат</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
      '&sort= 5 ">Дата</a></td>';
  }

  return $sort_links;
}
```

Если переменная $sort имеет значение 1, это означает, что данные уже отсортированы по наименованию работ и нам необходимо пересортировать их в нисходящем порядке!

Если переменная $sort не имеет какого-либо значения или если она равна 2, 4 или 6, мы должны вывести информацию в первоначальном виде — отсортированную по датам регистрации объявлений в нисходящем порядке.

Дайте функции build_query() возможность сортировать данные

У нас теперь есть две функции для организации поиска в приложении «Рискованные работы». Функция `build_query()` создает SQL-запрос, основываясь на критериях поиска, введенных пользователем. Функция `generate_sort_links()` создает для заголовков таблицы с результатами поиска гиперссылки, которые позволяют пользователям сортировать данные в этой таблице. Но функция `build_query()` еще не закончена, так как в создаваемых запросах пока не определен порядок сортировки данных. Ко всем создаваемым функцией запросам необходимо добавить выражение `ORDER BY`. Но это должно быть **правильное** выражение `ORDER BY`, соответствующее новому аргументу `$sort`:

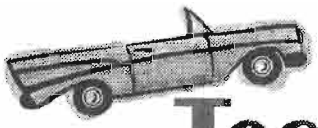
Теперь наряду с критериями поиска ($user_search) мы также передаем функции в качестве аргумента порядок сортировки ($sort).

```
function build_query($user_search, $sort) {
  $search_query = "SELECT * FROM riskyjobs";
  ...
  // Добавление к запросу условного выражения WHERE
  if (!empty($where_clause)) {
    $search_query .= " WHERE $where_clause";
  }

  // Добавление к запросу выражения, определяющего порядок сортировки
  switch ($sort) {
    // Сортировка по наименованию работ в восходящем алфавитном порядке
    // (от А к Я)
    case 1:
      $search_query .= " ORDER BY title";
      break;
    // Сортировка по наименованию работ в нисходящем алфавитном порядке
    // (от Я к А)
    case 2:
      $search_query .= " ORDER BY title DESC";
      break;
    // Сортировка по наименованию штата в восходящем алфавитном порядке
    // (от А к Я)
    case 3:
      $search_query .= " ORDER BY state";
      break;
    // Сортировка по наименованию штата в нисходящем алфавитном порядке
    // (от Я к А)
    case 4:
      $search_query .= " ORDER BY state DESC";
      break;
    // Сортировка по дате регистрации объявления в восходящем порядке
    // (от более ранних объявлений к более поздним)
    case 5:
      $search_query .= " ORDER BY date_posted";
      break;
    // Сортировка по дате регистрации объявления в нисходящем порядке
    // (от более поздних объявлений к более ранним)
    case 6:
      $search_query .= " ORDER BY date_posted DESC";
      break;
    default:
      // Данные по порядку сортировки отсутствуют,
      // поэтому записи выводятся в том порядке,
      //   в котором они расположены в таблице
  }

  return $search_query;
}
```

Это код, добавленный к функции build_query(). С помощью этой управляющей конструкции switch в конец запроса добавляется выражение ORDER BY, определяющее порядок сортировки в соответствии со значением аргумента $sort.

Если пользователь открывает страницу без указания порядка сортировки, аргумент $sort не будет иметь никакого значения, поэтому (по умолчанию) результат поиска будет выведен неотсортированным.

Так же, как и прежде, функция возвращает строку запроса, но на этот раз с выражением ORDER BY, определяющим порядок сортировки в соответствии со значением аргумента $sort.



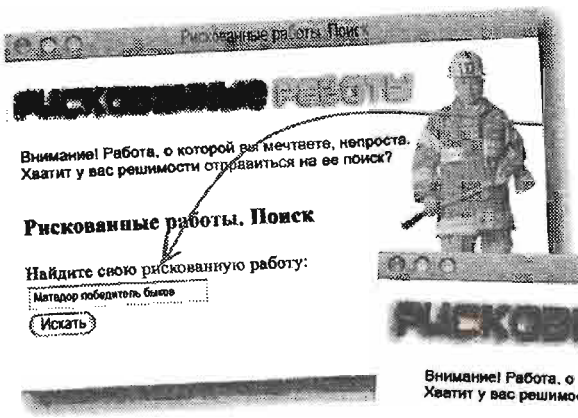
Тест-драйв

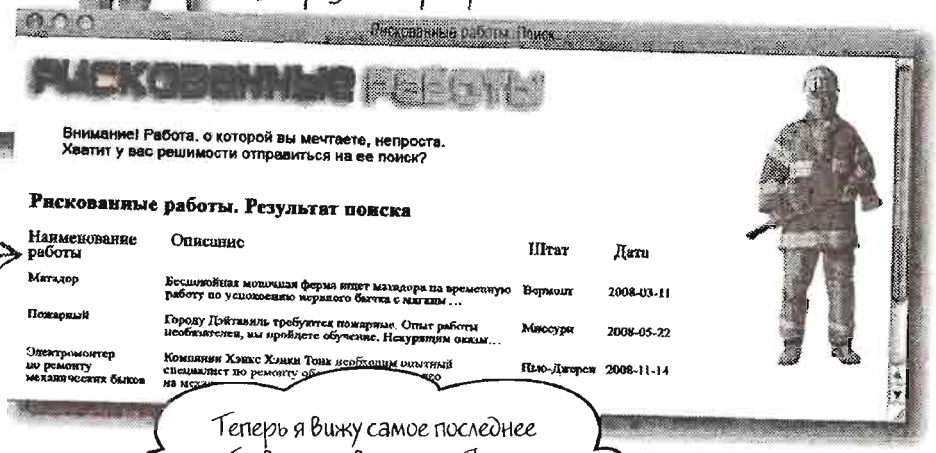
Откорректируйте сценарий «Поиск» так, чтобы в нем использовались две новые пользовательские функции.

Создайте новую функцию в сценарии `generate_sort_links()` и затем добавьте новый код к функции `build_query()` так, чтобы строка запроса создавалась с выражением `ORDER BY`, определяющим порядок сортировки его результатов. Не забудьте заменить в сценарии код, создающий гиперссылки — наименования колонок таблицы, на вызов функции `generate_sort_links()`.

Загрузите сценарий на ваш веб-сервер и откройте форму поиска `search.html` в браузере. Попробуйте сделать несколько вариантов поиска. Затем щелкните кнопкой мыши на гиперссылках (наименованиях колонок таблицы), чтобы проверить, как выполняется сортировка. Не забудьте щелкнуть на одной и той же гиперссылке больше одного раза, чтобы убедиться, что порядок сортировки меняется на обратный после каждого последующего щелчка.

Функция build_query() извлекает поисковую строку, которую пользователь ввел в форму, разбивает ее на отдельные элементы и сохраняет их в массиве, затем удаляет из массива пустые элементы и создает запрос с условным выражением и выражением порядка сортировки в соответствии со значением аргумента $sort при условии, что перед этим пользователем выбран один из порядков сортировки.





Функция generate_sort_links() создает гиперссылки — наименования колонок таблицы — и включает в их URL значение параметра сортировки.

Теперь я вижу самое последнее объявление о вакансии. Похоже, там, в Вермонте, не так-то легко нанять матадора.

Слишком большое количество объявлений, чтобы посмотреть их за один раз.

Внимание! Работа, о которой вы мечтаете, непроста. Хватит у вас решимости отправиться на ее поиск?

Рискованные работы. Результат поиска

Наименование работы	Описание	Штат	Дата
Ходящий по заварному крему	Мы ищем добровольцев, желающих проверить теорию, утверждающую, что по заварному крему можно ходить.	Нью-Мексико	2008-07-24
Дрессировщик акул	Дрессировка акул для выполнения различных трюков перед зрителями нашего нового аквапарка. Вы один бу…	Флорида	2008-04-28
Измеритель электрического напряжения	Вы будете проверять переменное и постоянное напряжение в пределах от 3-х до 250-ти вольт. Вам будет…	Северная Каролина	2008-06-28
Установщик антенн	Вы будете устанавливать антенны и другое радиооборудование на крышах самых высоких зданий Майами. Вы…	Флорида	2008-09-04
Слоновий проктолог	Требуется опытный проктолог, желающий работать с животными больших размеров. Слоны в нашем зоопарке…	Калифорния	2008-07-29
Специалист по чистке авиационных двигателей	Двигатели реактивных самолетов требуют периодической очистки. Необходим человек, любящий чистоту и ж…	Техас	2008-08-17
Матадор	Беспокойная молочная ферма ищет матадора на временную работу по успокоению нервного бычка с мягким…	Вермонт	2008-03-11
Папарацци	Студия по фотографированию популярных личностей ищет опытного папарацци для тайной съемки темпераме…	Калифорния	2008-03-24
Канатоходец	Недавно созданный цирк ищет специалиста …сканенном цирковом представлении …т работы 1—3 года) …	Техас	2008-11-14
	…вам любите животных и ненавидите камни … Нашей крокодильей ферме тре…	Флорида	2008-07-14
	… Полная страховка …ляются. Должен любить детей…	Нью-Йорк	2008-11-02
	…корма для животных. …лучше. Обращайтесь прямо сей…	Миссури	2008-11-09
	…т ваших совершенных, …ой. Вы должны пройти базовый …	Айдахо	2008-11-14
по ремонту	…т необходим опытный специалист … по ремонту оборудования для родео на механике…	Нью-Джерси	2008-11-14

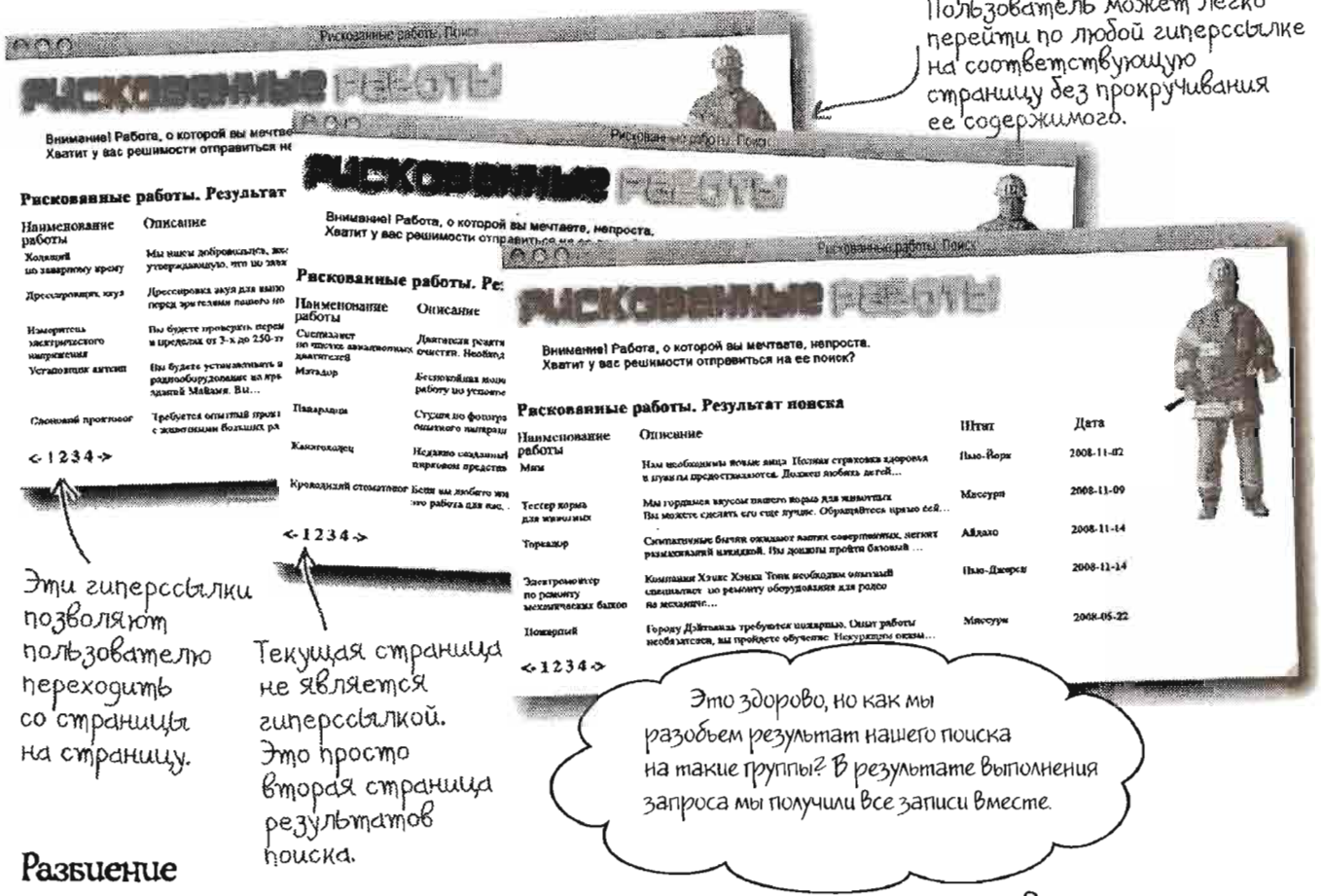
СИЛА МЫСЛИ

А как другие сайты избегают такого большого объема результатов поиска на одной странице?

Мы можем разбить результат поиска на страницы

Сейчас мы показываем весь результат поиска на одной странице, что не слишком удобно, когда заданным пользователем критериям поиска соответствует множество объявлений. Вместо того чтобы заставлять пользователя в поисках нужной ему информации прокручивать вверх и вниз длинную страницу, мы можем использовать разбивку всего результата поиска на отдельные страницы (пагинацию). Этот процесс заключается в разбивке всего набора объявлений на отдельные группы и выводе каждой группы на своей странице, как показано ниже.

Каждая страница содержит по пять объявлений вместе с гиперссылками на другие страницы результатов поиска. Пользователь может легко перейти по любой гиперссылке на соответствующую страницу без прокручивания ее содержимого.



Эти гиперссылки позволяют пользователю переходить со страницы на страницу.

Текущая страница не является гиперссылкой. Это просто вторая страница результатов поиска.

Это здорово, но как мы разобьем результат нашего поиска на такие группы? В результате выполнения запроса мы получили все записи вместе.

Разбиение результата запроса на группы из нескольких записей в каждой дает возможность выводить каждую группу на своей странице.

Нам необходим такой запрос, в результате выполнения которого выйдет набор частичных результатов, а не один общий результат.

К счастью, SQL предоставляет способ осуществлять такие вещи с помощью выражения `LIMIT`. Давайте обратимся к нему и посмотрим, как мы сможем его использовать для того, чтобы разбить результат на группы, состоящие из пяти записей каждая...

Извлекайте столько записей, сколько вам необходимо, используя запрос с выражением LIMIT

Решение проблемы управления количеством записей, демонстрируемых на странице, заключается в добавлении к поисковому запросу еще одного выражения, начинающегося с ключевого слова `LIMIT`. Для ограничения количества извлекаемых записей пятью мы добавляем выражение `LIMIT 5` в конец нашего запроса, как показано ниже:

```
SELECT * FROM riskyjobs
  ORDER BY job_title
  LIMIT 5
```

Без условного выражения WHERE в этом запросе будут извлечены все записи из таблицы, что равносильно поиску без каких-либо критериев.

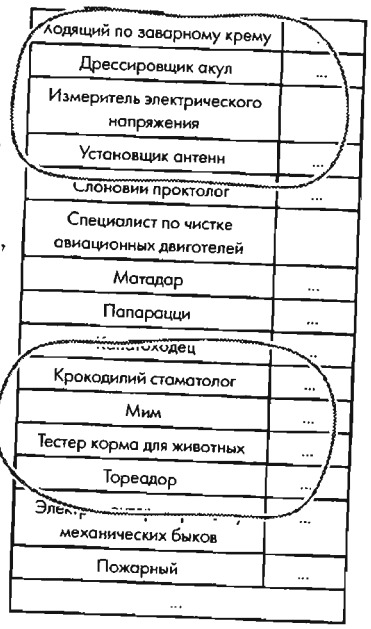
Будут извлечены только первые пять записей, независимо от того, сколько их соответствует условиям запроса.

Выражение с ключевым словом LIMIT управляет количеством записей, извлекаемых в результате выполнения SQL-запроса.

Если вы помните, мы использовали функцию `build_query()` для того, чтобы создать наш поисковый запрос в приложении «Рискованные работы». Для того чтобы извлекать только первые пять записей, мы просто добавим выражение `LIMIT 5` в конец строки запроса после того, как она будет создана:

```
$query = build_query($user_search, $sort);
$query = $query . " LIMIT 5";
```

Добавление выражения LIMIT в конец строки запроса ограничивает количество извлекаемых в результате выполнения запроса записей пятью.



Это работает хорошо для извлечения первых пяти записей, но как нам получить следующие пять, а потом следующие пять после этих пяти? Для извлечения последующих групп записей мы должны немного изменить наше выражение `LIMIT`. Но как? Если добавить `LIMIT 10`, будут извлечены первые 10 записей, а это не то, что нам нужно. Мы должны извлечь записи с шестой по десятую и, чтобы достичь этого, используем немного другой синтаксис выражения `LIMIT`. Если вы укажете два аргумента в выражении `LIMIT`, тогда первый из них определяет количество записей, которые будут пропущены, а второй — количество записей, которые будут извлечены. Например, вот так вы можете извлечь записи с 11-й по 15-ю, которые должны быть выведены на третьей странице результата поиска.

```
$query = build_query($user_search, $sort);
$query = $query . " LIMIT 10, 5";
```

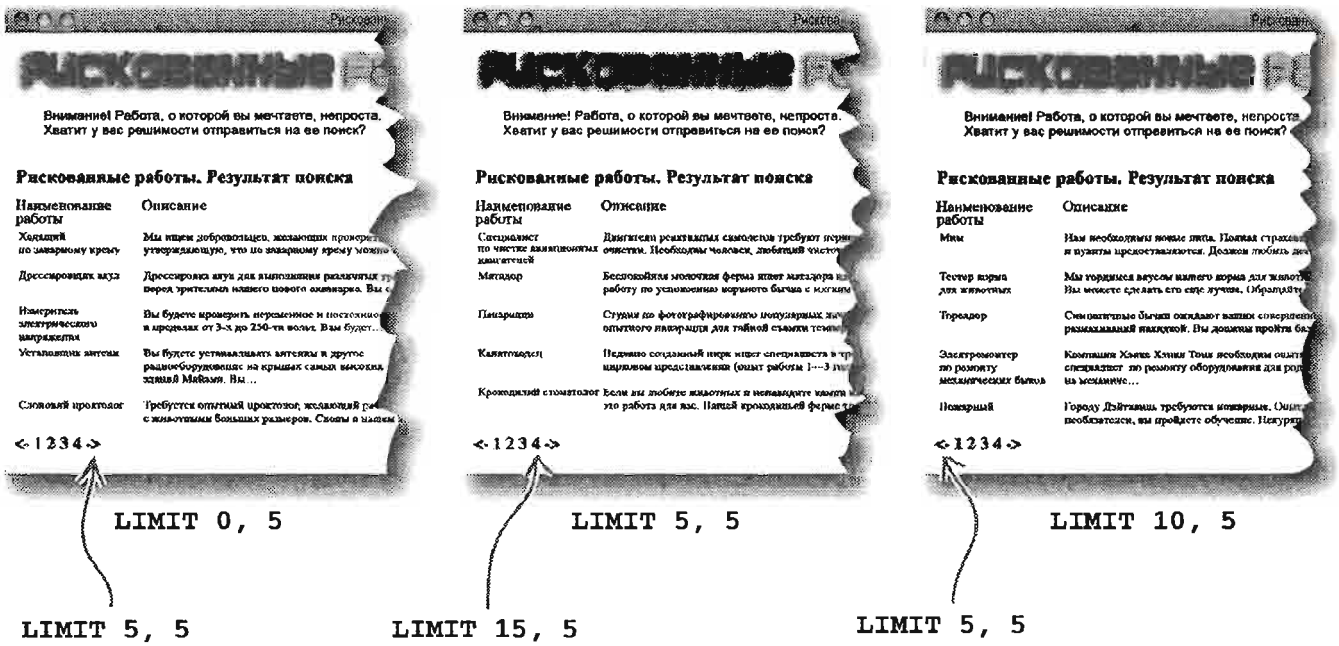
Первый аргумент говорит, сколько записей должно быть пропущено — первых десять.

Второй аргумент управляет количеством извлекаемых записей (пять, так же как и в предыдущий раз).

Управляйте гиперссылками на страницы, используя выражение LIMIT

Важной частью процесса разбиения текста на группы для вывода на отдельных страницах является создание навигационных гиперссылок в нижней части каждой страницы, позволяющих пользователю перемещаться вперед и назад между страницами. Мы можем использовать выражение `LIMIT` для определения параметров этих гиперссылок. Например, гиперссылки «следующая страница» (>>) и «предыдущая страница» (<<) имеют свое собственное выражение `LIMIT`. То же самое можно сказать о гиперссылках в виде номеров страниц: с их помощью пользователь переходит на конкретные страницы результата поиска.

Ниже приведены выражения `LIMIT` для первых трех страниц вместе с выражениями для некоторых гиперссылок на страницы.



Не вижу никаких проблем. Нам просто необходимо написать несколько запросов с различными выражениями LIMIT, правильно?

Что-то вроде этого. Нам необходимо добавлять к запросу выражения LIMIT с разными параметрами в зависимости от страницы и гиперссылки, но мы сможем генерировать их динамически вместо того, чтобы писать множество запросов вручную.

Все, что нам необходимо, — это еще немного модифицировать функцию `build_quiry()` так, чтобы в конце каждой строки запроса, которая в ней создается, добавлялось правильное выражение `LIMIT`.

Отслеживайте данные, необходимые для разбиения результатов поиска на отдельные страницы

Для того чтобы добавить в функцию `build_quiry()` возможность разбиения результатов поиска на отдельные страницы, нам необходимо создать и отслеживать несколько переменных, которые будут управлять созданием строки запроса вывода информации, связанной с текущей страницей, а также навигационных гиперссылок в нижней части страницы.

`$cur_page`

Извлеките номер текущей страницы из URL-сценария с использованием суперглобального массива `$_GET`. Если в URL не содержится данных о номере текущей страницы, присвойте этой переменной значение 1 (первая страница).

`$results_per_page`

В этой переменной содержится количество записей на одной странице, которое вы выбираете исходя из внешнего вида страницы на экране монитора, в том числе с учетом количества записей, помещающихся на экране без необходимости прокручивания. Эта переменная будет определять второй аргумент в выражении `LIMIT`.

`$skip`

Содержит количество записей, которые необходимо пропустить, прежде чем начать извлекать их для текущей страницы. Эта переменная управляет позицией начала вывода информации на странице и определяет первый аргумент в выражении `LIMIT`.

`$total`

Выполните запрос без выражения `LIMIT`, в результате чего получите общее количество записей, которое сохраните в переменной `$total`. Иначе говоря, в этой переменной содержится количество всех объявлений на всех страницах.

`$num_pages`

Определите общее количество страниц, равное общему количеству записей в результате поиска (`$total`), деленному на количество записей на странице (`$results_per_page`), и сохраните его в переменной `$num_pages`. Таким образом, в результате любого поиска будет получено общее количество записей, равное значению переменной `$total`. Эти записи будут выводиться по одной странице за раз с количеством записей на странице, равным значению переменной `$results_per_page`. Всего будет `$num_pages` страниц, и текущая страница будет иметь номер `$cur_page`.

Создание переменных, необходимых для разбиения результатов поиска на страницы

Большинство из значений могут быть присвоены переменным, необходимым для разбиения текста результатов поиска на страницы, исходя из данных, имеющихся в URL страницы. Эти данные доступны через суперглобальный массив `$_GET`. Например, значения переменных `$sort`, `$user_search` и `$cur_page` передаются сценарию непосредственно в составе URL. Исходя из этих данных, мы можем рассчитать, сколько записей должно быть пропущено (`$skip`), чтобы достичь первой записи для этой страницы. Переменная `$results_per_page` немного отличается от всех предыдущих тем, но это, скорее, вопрос личных предпочтений, и мы просто устанавливаем количество объявлений, которые хотим одновременно видеть на одной странице, исходя из расположения элементов изображения на экране монитора.

Извлечение идентификатора вида сортировки, который является целым числом в пределах от 1 до 6 включительно.

Извлечение номера текущей страницы (`$cur_page`) из URL через суперглобальный массив `$_GET`. Если номер текущей страницы не указан, принять его равным 1.

```
// Извлечение идентификатора вида сортировки и поисковой строки
// из URL с помощью суперглобального массива $_GET
$sort = $_GET['sort'];

$user_search = $_GET['usersearch'];

// Расчет данных, необходимых для разбиения
// текста результатов поиска на страницы

$cur_page = isset($_GET['page']) ? $_GET['page'] : 1;

$results_per_page = 5;  // количество объявлений на странице

$skip = (($cur_page - 1) * $results_per_page);
```

Извлечение поисковой строки, которую пользователь ввел в форму поиска.

Если номер страницы не указан, значение по умолчанию — 1 (первая страница).

Установка значения количества объявлений на страницу.

Вычисление номера первой записи, с которой будет начат вывод объявлений на этой странице.

Результатом этого вычисления будет 0 для первой страницы, 5 для второй страницы, 10 для третьей страницы и т. д.

У нас все еще не установлены значения пары важных переменных — `$total` и `$num_pages`. Они могут быть заданы после выполнения первоначального запроса, в результате которого будет определено количество всех записей, соответствующих критериям поиска. Как только мы определим их количество, у нас появится возможность присвоить значения этим переменным и приступить к постраничному извлечению данных...

Внесение в запрос изменений для разбиения результатов поиска на страницы

Теперь, после того как мы установили значения переменных, нам необходимо пересмотреть сценарий поиска так, чтобы вместо всех записей, соответствующих критериям поиска, в запросе вызывались только те из них, которые необходимы для страницы, просматриваемой пользователем в настоящий момент. Для этого требуется предварительно сделать запрос для установки значения переменной `$total` и вычисления значения переменной `$num_pages`. Далее мы можем продолжать делать запросы с целью извлечения данных для конкретных страниц. Для этого нам необходимо будет добавлять к каждому из этих запросов выражение `LIMIT` с использованием переменных `$skip` и `$results_per_page`. Ниже приведен фрагмент сценария `search.php` с изменениями, выделенными серым фоном:

В результате выполнения этого запроса (без выражения LIMIT) извлекаются все записи, соответствующие критериям поиска.

Функция mysqli_num_rows() возвращает количество всех записей, извлеченных в результате выполнения запроса.

```
  // Строка запроса для извлечения всех записей,
  // соответствующих критериям поиска
$query = build_query($user_search, $sort);

$result = mysqli_query($dbc, $query);
$total = mysqli_num_rows($result);
$num_pages = ceil($total / $results_per_page);

  // Строка запроса для извлечения записей
  // только для текущей страницы
$query = $query . " LIMIT $skip, $results_per_page";

$result = mysqli_query($dbc, $query);

while ($row = mysqli_fetch_array($result)) {
  echo '<tr class="results">';
  echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
  echo '<td valign="top" width="50%">' . substr($row['description'], 0, 100) . '...</td>';
  echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
  echo '<td valign="top" width="20%">' . substr($row['date_posted'], 0, 10) . '</td>';
  echo '</tr>';
}

echo '</table>';
```

Сохранение общего количества записей, соответствующих критериям поиска, вызовом функции mysqli_num_rows().

Вычисление количества страниц, делением общего количества записей, соответствующих критериям поиска, на количество объявлений на страницу. Если результат не является целым числом, он увеличивается до ближайшего целого.

Функция ceil() (здесь) округляет число до ближайшего большего целого.

Пропустить столько записей...

...и извлечь столько записей.

Выполнение следующего запроса, но на этот раз с выражением LIMIT, что ограничивает количество извлекаемых записей только теми, которые необходимы для текущей страницы.

Создание навигационных гиперссылок

Итак, мы установили значения нужных нам переменных и создали SQL-запрос, в результате выполнения которого будут извлекаться только те записи, которые нужны нам для текущей страницы. Все, что нам осталось сделать, — это создать в нижней части каждой страницы навигационные гиперссылки, позволяющие пользователю перемещаться вперед и назад между страницами: гиперссылки «следующая страница» (>>) и «предыдущая страница» (<<), а также гиперссылки в виде номеров страниц, с помощью которых пользователь переходит на конкретные страницы результата поиска. У нас уже есть вся информация, необходимая для их создания. Давайте взглянем на нее еще раз, чтобы понять, как мы будем использовать ее для достижения нашей цели.

`$user_search`

Гиперссылка каждой страницы должна знать, что именно пользователь ищет, поэтому мы должны включить критерии поиска в состав URL.

`$cur_page`

Навигационные гиперссылки полностью зависят от текущей страницы, поэтому очень важно также включать их значения в состав URL.

`$num_pages`

Мы должны знать общее количество страниц, чтобы сгенерировать гиперссылку для каждой из них.

`$sort`

Вид сортировки также играет важную роль при разбиении результата поиска на странице и должен быть учтен в запросе.

Итак, мы знаем, какая информация нам необходима для создания навигационных ссылок на каждой из страниц, и поэтому готовы приступить к модификации PHP-кода, чтобы воплотить все это в жизнь. Этот код может быть просто помещен в сценарий `search.php`, но что если мы создадим для него отдельную пользовательскую функцию? В этом случае основной код, который генерирует результат поиска, может получиться значительно проще, так как в нем потребуется ввести всего одну строку для создания навигационных гиперссылок — строку вызова новой пользовательской функции, которую мы можем назвать `generate_page_links()`.

Единственное условие, которое нам необходимо выполнить, заключается в том, что мы не должны вызывать эту функцию, если весь результат поиска помещается на одну страницу. Значит, нам необходимо проверить это условие, перед тем как вызывать функцию `generate_page_links()`. Ниже показано, как мы можем выполнить такую проверку и вызвать функцию, передав ей все необходимые данные в виде аргументов:

```
if ($num_pages > 1) {
  echo generate_page_links($user_search, $sort, $cur_page, $num_pages);
}
```

Вначале производится проверка того, что общее количество страниц с результатами поиска больше 1. В противном случае навигационные гиперссылки не создаются.

Передача значений критериев поиска, вида сортировки, номера текущей строки и общего количества строк для использования их в процессе создания навигационных гиперссылок.

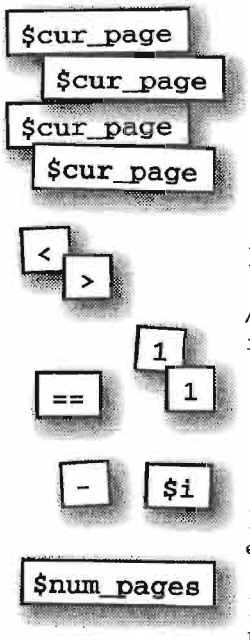
Магниты PHP и MySQL

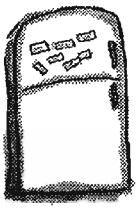
Создание функции `generate_page_links()` почти закончено, но часть кода утрачена. Используя магниты, вставьте недостающий код и дайте приложению «Рискованные работы» возможность создавать навигационные гиперссылки на страницах результатов поиска.

```
// Эта функция создает навигационные гиперссылки на странице результатов поиска,
// основываясь на значениях номера текущей страницы и общего количества страниц

function generate_page_links($user_search, $sort, $cur_page, $num_pages) {
  $page_links = '';
  // Если это не первая страница – создание гиперссылки «предыдущая страница» (<<)
  if (.....) {

    $page_links .= '<a href="' . $_SERVER['PHP_SELF'] .
      '?usersearch=' . $user_search .
      '&sort=' . $sort .

      '&page=' . (.....) . '"><-</a> ';

  }
  else {
    $page_links .= '<- ';
  }

  // Прохождение в цикле всех страниц и создание гиперссылок,
  // указывающих на конкретные страницы

  for ($i = 1; $i <= $num_pages; $i++) {
    if (.....) {

      $page_links .= ' ' . $i;
    }
    else {
      $page_links .= ' <a href="' . $_SERVER['PHP_SELF'] .
        '?usersearch=' . $user_search .
        '&sort=' . $sort .
        '&page=' . $i . '"> ' . $i . '</a>';
    }
  }

  // Если это не последняя страница – создание гиперссылки «следующая страница» (>>)
  if (.....) {

    $page_links .= ' <a href="' . $_SERVER['PHP_SELF'] .
      '?usersearch=' . $user_search .
      '&sort=' . $sort .
      '&page=' . ($cur_page + 1) . '">></a>';
  }
  else {
    $page_links .= ' >';
  }
  return $page_links;
}
```

Гиперссылка «предыдущая страница» выводится на экран в виде символа «<<».

Гиперссылка «следующая страница» выводится на экран в виде символа «>>».





Магниты PHP и MySQL. Решение

Создание функции `generate_page_links()` почти закончено, но часть кода утрачена. Используя магниты, вставьте недостающий код и дайте приложению «Рискованные работы» возможность создавать навигационные гиперссылки на страницах результатов поиска.

```
// Эта функция создает навигационные гиперссылки на странице результатов поиска,
// основываясь на значениях номера текущей страницы и общего количества страниц

function generate_page_links($user_search, $sort, $cur_page, $num_pages) {
  $page_links = '';
  // Если это не первая страница — создание гиперссылки «предыдущая страница» (<<)
  if ( $cur_page > 1 ) {

    $page_links .= '<a href="' . $_SERVER['PHP_SELF'] .
      '?usersearch=' . $user_search .
      '&sort=' . $sort .

      '&page=' . ( $cur_page - 1 ) . '"><-</a> ';

  }
  else {
    $page_links .= '<- ';
  }

  // Прохождение в цикле всех страниц и создание гиперссылок,
  // указывающих на конкретные страницы

  for ($i = 1; $i <= $num_pages; $i++) {
    if ( $cur_page == $i ) {

      $page_links .= ' '   $i;
    }
    else {
      $page_links .= ' <a href="' . $_SERVER['PHP_SELF'] .
        '?usersearch=' . $user_search .
        '&sort=' . $sort .
        '&page=' . $i . '"> ' . $i . '</a>';
    }
  }

  // Если это не последняя страница — создание гиперссылки «следующая страница» (>>)
  if ( $cur_page < $num_pages ) {

    $page_links .= ' <a href="' . $_SERVER['PHP_SELF'] .
      '?usersearch=' . $user_search .
      '&sort=' . $sort .
      '&page=' . ($cur_page + 1) . '">-></a>';
  }
  else {
    $page_links .= ' ->';
  }
  return $page_links;
}
```

Мы все время должны передавать критерии поиска, введенные пользователем, и вид сортировки для URL каждой гиперссылки.

Гиперссылка «предыдущая страница» выводится на экран в виде символа «<<».

Убедитесь, что каждая гиперссылка указывает на один и тот же сценарий. Мы просто добавляем разные страницы в URL каждой гиперссылки.

Имя гиперссылки, указывающей на конкретную страницу, — это просто номер этой страницы.

Гиперссылка «следующая страница» выводится на экран в виде символа «>>».

Сборка законченного сценария поиска

И вот мы подошли к законченному сценарию поиска приложения «Рискованные работы», который показывает результаты, основанные на критериях, введенных пользователем, генерирует заголовки таблицы результатов в виде гиперссылок (щелкая на которых пользователь задает вид сортировки результатов поиска) и, наконец, создает навигационные гиперссылки в нижней части страницы.

```
<?php
  // Эта функция создает строку поискового запроса, используя для этого критерии поиска
     и вид сортировки
  function build_query($user_search, $sort) {
    ...

    return $search_query;
  }

  // Эта функция создает заголовки таблицы результатов поиска в виде гиперссылок,
  // щелкая кнопкой мыши по которым пользователь задает вид сортировки результатов поиска

  function generate_sort_links($user_search, $sort) {
    ...

    return $sort_links;
  }

  // Эта функция создает навигационные гиперссылки на странице результатов поиска,
  // основываясь на значениях номера текущей страницы и общего количества страниц

  function generate_page_links($user_search, $sort, $cur_page, $num_pages) {
    ...

    return $page_links;
  }

    // Извлечение идентификатора типа сортировки и поисковой строки из URL
       с помощью суперглобального массива $_GET
  $sort = $_GET['sort'];
  $user_search = $_GET['usersearch'];

  // Расчет данных, необходимых для разбиения текста результатов поиска на страницы
  $cur_page = isset($_GET['page']) ? $_GET['page'] : 1;
  $results_per_page = 5;  // количество объявлений на странице
  $skip = (($cur_page - 1) * $results_per_page);

  // Создание таблицы с результатами поиска
  echo '<table border="0" cellpadding="2">';

  // Вывод заголовков таблицы результатов поиска
  echo '<tr class="heading">';
  echo generate_sort_links($user_search, $sort);
  echo '</tr>';
```

Мы уже создали эти три функции, поэтому нет никакой необходимости переписывать один и тот же код.

Извлечение вида сортировки и критериев поиска из URL через суперглобальный массив $_GET.

Инициализация переменных, необходимых для разбиения результатов поиска и вывода их на отдельных страницах. Понадобятся нам при создании выражений LIMIT и навигационных гиперссылок.

Вызов функции generate_sort_links() для создания заголовков таблицы результатов поиска в виде гиперссылок и вывод их с помощью команды echo.

search.php

Не спешите, есть еще код!

Окончательный вариант сценария поиска, продолжение...

```
// Соединение с базой данных
require_once('connectvars.php');
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

// Выполнение запроса для извлечения всех записей
$query = build_query($user_search, $sort);
$result = mysqli_query($dbc, $query);
$total = mysqli_num_rows($result);
$num_pages = ceil($total / $results_per_page);

// Выполнение запроса для извлечения записей для одной страницы
$query =  $query . " LIMIT $skip, $results_per_page";
$result = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($result)) {
  echo '<tr class="results">';
  echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
  echo '<td valign="top" width="50%">' . substr($row['description'], 0, 100) . '...</td>';
  echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
  echo '<td valign="top" width="20%">' . substr($row['date_posted'], 0, 10) . '</td>';
  echo '</tr>';
}
echo '</table>';

// Если вся информация не помещается на одной странице — создание навигационных гиперссылок
if ($num_pages > 1) {
  echo generate_page_links($user_search, $sort, $cur_page, $num_pages);
}

mysqli_close($dbc);
?>
```

Вызов функции build_query() для создания поискового запроса.

Это выражение LIMIT, которое мы создали для того, чтобы извлекать данные только для одной страницы.

А этот код мы создали для того, чтобы, используя функцию substr(), укоротить строки описания работ и дат регистрации объявлений.

Вызов функции generate_sort_links() для создания заголовков таблицы результатов поиска в виде гиперссылок и вывод их с помощью команды echo.

Закрытие базы данных для наведения порядка после окончания работы.

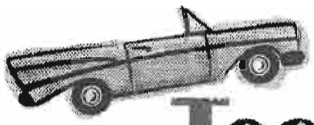
Не бывает глупых вопросов

В: Мы действительно должны передавать функции generate_page_links() критерии поиска, вид сортировки и данные, необходимые для постраничного вывода информации?

О: Да. И причина этого заключается в том, что грамотно разработанная функция не должна манипулировать необходимыми ей данными, находящимися за пределами блока ее кода. Функция должна иметь доступ только к тем данным, которые переданы ей в качестве аргументов, и изменять только свои собственные данные, в том числе те, которые она будет возвращать вызвавшему ее коду.

В: Хорошо, а что по поводу вывода данных с помощью команды echo? Почему функция generate_page_links() просто не выводит гиперссылки с помощью команд echo, а возвращает их в виде строки?

О: По той же самой причине. Выводя гиперссылки с помощью команд echo, функция фактически выполняет действия вне своих пределов. Значительно труднее искать ошибки и проводить изменения в ситуациях, когда не совсем ясно, какие данные изменяются. Необходимо всегда обрабатывать данные «на месте» и возвращать их вызвавшему коду, чтобы он использовал эти данные по назначению за пределами функции.

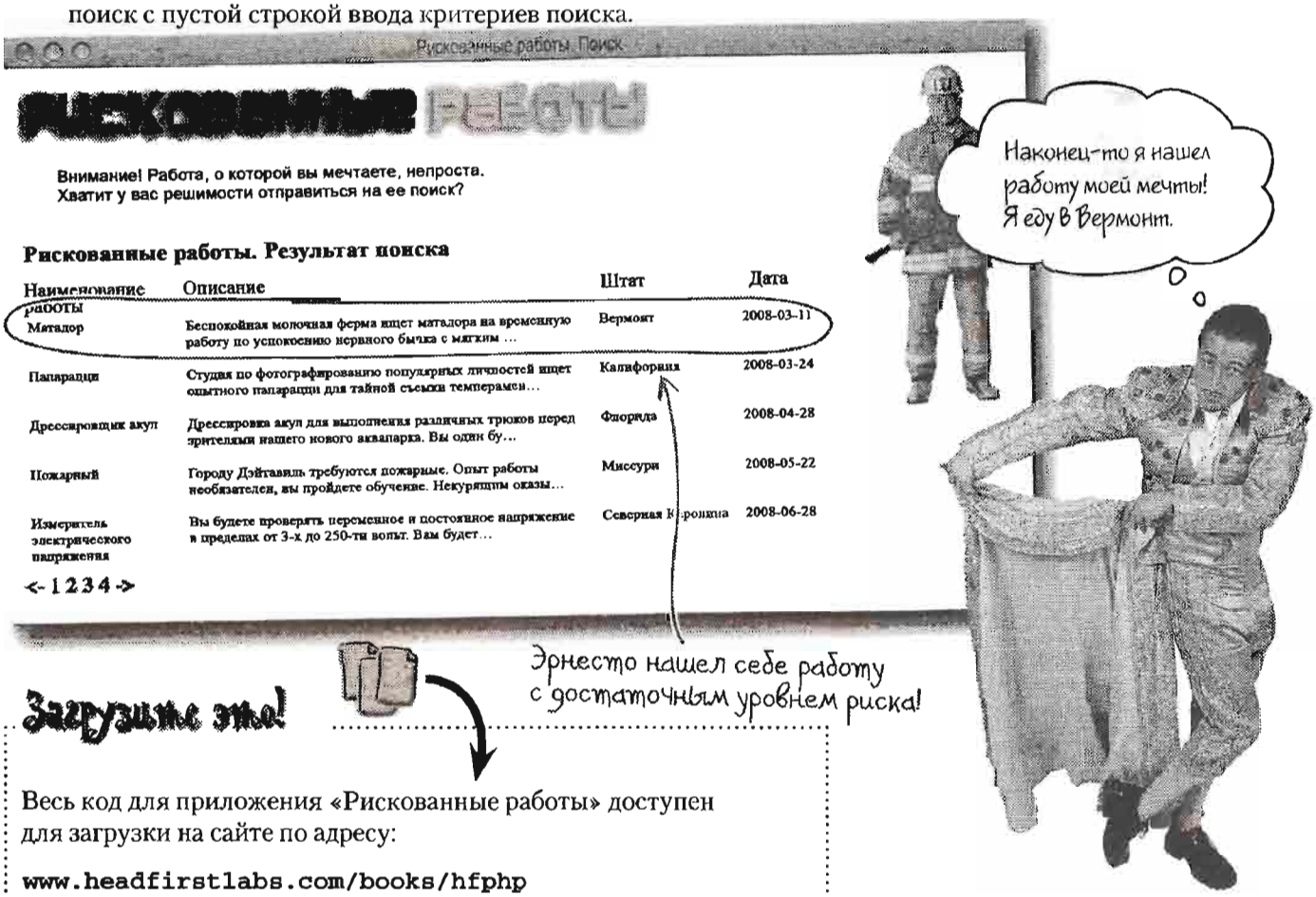


Тест-драйв

Закончите сценарий поиска приложения «Рискованные работы».

Добавьте новую функцию `generate_page_links()` в сценарий `search.php`, не забыв внести код, который будет вызывать эту функцию только после проверки того, что данные результатов поиска не помещаются на одну страницу. Создайте и инициализируйте также переменные, которые будут переданы функции в качестве аргументов. И не забудьте изменить код, создающий поисковый запрос, чтобы в нем использовалось выражение `LIMIT` для извлечения тех записей, которые необходимы для текущей страницы.

После окончания всех этих корректировок загрузите сценарий и остальные файлы приложения «Рискованные работы» на ваш веб-сервер и откройте форму поиска `search.html` в браузере. Попробуйте задать несколько различных критериев поиска, чтобы увидеть, как код сценария создает запросы. Для проверки работоспособности системы разбиения данных на страницы задайте такие критерии поиска, при которых будет получен достаточно объемный результат и для его вывода потребуется несколько страниц. Для того чтобы получить максимально объемный результат, задайте поиск с пустой строкой ввода критериев поиска.



Загрузите это!

Весь код для приложения «Рискованные работы» доступен для загрузки на сайте по адресу:

`www.headfirstlabs.com/books/hfphp`

Ваш инструментарий PHP и MySQL

Сценарий поиска приложения «Рискованные работы» потребовал использования еще нескольких новых приемов, доступных в PHP и MySQL. Давайте сделаем резюме.

LIKE

Используется в SQL-запросах для извлечения данных с приблизительным соответствием заданному критерию. Наличие группового символа % в начале и/или в конце текста поиска говорит о том, что перед и/или после указанного текста поиска могут быть любые символы в любом количестве.

explode(), implode()

Функция PHP explode() разбивает строку с обычными разделителями, такими как символ пробела или запятая, на несколько подстрок, которые сохраняет как элементы массива. Функция PHP implode() выполняет обратную задачу: создает единую строку из элементов массива, вставляя между ними заданный разделитель.

str_replace()

Вызывайте эту функцию для выполнения задачи поиска и замены в строке текста, заменяя один или группу нескольких символов другим символом или группой нескольких символов.

switch-case

Управляющая конструкция PHP, которая позволяет вам выполнить один из нескольких блоков кода в зависимости от значения переменной. Если вы столкнетесь с ситуацией, вынуждающей вас использовать множество вложенных управляющих конструкций if-else, возможно, вы найдете использование управляющей конструкции if-else более эффективным.

substr(

Извлекае
основыва
передан
Вы може
начальн
ее конеч
какой-л
из сере

Cust

Блок P
органи
в виде
многок
пакет
заклю
кода, В
опреде
он мог
многок
с мини
усилий

LIMIT

Позвол
контр
количе
извлеч
выпол
Но эт
Выраж
пропу
колич
для и
опред

10 Регулярные выражения

Правила замены

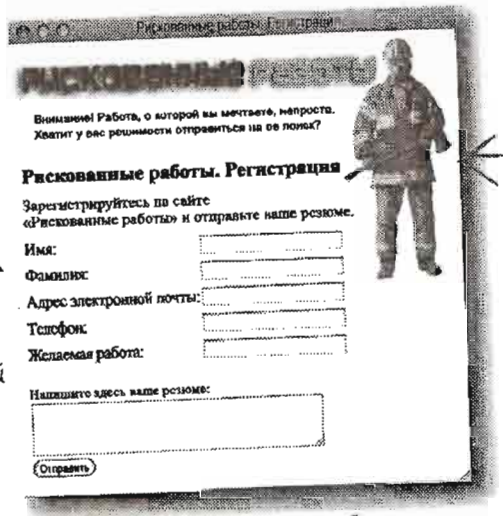
Функции для обработки строк — похоже, очень неплохое средство, но в то же время они достаточно ограничены в своих возможностях. Конечно, они могут определить длину вашей строки, укоротить ее и заменить отдельные символы другими. Но иногда у вас возникает необходимость делать более сложные преобразования текстовых строк. Здесь вам на помощь приходят **регулярные выражения**. Используя их, вы можете выполнять достаточно сложные преобразования текстовых строк, основываясь на **группе правил**, а не на каком-то одном критерии.

Приложение «Рискованные работы» дает пользователям возможность загружать свои резюме

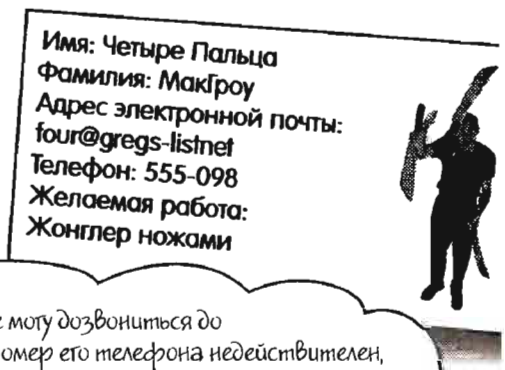
Сайт `RiskyJobs.biz` разросся. Теперь компания дает пользователям возможность вводить свои резюме и контактную информацию в веб-форму, чтобы нанимателям было легче их разыскать. Ниже показано, как выглядит эта форма.

Новая регистрационная форма приложения «Рискованные работы» дает претенденту на получение рискованной работы возможность ввести информацию о себе, чтобы потенциальные наниматели смогли легко его разыскать.

В дополнение к обычной контактной информации претендент на получение рискованной работы должен также ввести вид желаемой работы и свое резюме.



Информация о наших соискателях рискованных работ сохраняется в таблице, данные которой доступны для поиска новых потенциальных работников нанимателями, охотниками за головами и другими специалистами по подбору персонала. Но здесь возникает проблема... Данным, введенным в форму, по всей видимости, полностью доверять нельзя!



Прежде всего, я не могу дозвониться до ниндзя, потому что номер его телефона недействителен, а на мое электронное письмо жонглеру ножами в ответ пришло сообщение об ошибке с почтового сервера. Похоже на то, что в контактной информации этих пользователей имеются ошибки.

Наниматели могут производить поиск в базе данных соискателей и затем связываться с ними для решения вопроса о возможном приеме на работу... при условии, что имеющаяся контактная информация верна!

Ниже приведен код для сценария `registration.php`, который выводит и обрабатывает данные, внесенные пользователем в форму регистрации нового соискателя рискованной работы. Напишите, что, по вашему мнению, в этом коде неправильно и как он может быть изменен для того, чтобы решить проблему плохих данных.

```
<?php
  if (isset($_POST['submit'])) {
    $first_name = $_POST['firstname'];
    $last_name = $_POST['lastname'];
    $email = $_POST['email'];
    $phone = $_POST['phone'];
    $job = $_POST['job'];
    $resume = $_POST['resume'];
    $output_form = 'no';

    if (empty($first_name)) {
      // Переменная $first_name не определена
      echo '<p class="error">Вы забыли ввести свое имя.</p>';
      $output_form = 'yes';
    }

    if (empty($last_name)) {
      // Переменная $last_name не определена
      echo '<p class="error"> Вы забыли ввести свою фамилию.</p>';
      $output_form = 'yes';
    }

    if (empty($email)) {
      // Переменная $email не определена
      echo '<p class="error"> Вы забыли ввести адрес вашей электронной
почты.</p>';
      $output_form = 'yes';
    }

    if (empty($phone)) {
      // Переменная $phone не определена
      echo '<p class="error"> Вы забыли ввести номер своего телефона.</p>';
      $output_form = 'yes';
    }
    ...  <-- Продолжение проверки ввода данных.
  }
  else {
    $output_form = 'yes';
  }

  if ($output_form == 'yes') {
?>
  <-- Вывод формы.
...
```

Ниже приведен код для сценария `registration.php`, который выводит и обрабатывает данные, внесенные пользователем в форму регистрации нового соискателя рискованной работы. Напишите, что, по вашему мнению, в этом коде неправильно и как он может быть изменен для того, чтобы решить проблему плохих данных.

```
<?php
  if (isset($_POST['submit'])) {
    $first_name = $_POST['firstname'];
    $last_name = $_POST['lastname'];
    $email = $_POST['email'];
    $phone = $_POST['phone'];
    $job = $_POST['job'];
    $resume = $_POST['resume'];
    $output_form = 'no';

    if (empty($first_name)) {
      // Переменная $first_name не определена
      echo '<p class="error">Вы забыли ввести свое имя.</p>';
      $output_form = 'yes';
    }

    if (empty($last_name)) {
      // Переменная $last_name не определена
      echo '<p class="error"> Вы забыли ввести свою фамилию.</p>';
      $output_form = 'yes';
    }

    if (empty($email)) {
      // Переменная $email не определена
      echo '<p class="error"> Вы забыли ввести адрес вашей электронной
почты.</p>';
      $output_form = 'yes';
    }

    if (empty($phone)) {
      // Переменная $phone не определена
      echo '<p class="error"> Вы забыли ввести номер своего телефона.</p>';
      $output_form = 'yes';
    }

  }
  else {
    $output_form = 'yes';
  }

  if ($output_form == 'yes') {
?>
...
```

Сценарий проверяет поля ввода на наличие данных, что само по себе очень хорошо, но некоторые поля требуют ввода более специфичных данных, которые должны соответствовать определенному формату.

Нет ничего, что нужно было бы проверять в полях ввода имени и фамилии, кроме того, что они не должны быть пустыми.

Адрес электронной почты имеет специфический формат, на соответствие которому мы должны проверять адрес, введенный пользователем, прежде чем принять его для дальнейшей обработки.

Четыре Пальца МакГроу не поставил точку в конце адреса электронной почты (между словами gregs-list и net). Сценарий, обрабатывающий данные формы, должен перехватывать подобные ошибки!

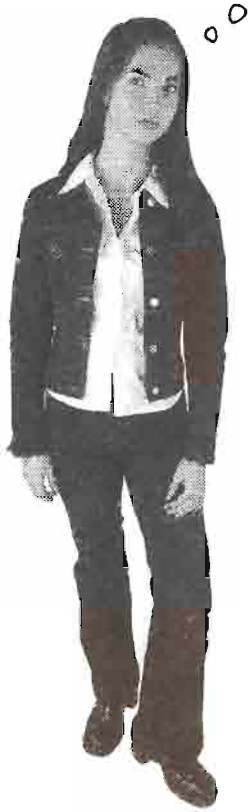
Аналогичная проблема с номером телефона. Дальнейшая обработка данных формы должна производиться только при условии, что номер телефона введен в правильном формате.

Продолжение проверки ввода данных.

Джимми Смит не указал код города во введенном им номере телефона. Сценарий должен обеспечить проверку наличия этих данных.

Вывод формы.

Что нам действительно необходимо, так это способ проверки двух полей ввода формы — адреса электронной почты и номера телефона — на соответствие определенным форматам. Для остальных полей ввода достаточно убедиться только в том, что они не пусты.



Почему бы нам не использовать для корректировки неправильных данных какие-нибудь из строковых функций? Разве мы не сможем использовать функцию str_replace(), чтобы добавить пропущенные данные?

Вы, конечно, можете использовать для корректировки неправильных данных какие-нибудь из строковых функций, но они мало что дадут вам в тех случаях, когда данные должны соответствовать специфическому шаблону.

Строковые функции хорошо подходят для решения задач, сводящихся к простой схеме «поиск и замена». Например, если пользователь при вводе номера своего телефона использовал для отделения блоков цифр друг от друга точки, вы успешно сможете использовать функцию `str_replace()`, чтобы заменить их дефисами.

Но когда мы сталкиваемся с ситуацией, в которой точно видим, что введены неправильные данные, но не можем знать значения правильных данных, как в случае с кодом города в номере телефона, нам необходимо попросить человека, вводящего данные, уточнить их. И единственный способ определить, ввел он или не ввел код города, — это проверить на соответствие шаблону телефонного номера. В подобных случаях необходим более совершенный метод проверки того, что такие данные, как номер телефона или адрес электронной почты, введены совершенно правильно.

Понятно, но почему мы все-таки не можем использовать строковые функции, чтобы осуществлять такую проверку?

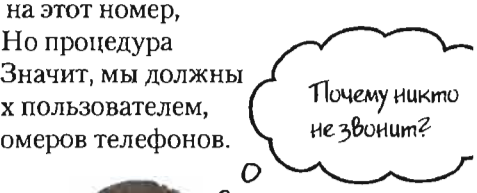
На самом деле строковые функции не способны осуществлять больше, чем примитивную проверку данных.

Подумайте, как бы вы пытались проверить адрес электронной почты, используя строковые функции. В PHP есть функция `strlen()`, которая говорит вам о том, сколько символов содержится в данной строке. Но для данных, подобных адресу электронной почты, не существует ограничений длины строки. Конечно, потенциально это было бы полезно при проверке номера телефона, так как эти номера обычно содержат одинаковое количество цифр, но вам все равно предстоит еще иметь дело с точками, дефисами и скобками.

Говоря об адресах электронной почты, нужно заметить, что их формат просто слишком сложен для анализа с использованием строковых функций. По сути дела, здесь нам необходим анализ последовательности символов, которому требуется алгоритм проверки на соответствие строки определенному шаблону. Подобные шаблоны для вводимых пользователем данных являются центральной частью такого алгоритма.

Определите, как должны выглядеть ваши данные

Наша задача заключается в том, чтобы точно и ясно определить, как должны выглядеть конкретные данные, вплоть до каждого отдельного символа. Рассмотрим номер телефона, введенный Джимми. Для любого человека, взглянувшего на этот номер, становится совершенно очевидным, что в нем не указан код города. Но процедура проверки данных формы выполняется не человеком, а PHP-кодом. Значит, мы должны «научить» наш код, как проанализировать строку данных, введенных пользователем, и определить, соответствует ли она шаблону, установленному для номеров телефонов.



Определение такого шаблона — достаточно сложная задача, и она действительно связана с поиском соответствия не просто значению, а группе значений в определенном диапазоне, допустимом для анализируемых данных конкретного вида. Номер телефона является достаточно простым видом данных, так как он состоит из десяти цифр и каких-либо разделителей. Что касается адреса электронной почты, то это совсем другая история, но мы займемся этими данными немного позднее в этой главе.

Для человека не представляет труда, взглянув на номер телефона Джимми, понять, что не хватает кода города. Но создать PHP-код для подобного «заключения» — задача не из тривиальных.

Не бывает глупых вопросов

В: Я все равно не понимаю, почему для проверки данных нашей формы мы не можем просто ограничиться функциями isset() и empty().

О: Эти две функции скажут вам только, ввел пользователь какие-либо данные в поле формы или нет. Но они ничего не скажут о том, какие именно он ввел данные. Для функции `empty()` нет абсолютно никакой разницы, ввел пользователь в поле ввода нашей формы «Телефон» строку (707) 827-700 или 4FG8SXY12. Но для сайта «Рискованные работы» это стало бы серьезной проблемой, так как в его базе данных должны быть действительные данные для связи с соискателями рискованных работ.

В: Если функции isset() и empty() плохо подходят для проверки данных нашей формы, можем ли мы просто поручить кому-нибудь проверять данные после того, как они поступили в базу?

О: Вы можете это сделать, но тогда часто будут возникать ситуации, когда будет уже поздно что-либо исправлять. Если, например, в номере телефона не указан код города, нам необходимо просить пользователя уточнить данные путем повторного заполнения формы. Если вы заметите это после того, как данные будут занесены в базу, у вас может не быть никакой контактной информации, чтобы сообщить пользователю, что некоторые из его данных содержат ошибки. А так как пользователь, скорее всего, и не подозревает, что он совершил ошибку, он просто не будет знать, что у него имеются проблемы с данными. Поэтому наилучший алгоритм проверки предполагает, что данные должны проверяться на достоверность в процессе их обработки при поступлении на сервер.

В: Так как же тогда мы будем определять, действительны введенные пользователем данные или нет?

О: Все зависит от вида данных. Различные виды данных требуют различных правил, по которым будет осуществляться их проверка. Все зависит от того, какие символы содержатся в строке, каков порядок их расположения. И вы должны реализовать эти правила в PHP-коде. Давайте познакомимся поподробнее с правилами, относящимися к номерам телефонов…

Время поработать

Напишите все возможные способы, которыми, по вашему мнению, можно представить номера телефонов.

Какие правила должны соблюдать пользователи при вводе данных в форму? Например, номера телефонов не должны включать букв.

Вы можете начать с такого правила.

Мы можем требовать, чтобы строка, содержащая номер телефона, состояла только из цифр, причем все десять цифр должны следовать подряд.

Решение задачи

Напишите все возможные способы, которыми, по вашему мнению, можно представить номера телефонов.

В строке, содержащей номер телефона, могут находиться дефисы, правые и левые скобки, точки.

555 636 4652
(555) 636-4652
(555)636-4652
(555) 6364652
555636-4652
555 636-4652
555.636.4652
5556364652
555-636-4652
555 ME NINJA

Можно даже допустить наличие букв в строке, содержащей номер телефона, но это значительно расширит число вариантов строк, которые мы будем рассматривать как номера телефонов.

Какие правила должны соблюдать пользователи при вводе данных в форму? Например, номера телефонов не должны включать букв.

Вы можете начать с такого правила.

Мы можем требовать, чтобы строка, содержащая номер телефона, состояла только из цифр, причем все десять цифр должны следовать подряд.

Мы можем разбить номер телефона на три части и вводить каждую в свое поле формы: одна часть — для кода города, вторая — для первых трех цифр и третья — для последних четырех цифр.

Или мы можем предложить пользователям вводить в форме номер их телефона так, чтобы он соответствовал шаблону: (555) 636-4652. Мы можем сами определять правила.

Здесь слишком много вариантов представления номера телефона. Как мы можем создать шаблон, соответствующий каждому из них?

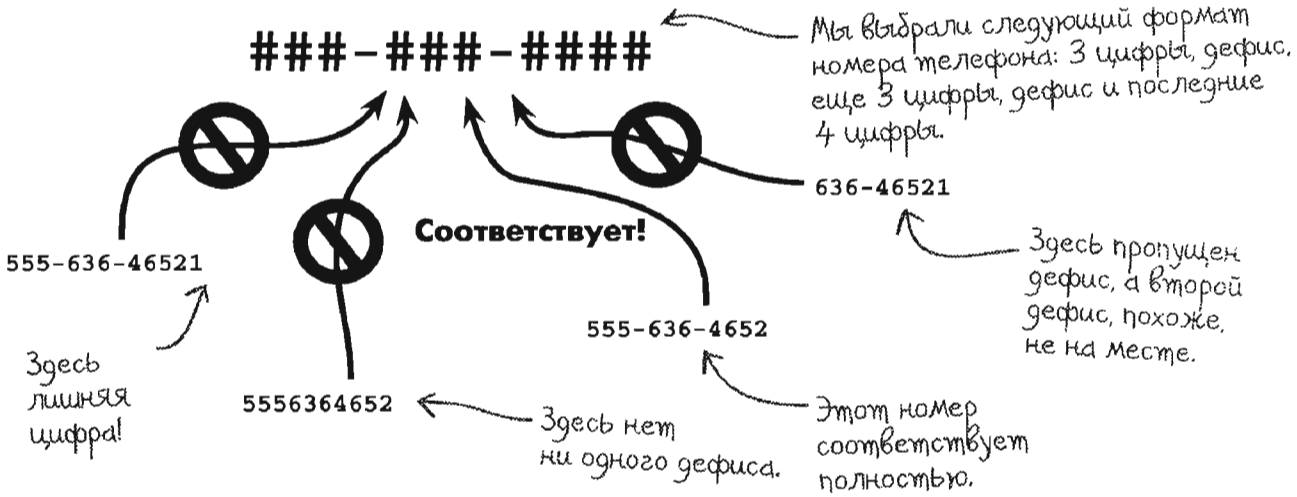
Что касается телефонных номеров, есть некоторые вещи, которые мы знаем наверняка, и мы можем использовать это при определении правил.

Во-первых, телефонный номер не должен начинаться с цифры 0 (с этой цифры начинаются служебные номера). Во-вторых, он должен состоять из десяти цифр. И даже если есть некоторые люди, которые имеют возможность получить номер телефона, состоящий из букв, основные номера телефонов включают только десять цифр (в том числе код города).

Создание шаблона для номера телефона

Для того чтобы двигаться дальше простейших способов проверки, которые предоставляют функции `empty()` и `isset()`, нам необходимо создать шаблон, на соответствие которому мы хотим проверять наши данные. В случае с номером телефона это означает, что мы должны установить единый формат, в котором пользователи будут вводить в форму номера своих телефонов. После того как мы установим такой формат, можем проверять все вводимые номера телефонов на соответствие ему.

Ниже показан наиболее распространенный формат номеров телефонов. Его принятие означает, что если номер телефона, вводимый пользователем в форму, не соответствует ему, PHP-сценарий отвергнет форму и выведет сообщение об ошибке.



не бывает глупых вопросов

В: Должны ли мы всегда использовать такой шаблон для номеров телефонов?

О: Этот шаблон мы используем в приложении «Рискованные работы», потому что соответствующий ему формат даты достаточно распространен. Но при разработке вашей формы вам необходимо выбрать такой шаблон, который будет иметь смысл для вашего конкретного приложения. Только имейте в виду: чем более распространенный формат даты вы выбираете, тем проще будет пользователям иметь дело с вашей формой.

В: Разве мы не можем просто сказать пользователям вводить номера телефонов в формате ##########, а потом, используя строковые функции PHP, проверить, все ли 10 символов являются цифрами?

О: Вы можете, и этого было бы вполне достаточно, если бы все люди пользовались таким форматом номеров телефонов в повседневной жизни. К сожалению, в действительности такой формат не слишком хорош, потому что большинство не используют его при записи номеров телефонов. Для многих он непривычен, и вряд ли все будут ему аккуратно следовать.

В: Но ведь это наш формат, значит, мы можем выбирать такой, какой хотим, разве нет?

О: Конечно, но в то же время вы ведь хотите, чтобы у пользователей оставалось хорошее впечатление от вашего сайта? Иначе они просто перестанут посещать его.

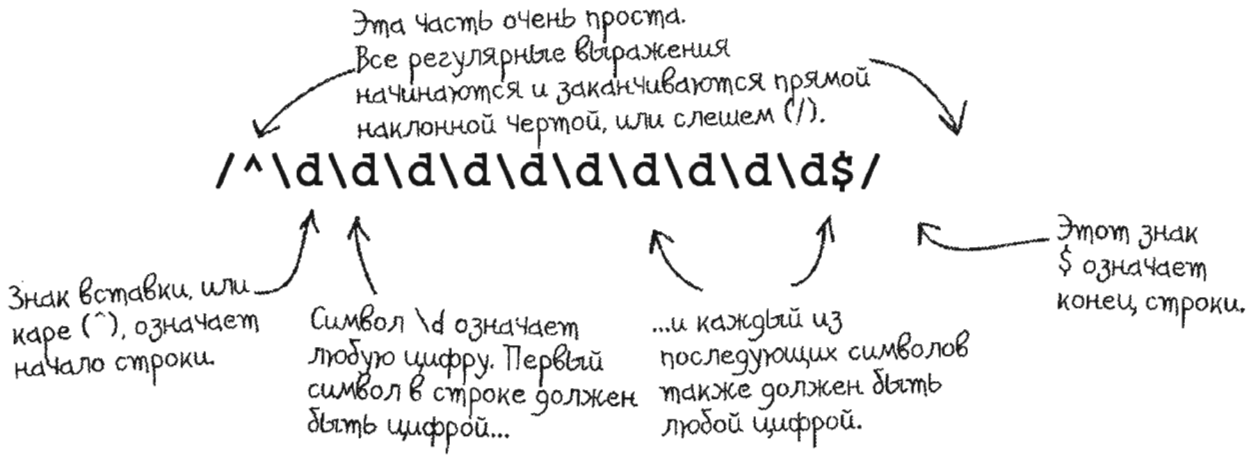
В: Ладно, тогда почему бы нам не использовать три текстовых поля для ввода номера телефона в форму: одно для кода города, второе для первых трех цифр и третье для последних четырех? Потом мы сможем использовать строковые функции PHP.

О: Конечно, вы могли бы так сделать, и на некоторых сайтах используется именно такой метод. Но возможность использования шаблона для сравнения обеспечивает большую гибкость. Кроме того, сравнение с шаблоном оказывается исключительно полезным не только для того, чтобы убедиться, что пользователь правильно ввел номер телефона. Вы увидите это позднее.

Проверка соответствия шаблону с использованием регулярных выражений

PHP предлагает исключительно мощное средство для создания шаблонов и проверки строк на соответствие им. Вы можете создавать правила, которые позволяют вам находить в строках текста соответствие определенным шаблонам. Система анализа текста, основанная на применении таких правил для поиска соответствия заданному шаблону, относится к разделу **регулярных выражений**. Регулярное выражение является шаблоном (набором правил), на соответствие которому мы проверяем строку текста.

В качестве примера ниже приведено регулярное выражение, которому соответствует строка, состоящая из 10 цифр. Этот шаблон будет соответствовать только строке, состоящей ровно из 10 цифр. Если строка короче или длиннее, она не будет соответствовать этому шаблону. Если в строке встречается какой-либо символ, не являющийся цифрой, она также не будет соответствовать этому шаблону. Давайте посмотрим подробнее.



Существует более удобный способ записи этого регулярного выражения, основанный на использовании фигурных скобок. Фигурные скобки применяются для того, чтобы обозначить повторение символа:

```
/^\d{10}$/
```

Это то же самое регулярное выражение, что и показанное выше. Выражение \d{10} — это сокращенный способ указать, что метасимвол \d, стоящий перед числом в фигурных скобках, нужно повторить 10 раз.

Регулярные выражения используются для поиска в строке соответствия заданному шаблону.

Это правда, регулярные выражения не слишком просты для чтения и понимания...
Но они обладают огромными возможностями.

За предоставление широких возможностей приходится платить, и в данном случае эта плата заключается в сложном синтаксисе регулярных выражений. Вы не станете мастером по их использованию за один вечер, но вам это во многих случаях и не нужно. Вы сможете сделать что-нибудь исключительно эффективное и полезное с помощью несложных регулярных выражений, имея самые базовые знания в этой области, особенно когда это касается проверки данных, введенных пользователем в форму. Кроме того, чем больше вы будете иметь с ними дело, тем легче вам будет понимать механизмы их создания и использования.

Создание шаблонов с использованием метасимволов

Иметь возможность находить в строке текста цифры с использованием выражения `\d` очень удобно, но если бы все возможности регулярных выражений сводились только к этому, их использование было бы крайне ограниченным. Проверки на соответствие одним только цифрам для приложения «Рискованные работы» совершенно недостаточно. Нам нужно проводить аналогичные операции с символами пробелов, дефисами и даже буквами.

К счастью, реализация поддержки регулярных выражений в PHP дает возможность использовать еще множество выражений, подобных `\d`, для поиска и других соответствий. Эти выражения называются **метасимволами**. Давайте рассмотрим некоторые наиболее часто используемые в регулярных выражениях метасимволы.

Метасимволы предоставляют нам возможность описывать элементы шаблонов в регулярных выражениях.

\d

Как вы уже знаете, этот метасимвол соответствует цифре — любой цифре от 0 до 9. Имейте в виду: взятый сам по себе, он соответствует только одной цифре, поэтому если вы хотите проверить соответствие двухзначному числу, то должны использовать выражение `\d\d` или `\d{2}`.

\w

Соответствует любому алфавитно-цифровому символу, другими словами, и букве, и цифре. Он будет соответствовать одному символу из следующей последовательности: a—z, A—Z, а—я, А—Я (как прописным, так и строчным), а также 0—9 (как `\d`).

\s

Соответствует пробельному символу. Это не означает, что он соответствует только символу пробела, который появляется на вашем экране, когда вы нажимаете клавишу «Пробел». К пробельным символам относятся также символы горизонтальной и вертикальной табуляции и символ перевода строки. Имейте в виду, что взятый сам по себе, он также соответствует только одному из перечисленных символов, поэтому если вы хотите проверить соответствие двум пробельным символам, то должны использовать выражение `\s\s` или `\s{2}`.

^

Мы видели знак вставки, или каре (^), на предыдущей странице. Этот метасимвол соответствует началу строки, поэтому вы можете использовать его для того, чтобы указать, что соответствие должно быть в начале строки текста, а не в любом другом ее месте. Например, регулярному выражению `/^\d{3}/` будет соответствовать строка «300 приложений», но не будет соответствовать строка «Мы получили 300 приложений».

.

Метасимвол «точка» соответствует любому символу, кроме символа перевода строки. Так же как и метасимвол `\w`, он соответствует и букве, и цифре, а также символу пробела и символу табуляции, как `\s`.

$

Соответствует концу строки. Вы можете использовать его в паре с метасимволом ^, чтобы брать для проверки на соответствие всю строку полностью, от начала до конца. Например, регулярному выражению `/^\w{7}\s\d{3}$/` будет соответствовать строка «Нянечка 411», но не будут соответствовать строки «Нянечка 411 очень хорошая» или «Очень хорошая нянечка 411».

Все эти метасимволы очень удобны, но что делать, если вам понадобится использовать в регулярном выражении обыкновенные символы? Просто вставьте их в свое регулярное выражение. Например, если вы захотите создать регулярное выражение, которому должен соответствовать конкретный номер телефона 707-827-7000, создайте его в виде /707-827-7000/.

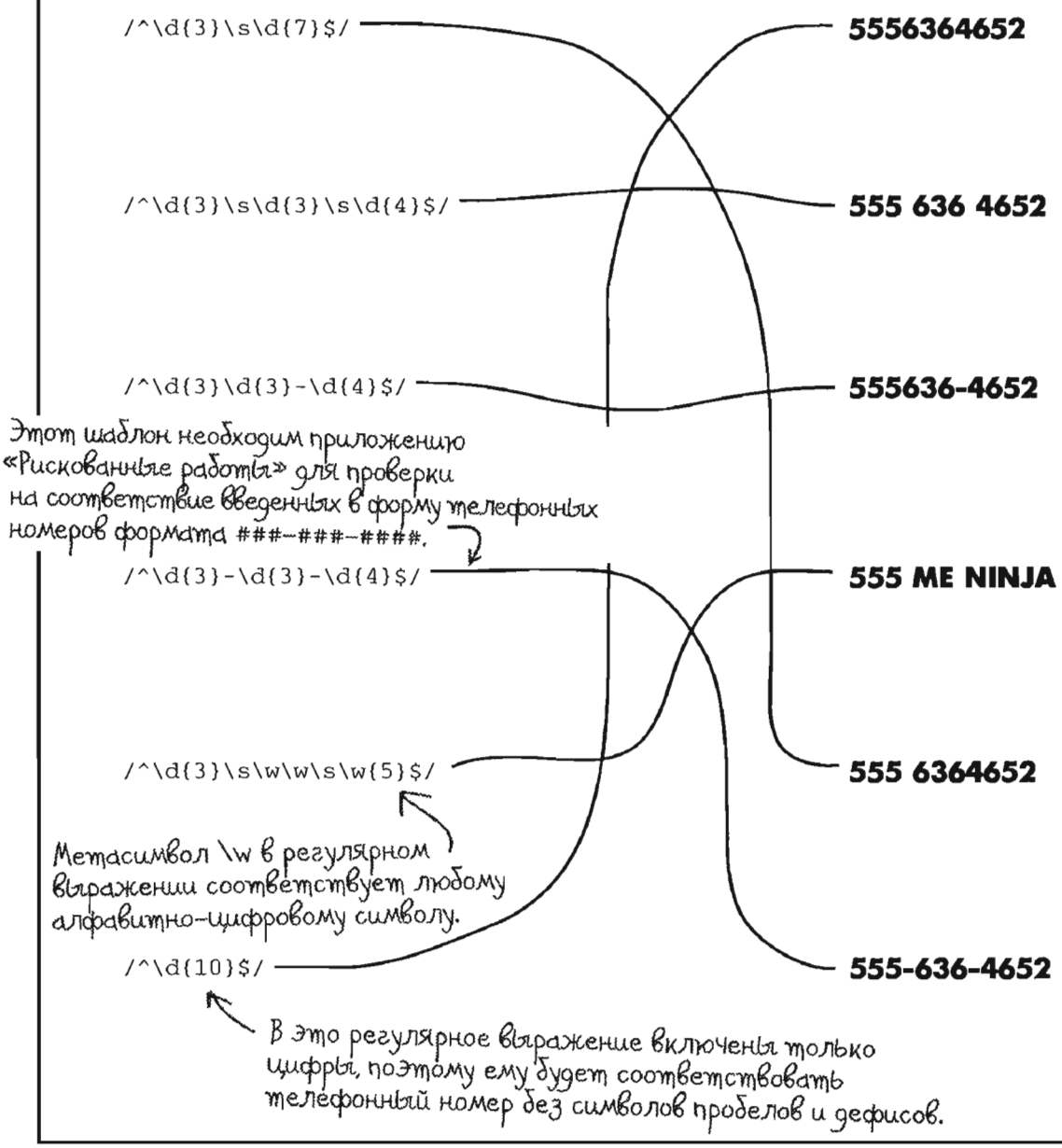
Найдите соответствие между различными регулярными выражениями и конкретными номерами телефонов.

Регулярное выражение	Номер телефона
/^\d{3}\s\d{7}$/	**5556364652**
/^\d{3}\s\d{3}\s\d{4}$/	**555 636 4652**
/^\d{3}\d{3}-\d{4}$/	**555636-4652**
/^\d{3}-\d{3}-\d{4}$/	**555 ME NINJA**
/^\d{3}\s\w\w\s\w{5}$/	**555 6364652**
/^\d{10}$/	**555-636-4652**

ЧТО К ЧЕМУ

РЕШЕНИЕ

Найдите соответствие между различными регулярными выражениями и конкретными номерами телефонов.



Станьте регулярным выражением

Ваша задача — играть роль регулярного выражения: или принимать, или отвергать номера телефонов, вводимые пользователями приложения «Рискованные работы». Нажмите кнопку с независимой фиксацией, расположенную слева от номера телефона, который вы считаете соответствующим вам. В противном случае оставьте кнопку в ненажатом состоянии. Прокомментируйте все случаи несоответствия.

Это регулярное выражение — шаблон номера телефона, станьте им!

```
/^\d{3}-\d{3}-\d{4}$/
```

☐ (555) 935-2659

☐ (555)672-0953

☐ 555-343-8263

Серфинг может быть довольно рискованным занятием, особенно если ваша профессия — выступать в качестве приманки для акул!

☐ 555-441-9005

☐ 555.903.6386

☐ 555-612-8527-8724

Станьте регулярным выражением

Ваша задача — играть роль регулярного выражения: или принимать, или отвергать номера телефонов, вводимые пользователями приложения «Рискованные работы». Нажмите кнопку с независимой фиксацией, расположенную слева от номера телефона, который вы считаете соответствующим вам. В противном случае оставьте кнопку в ненажатом состоянии. Прокомментируйте все случаи несоответствия.

Это регулярное выражение — шаблон номера телефона, будьте им!

```
/^\d{3}-\d{3}-\d{4}$/
```

Недопустимо использование скобок, так же как и символов пробела.

☐ (555) 935-2659

☐ (555)672-0953

Никаких скобок, пожалуйста.

☑ 555-343-8263

☑ 555-441-9005

Наше регулярное выражение позволяет использовать в качестве разделителей дефисы, а не точки.

☐ 555.903.6386

Стоп, здесь какие-то четыре лишние цифры. Это добавочный номер?

☐ 555-612-8527-8724

Иногда встречаются номера телефонов с большим количеством цифр, например с добавочными офисными номерами в конце основного. Имеется ли какая-нибудь возможность проверять на соответствие регулярным выражениям и такие номера телефонов?

Такая возможность имеется, и главное, что наличие добавочного номера должно рассматриваться регулярным выражением как необязательное условие соответствия.

Если мы заменим прежнее регулярное выражение новым — `/^\d{3}-\d{3}-\d{4}-\d{4}$/`, то сделаем наличие дополнительного номера **обязательным** условием соответствия и такой номер, как, например, 555-636-4652, уже не будет соответствовать этому регулярному выражению. Но мы можем составить регулярное выражение, в котором какая-то его часть не будет являться обязательным условием соответствия. В синтаксисе регулярных выражений существует понятие **квантификатора**, который дает возможность указать, **сколь раз** символ или метасимвол может повторяться в строке. Вы уже использовали квантификаторы в таких регулярных выражениях:

```
/^\d{10}$/
```

Такая запись говорит о том, что цифра должна встретиться в строке 10 р

В данном случае выражение в фигурных скобках является квантификатором и устанавлив сколько раз подряд символ, ему предшествующий, должен встретиться в строке. Давайте рассмотрим еще несколько часто встречающихся квантификаторов.

{min,max}

Когда в фигурных скобках указываются два числа, разделенных запятой, они определяют диапазон чисел возможных повторений символа или метасимвола, предшествующего квантификатору. В этом примере регулярного выражения цифра может повторяться в строке 2, 3 или 4 раза подряд: /^\d{2,4}$/.

+

Символ или метасимвол, предшествующий в регулярном выражении этому квантификатору, может повторяться в строке один и более раз.

?

Символ или метасимвол, предшествующий в регулярном выражении этому квантификатору, может повторяться в строке **один и более раз... или вообще не встречаться ни разу.**

?

Символ или метасимвол, предшествующи в регулярном выражении этом квантификатор должен встретиться в строке **один ра или не встречать ни разу.**

Квантификатор определяет, сколько раз может появляться в строке символ, или метасимвол, или их группа.

Итак, если мы хотим, чтобы необязательное наличие добавочного номера в строк соответствовало регулярному выражению, оно должно иметь следующий вид:

```
/^\d{3}-\d{3}-\d{4}(-\d{4})?$/
```

Группа метасимволов, на которые распространяется действие квантификатора, заключена в круглые скобки.

Квантификатор ? делает последни дефис и следующие за ним четыр цифры необязательными для соответствия.

Вы совершенно правы.
С нуля начинаются служебные номера.

Все, что нам нужно, — это код города и номер телефона. Мы должны быть уверены, что первая цифра не ноль. И для того чтобы обеспечить это требование, нам понадобятся **символьные классы**.

Символьный класс позволяет установить соответствие любому символу из определенной их группы. С помощью символьного класса вы можете искать соответствие группе цифр или любых других символов. Если вы добавите в начале группы символов знак вставки (^), то тем самым установите соответствие для любого символа, не ключенного в **символьный класс**.

Для того чтобы определить, что группа символов или метасимволов образует символьный класс, она должна быть заключена в квадратные скобки []. Давайте рассмотрим несколько примеров символьных классов.

`[0-2]`

Этому символьному классу соответствует любая из цифр 0, 1 или 2.

`[A-D]`

Этому символьному классу соответствует любая из букв A, B, C или D.

В символьном классе символ вставки (^) означает «не соответствует».

`[^b-f]`

Символ вставки имеет разное значение, когда используется в символьном классе и вне его. Вместо того чтобы соответствовать началу строки, как это происходит при использовании вне класса, в символьном классе он означает «соответствует всем символам, кроме перечисленных в классе».

Символьный класс — это шаблон соответствия одному символу.

Составьте регулярное выражение на соответствие международному номеру телефона:

Тонкая настройка шаблонов с помощью символьных классов

С помощью символьных классов мы можем настроить наше регулярное выражение для номеров телефонов так, чтобы оно не соответствовало номерам с неправильными комбинациями цифр. Таким образом, если кто-нибудь введет номер телефона, начинающийся с нуля, мы можем вывести сообщение об ошибке. Вот так будет выглядеть новое, исправленное регулярное выражение:

```
/^[2-9]\d{2}-\d{3}-\d{4}$/
```

Символ вставки (^) и з $ определяют, что наш регулярное выражени должно проверяться на соответствие всей строке. Иначе говоря, строка не должна содержать никаких символов, не относящ к номеру телефона.

Код города должен состоять из трех цифр.

Символьный класс говорит, что первым символом может быть любая цифра в диапазоне от 1 до 9 включительно...

...а последующие две могут быть любыми в диапазоне от 0 до 9 включительно...

...и в конце через еще четыре любые цифры, отделенные от предыд группы дефисом.

не бывает глупых вопросов

В: Значит, символьный класс позволяет задать группу символов, одному из которых должно быть найдено соответствие в строке?

О: Да, символьный класс позволяет задать в регулярном выражении вместо одного символа группу символов, одному из которых должно быть найдено соответствие в строке.
Например, символьный класс [аеиоуыэюя] будет соответствовать одной строчной гласной букве, а класс [п-яП-Я] будет соответствовать одной строчной или прописной букве из второй половины алфавита.
Символьный класс [0–9] — эквивалент метасимвола \d, который, по сути дела, является сокращенным вариантом этого символьного класса.

В: Разве мы не должны вставлять запятые или пробелы между символами или диапазонами символов в символьном классе?

О: Нет, если вы сделаете это, эти дополнительные символы будут расценены интерпретатором регулярных выражений как часть набора символов в символьном классе и будут также использоваться для поиска соответствия.
Например, символьный класс [п-я, П-Я] будет соответствовать не только одной строчной или прописной букве из второй половины алфавита, но также и запятой, и символу пробела, что, скорее всего, не то, что вы от него хотите.

В: Что если я хочу, чтобы символ из символьного класса соответствовал более одного раза? Например, для нахождения соответствия одной или более гласных, следующих подряд?

О: Просто добавьте квантификатор после символьного класса. Регулярное выражение /[аеиоуыэюяАЕИОУЫЭЮЯ]+/ будет соответствовать одной или нескольким строчным или прописным гласным, следующим подряд.

В: Я думал, что квантификаторы относятся только к символу, непосредственно им предшествующему.

О: Обычно это так, но если квантификатор следует непосредственно за символьным классом, он относится ко всему этому классу.
А если вы хотите, чтобы квантификатор относился к группе символов, которая не является символьным классом, вы можете заключить ее в круглые скобки, указывая таким образом, что эту группу в данном контексте следует рассматривать как одно целое. Например, регулярное выражение /(привет)+/ будет соответствовать одному слову «привет» в строке или нескольким таким словам, следующим подряд.

В: Что если мне необходимо задать два разных слова, имеющих близкое значение, скажем, «кетчуп» и «соус»?

О: Вы можете использовать вертикальную черту (|), чтобы указать несколько слов на выбор. Поэтому регулярное выражение /(кетчуп|соус|подливка)/ будет соответствовать любому из этих слов.

А как поместить в регулярное выражение такие символы, как точка или вопросительный знак? Если я вставлю в него такие символы, разве PHP не станет рассматривать их в качестве метасимволов и квантификаторов и интерпретировать мое регулярное выражение совсем не так, как я того хотела?

Если у вас возникнет необходимость использовать зарезервированные символы в вашем регулярном выражении, вам необходимо их экранировать.

В синтаксисе регулярных выражений имеется небольшой набор символов, которые имеют специальное значение, так как они используются в качестве метасимволов, квантификаторов или в создании символьных классов. К таким символам относятся точка (.), вопросительный знак (?), плюс (+), открытая и закрытая круглые скобки, открытая и закрытая квадратные скобки ([ ]), открытая и закрытая фигурные скобки ({ }), знак вставки, или каре (^), знак доллара ($), вертикальная черта (|), прямая и обратная косые черты (/ \ ) и звездочка (*).

Если у вас возникнет необходимость использовать в регулярном выражении зарезервированные символы в их прямом значении, а не в качестве метасимволов или квантификаторов, вы должны экранировать их путем установки непосредственно перед ними символа обратной наклонной черты (\ ).

Например, если вы хотите включить круглые скобки в регулярное выражение для номера телефона, вы не может просто написать:

Эти скобки здесь будут рассматриваться как ограничители группы.

(555)636-4652 ✗

```
/^(\d{3})\d{3}-\d{4}$/
```

Вместо этого и открывающей, и закрывающей круглой скобке должен предшествовать символ обратной наклонной черты. Это говорит о том, что обе скобки должны интерпретироваться именно как круглые скобки:

Теперь PHP знает, что это настоящие скобки.

(555)636-4652 ✓

```
/^\(\d{3}\)\d{3}-\d{4}$/
```

Каким требованиям должны отвечать строки текста, чтобы соответствовать каждому из приведенных ниже регулярных выражений?

```
/^[3-6]{4}/
```

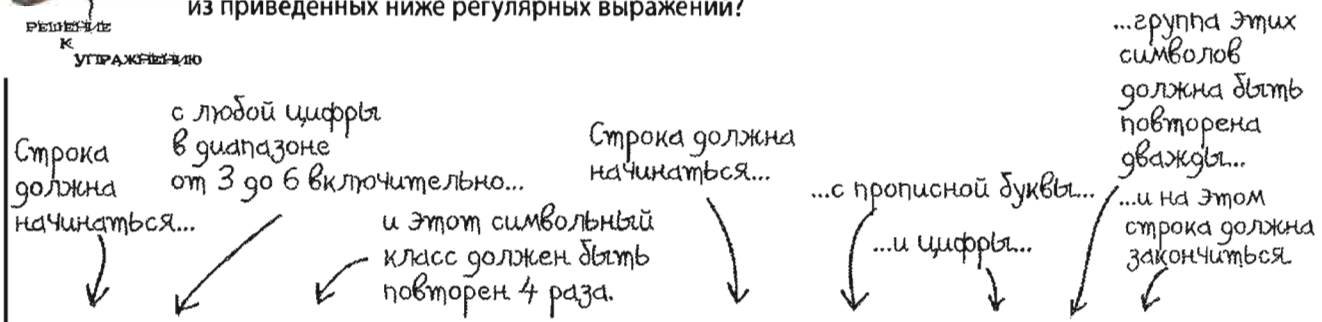
```
/^([A-Z]\d){2}$/
```

Предположим, мы хотим расширить возможности системы проверки на достоверность вводимых данных приложения «Рискованные работы» и предоставить пользователям возможность вводить номера своих телефонов еще в нескольких дополнительных форматах. Напишите единое регулярное выражение, которому будут соответствовать все показанные ниже текстовые строки, при этом номер телефона не должен соответствовать этому регулярному выражению, если первая его цифра является нулем. Шаблон должен позволять вводить только цифры, открывающие и закрывающие круглые скобки, символы пробела и дефисы.

555-636-4652 **555 636-4652**
(555)-636-4652 **(555) 636-4652**

упражнение решение

Каким требованиям должны отвечать строки текста, чтобы соответствовать каждому из приведенных ниже регулярных выражений?

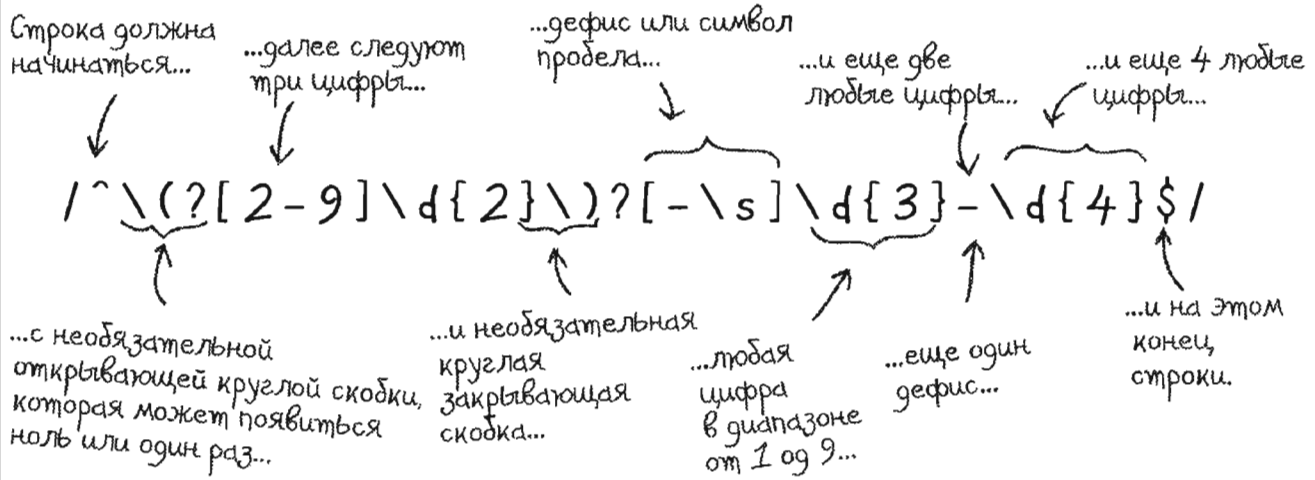


Любая строка, начинающаяся с четырех цифр в диапазоне от 3 до 6, будет соответствовать этому регулярному выражению. Например: «5533», «3546 — это число», «6533эюя».

Любая строка, состоящая из прописной буквы и числа, а затем опять прописной буквы и числа, причем на этом строка заканчивается: «Б5Д9», «Р2Д2».

Предположим, мы хотим расширить возможности системы проверки на достоверность вводимых данных приложения «Рискованные работы» и предоставить пользователям возможность вводить номера своих телефонов еще в нескольких дополнительных форматах. Напишите единое регулярное выражение, которому будут соответствовать все показанные ниже текстовые строки, при этом номер телефона не должен соответствовать этому регулярному выражению, если первая его цифра является нулем. Шаблон должен позволять вводить только цифры, открывающие и закрывающие круглые скобки, символы пробела и дефисы.

555-636-4652 **555 636-4652**
(555)-636-4652 **(555) 636-4652**



У меня голова идет кругом от неправильных данных в приложении «Рискованные работы». Какой прок от всех этих регулярных выражений, если мы не можем их как-то использовать?

Приложению «Рискованные работы» необходимо использовать регулярные выражения для проверки данных формы на соответствие шаблонам.

Проверка соответствия с помощью функции preg_match()

Мы разрабатываем регулярные выражения не просто для развлечения. Вы можете использовать их в PHP-функции `preg_match()`. Она принимает в качестве одного из своих аргументов регулярные выражения, подобные тем, что мы составляли. В качестве второго шаблона выступает строка текста. Функция возвращает `true` в том случае, если в строке текста найдено соответствие регулярному выражению, и `false`, если такое соответствие отсутствует.

```
preg_match($regex, $my_string)
```

Это ваше регулярное выражение. Тип этого аргумента — текстовая строка. Это означает, что регулярное выражение должно быть заключено в одинарные кавычки.

Это строка, которую вы хотите проверить на соответствие регулярно выражению.

Ниже приведен пример использования функции `preg_match()`:

При передаче функции preg_match() регулярного выражения оно должно быть заключено в кавычки.

```
preg_match('/^\d{3}-\d{2}-\d{4}$/', '555-02-9983')
```

Возвращает целое число: 1, если имеется соответствие, и 0 в случае отсутствия такого соответствия.

Вы можете передавать функции строку регулярного выражения, как здесь, но обычно лучше предварительно сохранять значение регулярного выражения в переменной.

Эта строка соответствует регулярному выражению, поэто функция возврати

Мы можем использовать функцию `preg_match()` в управляющей конструкции `if`, чтобы создать код для проверки соответствия вводимых данных шаблону в сценарии PHP, основываясь на значении, возвращаемом функцией.

Функция preg_match() встроена в усл выражение, поэтому значение, котор она возвращает, определяет, какой б кода будет выполнен.

```
if (preg_match('/^\d{3}-\d{2}-\d{4}$/', '555-02-9983')) {
    echo 'Действительный номер социальной страховки';
} else {
    echo 'Недействительный номер социальной страховки';
}
```

Если соответствие не найдено, функция preg_match() возвратит false, в результате чего будет выполнен второй блок кода.

Если соответствие найдено, функция preg_match() возвратит true, в результате чег будет выполнен первый блок кода.

Перепишите в PHP-сценарии приложения «Рискованные работы» блок кода, ответственный за проверку текста, введенного в поле формы «Номер телефона». Вместо функции `empty()` используйте функцию `preg_match()` с передачей ей в качестве аргумента регулярного выражения, созданного вами ранее.

```
if (empty($phone)) {
  // Переменная $phone не определена
      echo '<p class="error"> Вы забыли ввести номер своего
телефона.</p>';
      $output_form = 'yes';
}
```

Перепишите в PHP-сценарии приложения «Рискованные работы» блок кода, ответственный за проверку текста, введенного в поле формы «Номер телефона». Вместо функции `empty()` используйте функцию `preg_match()` с передачей ей в качестве аргумента регулярного выражения, созданного вами ранее.

```
if (empty($phone)) {
  // Переменная $phone не определена
    echo '<p class="error"> Вы забыли ввести номер своего
телефона.</p>';
    $output_form = 'yes';
}
```

Для проверки правильности введенного номера телефона вместо функции empty() мы используем функцию preg_match(). Наименованию функции предшествует оператор отрицания (!), потому что мы будем выводить сообщение об ошибке только в случае, если номер телефона НЕ соответствует шаблону.

Наше регулярное выражение для номера телефона, составленное ранее.

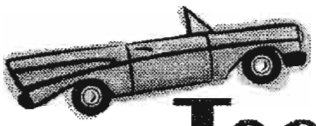
```
if (!preg_match('/^\(?[2-9]\d{2}\)?[-\s]\d{3}-\d{4}$/', $phone))
 // $phone is not valid
 echo '<p class="error"> Вы ввели неправильный номер телефона.</
 $output_form = 'yes';
}
```

Сообщение об ошибке должно быть немного изменено, потому что теперь мы проверяем не только то, ввел пользователь номер телефона или нет, но также и соответствие введенного номера телефона стандартному шаблону.

Мы присваиваем переменной формы $output значение yes, так же как и в первоначальной версии сценария.

Получив сообщение об ошибке в номере моего телефона, я исправил ее. И вот я получил работу ниндзя!

Имя: Джимми
Фамилия: Свифт
Адрес электронной почты:
JS@sim-u-duck.com
Телефон: 636 4652
Желаемая работа: Ниндзя



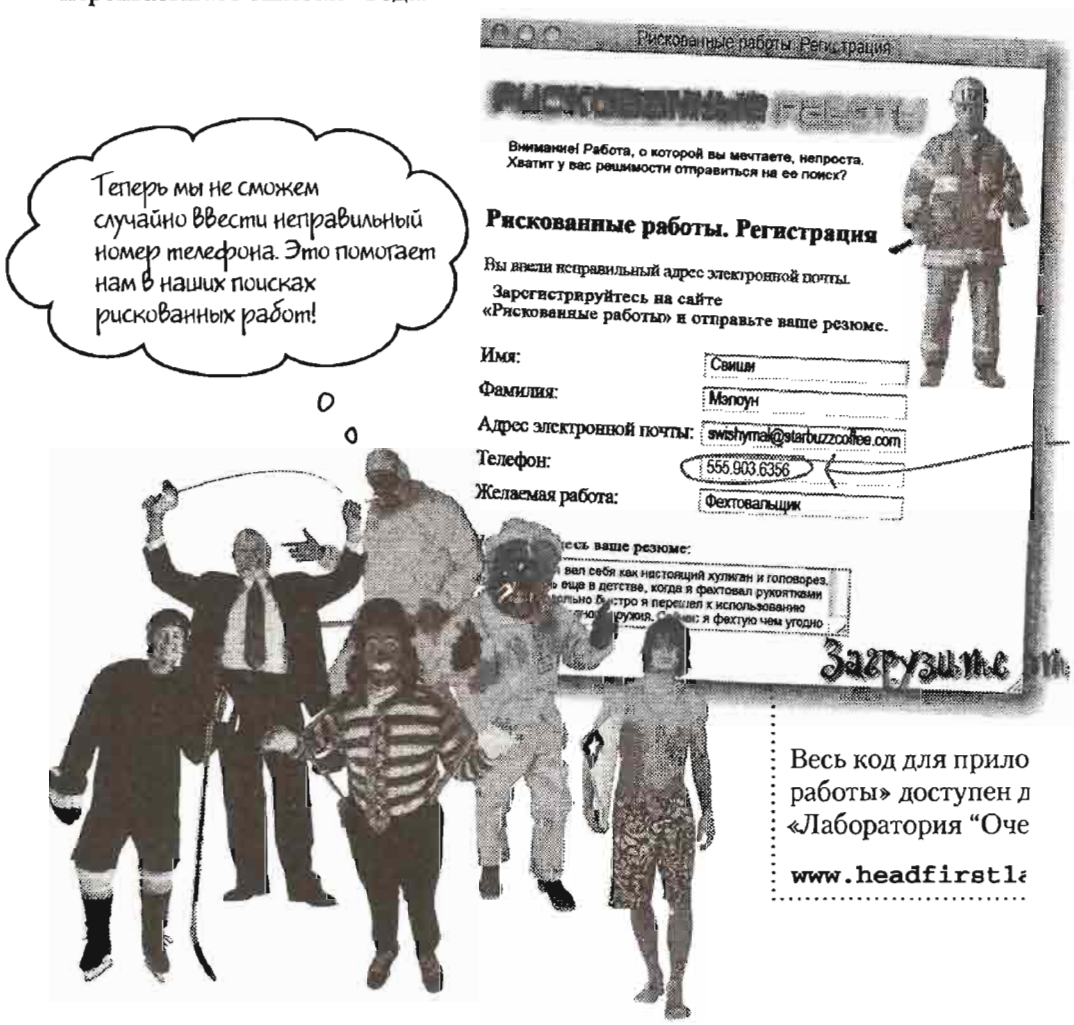
Тест-драйв

Поле ввода номера телеф
содержать не просто ка
число должно быть в опр

**Проверка правильности введенного номера телефона в реги
сценарии приложения «Рискованные работы».**

Загрузите сценарий `registration.php` с сайта по адресу `www.headfirstlabs` вместе с каскадными таблицами стилей приложения «Рискованные работы» (`sty` изображений (`riskyjobs_title.gif` и `riskyjobs_fireman.png`). Затем вне `registration.php` изменения, обеспечивающие использование в нем функции для проверки соответствия введенного номера телефона регулярному выражени Не забудьте подкорректировать сообщение об ошибке так, чтобы пользователь зн неправильный номер телефона, а не просто оставил поле ввода пустым.

Загрузите откорректированный сценарий на ваш веб-сервер и затем откройте его Попробуйте ввести несколько номеров телефонов в разном формате и проверьте перехватывает ошибки ввода.



Весь код для прило
работы» доступен д
«Лаборатория “Оче

www.headfirstl

Хм... Если нашему регулярному выражению соответствуют несколько форматов номеров телефонов, разве это не означает, что в базе данных эта информация также будет записана в разных форматах? Это не слишком хорошо. Я думаю, нам необходимо стандартизировать эти вещи.

То, что вы разрешили вводить данные в разных форматах, совсем не означает, что вы хотите, чтобы все эти данные были сохранены во всех этих форматах.

К счастью, имеется еще одна функция, связанная с регулярными выражениями и дающая нам возможность взять данные, введенные пользователем в форму приложения «Рискованные работы», и изменить их так, чтобы они соответствовали лишь одному шаблону вместо четырех. PHP-функция `preg_replace()` продвигается на один шаг дальше функции `preg_match()` в использовании регулярных выражений для работы со строками текста. В дополнение к тому, что она способна определить, соответствует ли данная строка текста регулярному выражению, она также может заменить эту строку другой строкой, переданной ей в качестве аргумента. Это напоминает действия функции `str_replace()`, которую мы уже использовали, с той лишь разницей, что поиск производится на соответствие регулярному выражению, а не строке текста.

```
preg_replace($pattern, $replacement, $my_string)
```

Нам необходимо найти эти ненужные символы.

После того как мы нашли ненужные нам символы, мы хотим заменить их этим.

Строка, в которой мы проводим эту операцию поиска и замены.

Вот пример использования функции `preg_replace()`:

```
$new_year = preg_replace('/200[0-9]/', '2010', 'The year is 2009.');
```

Результат, возвращенный функцией `preg_replace()`, — это наша исправленная строка, полученная после того, как операция поиска и замены закончилась, и сохраненный в переменной `$new_year`.

Это регулярное выражение говорит функции `preg_replace()`, что нужно искать строку с одним из значений года в диапазоне от 2000 до 2009.

Как только соответствие регулярному выражению будет найдено, нужная строка должна быть заменена строкой «2010».

Каждый раз, когда строка, содержащая одно из значений года в диапазоне от 2000 до 2009, будет найдена, она будет заменена строкой со значением «2010».

Стандартизируйте номера телефонов, введенные в форму приложения «Рискованные работы», записав все эти номера в показанную ниже колонку phone таблицы базы данных. Запишите эти данные в таком формате, чтобы в нем использовалось как можно меньше символов.

(555) 935-2659

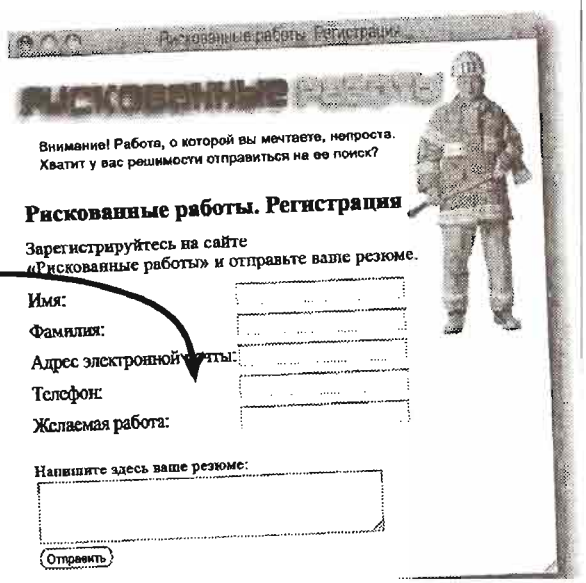
(555)672-0953

555-343-8263

555-441-9005

555.903.6386

555-612-8527-8724



Стандартизируйте номера телефонов, введенные в форму приложения «Рискованные работы», записав все эти номера в показанную ниже колонку phone таблицы базы данных. Запишите эти данные в таком формате, чтобы в нем использовалось как можно меньше символов.

(555) 935-2659

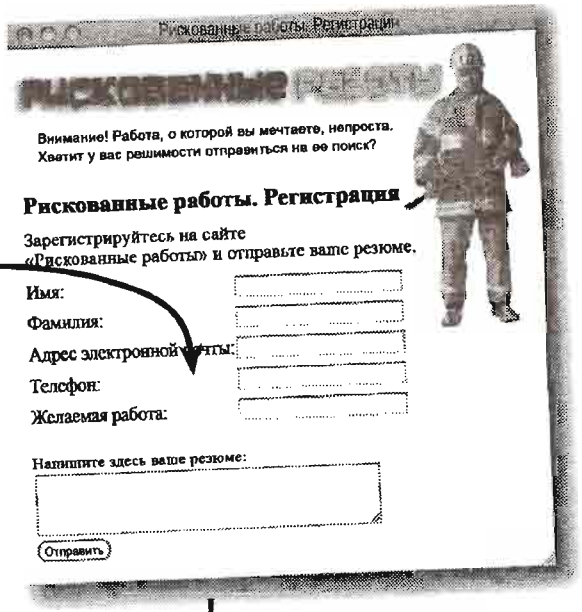
(555)672-0953

555-343-8263

555-441-9005

555.903.6386

555-612-8527-8724



Наиболее простой и надежный метод сохранять номера телефонов заключается в предварительном удалении из них всех символов, кроме цифр.

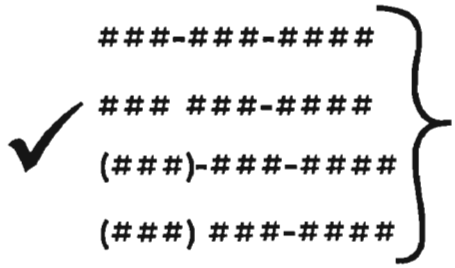
phone
...
5559352659
5556720953
5553438263
5554419005
5559036386
5556128527

Стандартизируйте номера телефонов

Сейчас приложение «Рискованные работы» использует следующее регулярное выражение, чтобы проверить соответствие номеров телефонов шаблону в процессе внесения их в форму:

```
/^\(?[2-9]\d{2}\)?[-\s]\d{3}-\d{4}$/
```

Этому регулярному выражению будут соответствовать номера телефонов, составленные по следующим четырем шаблонам:



Мы хотим переформатировать наши данные из такого вида...

В то время как все эти форматы достаточно удобны и легко понимаются людьми, совместное их использование может привести к проблемам при сортировке данных во время выполнения SQL-запросов к базе данных. Например, эти круглые скобки, скорее всего, помешают нашим попыткам сгруппировать записи в результате запроса по кодам городов, что могло бы оказаться очень важным для анализа распределения посетителей сайта по географическим регионам.

Для того чтобы сделать возможным подобную группировку, нам необходимо привести все номера телефонов к одному стандартному виду, используя функцию `preg_replace()`, перед тем как сохранять данные в базе. Наилучшее решение заключается в том, чтобы избавиться в строках номеров телефонов от всех символов, кроме цифр. В этом случае мы сохраняем в базе данных только 10 цифр без каких-либо других дополнительных символов. Мы хотим сохранять в базе данных номера телефонов в таком формате:

...в такой

```
##########
```

Для этого нам нужно найти в строке номера телефона четыре символа и удалить их. Мы хотим найти и удалить все открывающие и закрывающие круглые скобки, символы пробелов и дефисы. И мы хотим найти эти символы независимо от того, в каких местах строки они располагаются, поэтому нам не понадобится использовать символ вставки (^ — начало строки) и знак доллара ($ — конец строки). Мы знаем, что ищем любой символ из этого набора, поэтому нам лучше всего использовать символьный класс. Порядок поиска не имеет значения. Вот регулярное выражение, которое мы можем использовать:

```
/[\(\)\-\s]/
```

открывающая круглая скобка

закрывающая круглая скобка

дефис

символ пробела

Стандартизация формата данных поможет вам получать более удобные результаты SQL-запросов.

Удалите ненужные символы

Теперь, когда у нас есть регулярное выражение, которому соответствуют все ненужные нам символы, мы можем использовать его для того, чтобы очистить строки номеров телефонов от всех этих ненужных символов, прежде чем сохранять их в базе данных. Но как? Вот здесь-то нам и пригодится функция `preg_replace()`. Трюк заключается в том, что нам не нужно заменять ненужные символы другими: нам нужно удалить их. Тогда мы просто передадим функции `preg_replace()` в качестве второго аргумента пустую строку. Ниже приведен пример использования функции `preg_replace()` для поиска ненужных символов и замены их пустыми строками.

Мы сохраняем результат нашей процедуры «найти и заменить» в этой переменной нового номера телефона.

Замена производится в текстовой строке $phone.

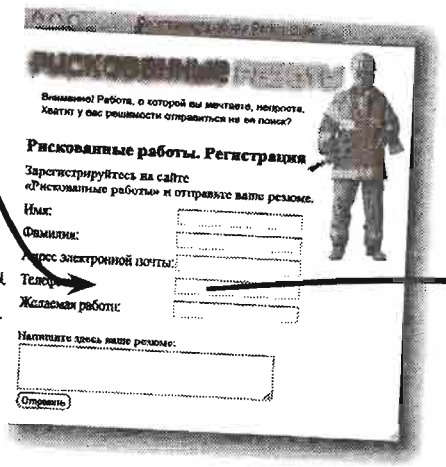
```
$new_phone = preg_replace('/[\(\)\-\s]/', '', $phone);
```

Найти символы, соответствующие этому регулярному выражению...

...и заменить их пустой строкой.

```
###-###-####
### ###-####
(###)-###-####
(###) ###-####
```

Все эти форматы номеров телефонов считаются правильными и принимаются формой приложения «Рискованные работы».



preg_replace()

phone
...
5559352659
5556720953
5553438263
5554419005
5559036386
5556128527

```
##########
```

Каждый номер телефона преобразуется в соответствии с этим форматом, чтобы в базе данных все номера телефонов были представлены в едином стандартном виде.

Можно, но позднее это повлечет за собой проблемы, так как запросы к базе данных будут возвращать не такие результаты, какие нам хотелось бы видеть.

Многие пользователи привыкли вводить номера телефонов с определенными комбинациями дефисов, скобок и символов пробела, поэтому нежелательно заставлять их вводить номера телефонов, состоящие из одних цифр, так как это может привести к проблемам. Значительно лучше встречать пользователей на полпути, давая им оправданно гибкие методы ввода данных и в то же самое время обеспечить наиболее оптимальный формат данных, в котором вы сохраняете их в базе.

Кроме того, для разрешения этой проблемы мы используем только один вызов функции `preg_replace()`, что само по себе не является слишком сложным делом. Если бы мы вели речь о создании какой-нибудь пользовательской функции с огромным количеством кода, это была бы совсем другая история. Но увеличение удобства использования приложения с одновременным повышением уровня целостности данных за счет одной лишь строки кода — не бином Ньютона!

Тест-драйв

Удалите ненужные нам символы из номеров телефонов.

Внесите изменения в сценарий `registration.php`, чтобы обеспечить удаление ненужных нам символов из номеров телефонов. Добавьте в сценарий следующие строки кода сразу после строк благодарности пользователю за то, что он регистрируется в приложении «Рискованные работы»:

```
$pattern = '/[\(\)\-\s]/';

$replacement = '';

$new_phone = preg_replace($pattern, $replacement, $phone);

echo 'Номер вашего телефона зарегистрирован как ' .  '.</p>';
```

Загрузите откорректированный сценарий на ваш веб-сервер и затем откройте его в браузере. Заполните форму, при этом введите номер телефона с дополнительными символами, например, так: (707) 827-7000. Отправьте форму на сервер для обработки и проверьте результат.

Попробуйте ввести номер телефона в других форматах, например, 707.827.7000, (707)-827-7000, 707 827-7000. Обратите внимание на то, как функция `preg_replace()` вместе с регулярным выражением удаляет все ненужные нам символы из строки номера телефона.

Так же как и в случае с номерами телефонов, адреса электронной почты должны не только иметь непустое значение, но и соответствовать определенному шаблону, что нам также необходимо проверять.

Так же как и для номеров телефонов, вначале мы должны определить правила составления адреса электронной почты. Затем мы должны формализовать их в виде регулярного выражения и включить в PHP-сценарий. Поэтому давайте вначале рассмотрим, из каких частей состоит адрес электронной почты.

Мы знаем, что в адресе электронной почты должны обязательно присутствовать эти два символа.

```
LocalName@DomainPrefix.DomainSuffix
```

Эти элементы состоят из алфавитно-цифровых символов. Локальное имя пользователя может состоять из одного символа и более, префикс домена — из двух и более символов.

Этот элемент обычно состоит из двух-трех символов.

СИЛА МЫСЛИ

Попробуйте составить регулярное выражение, которому будут соответствовать адреса электронной почты, приведенные справа.
Запишите это регулярное выражение ниже:

.....

```
aviator.howard@bannerocity.com
cube_lovers@youcube.ca
rocky@i-rock.biz
```

Создать шаблон адреса электронной почты — задача нетривиальная

С первого взгляда может показаться, что создать шаблон адреса электронной почты — задача несложная, на его символы ведь не распространяется такое количество ограничений, с которыми мы столкнулись, имея дело с номерами телефонов.

Например, казалось бы, что может быть проще, чем составить регулярное выражение для поиска соответствия части адреса, расположенной слева от символа @, которую мы назвали «Локальное имя пользователя». Поскольку этот элемент состоит только из алфавитно-цифровых символов, мы могли бы использовать следующее регулярное выражение:

```
/^\w+/
```

Начинается...

... с одного или нескольких алфавитно-цифровых символов.

Этому выражению будут соответствовать строки, состоящие из любых алфавитно-цифровых символов, но, к сожалению, оно не включает целый ряд других символов, также разрешенных к использованию в адресах электронной почты.

Вы можете не поверить, но перечисленные ниже символы также допустимы к применению в левой от символа @ части адреса электронной почты, хотя некоторые из них не должны быть на первом месте:

Все эти символы могут быть использованы в части «Локальное имя пользователя» адреса электронной почты.

```
! $ & * - = ^ ` | ~ # % ' + / ? _ { }
```

Если мы хотим предоставить пользователю возможность зарегистрировать адрес электронной почты, содержащий подобные символы, нам и вправду понадобится регулярное выражение, которое выглядит примерно так:

```
/^[a-zA-Z0-9][a-zA-Z0-9\._\-&!?=#]*/
```

Первый символ должен быть одним из этого набора.

Остальные символы могут быть любыми из этих...

...и их количество может быть в пределах от нуля (ни одного) и более.

Строки «Локальное имя пользователя» не со всеми перечисленными допустимыми символами будут соответствовать этому регулярному выражению, так как мы пропустили некоторые самые редко используемые из них. Но с практической точки зрения это регулярное выражение вполне приемлемо для большинства пользователей приложения «Рискованные работы».

С адресом электронной почты иметь дело нетрудно. Нужно создать регулярное выражение для проверки доменного имени, примерно такое же, какое мы создали для строки «Локальное имя пользователя»... Все очень просто.

Этот подход годится для первой части доменного имени — «Префикс домена», но не годится для второй его части — «Суффикс домена».

В то время как строка «Префикс домена» может быть комбинацией практически любых алфавитно-цифровых и некоторых специальных символов, почти как и строка «Локальное имя пользователя», ограничения, накладываемые на строку «Суффикс домена», более строгие.

Большинство адресов электронной почты оканчиваются на один из нескольких суффиксов `.com`, `.edu`, `.org`, `.gov`, `.by`, `.ru` и т. д. Мы должны убедиться, что адрес электронной почты оканчивается на действительный суффикс.

не бывает глупых вопросов

В: Что если я захочу разрешить все возможные из допустимых адресов?

О: Вы можете это сделать и, если это имеет смысл для вашего сайта, просто должны это сделать. Но иногда лучше выбрать несколько общепринятых форматов, а не ставить задачу учесть все допустимые варианты. Вам необходимо решить, какие форматы адресов электронной почты будут использовать 99,9 % ваших пользователей, и оставить без внимания остальные 0,1 %, чтобы не перегружать код исключительно редко используемыми подробностями. Проверка на соответствие — это всегда компромисс между тем, что соответствует, и тем, что приемлемо.
Если вы действительно хотите применить в своем приложении надежный и устойчивый код, вы можете загрузить отличный свободный PHP-код по адресу: `http://code.google.com/p/php-email-address-validation/`

В: А пользователи не будут на меня обижаться, если их адреса электронной почты будут отвергнуты моим приложением?

О: Скорее всего, будут, но мало у кого из пользователей имеются сумасшедшие адреса электронной почты. Большинство служб электронной почты в Интернете устанавливают ограничивающие правила, которые удерживают пользователей от создания сумасшедших, хотя и допустимых адресов, подобных таким:

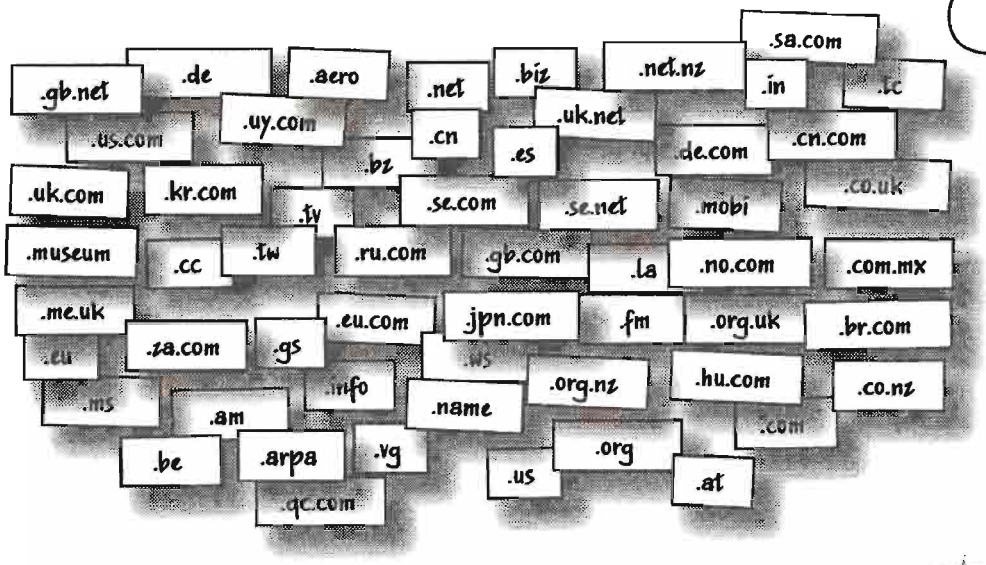
```
"_i'm crazy"@greg-list.net
```

Проверка на соответствие — это всегда компромисс между тем, что соответствует, и тем, что приемлемо.

Доменные суффиксы везде

В дополнение к наиболее общепринятым доменным суффиксам, которые вы видите очень часто, таким, например, как `.com` и `.org`, имеется огромное множество других доменных суффиксов, и они являются допустимыми для применения в адресах электронной почты. Все доменные суффиксы распознаются как допустимые системой доменных имен (Domain Name System, или сокращенно DNS). В дополнение к перечисленным ранее доменным суффиксам существуют также суффиксы, образованные от двухбуквенных кодов стран, например `.ca` — для Канады, `.tj` — для Таджикистана.

Ниже перечислено относительно небольшое количество доменных суффиксов. Это далеко не полный перечень.



Некоторые из этих доменов состоят только из двух букв. Некоторые — из двух или трех букв, точки и еще двух или трех букв. Другие состоят из четырех и даже пяти букв. Может, для проверки соответствия нам нужен список всех этих доменов?

Мы могли бы сделать так, и это бы работало.

Но есть более простой путь. Вместо того чтобы отслеживать все возможные имена доменов и при появлении новых имен корректировать свой код, мы можем проверять доменную часть адреса электронной почты (ту его часть, которая расположена справа от символа `@`), используя PHP-функцию `checkdnsrr()`. Она соединяется с системой доменных имен (DNS) и проверяет достоверность домена.

Чокнутые биты

Система доменных имен (Domain Name System, или сокращенно DNS) — это распределенная база данных, содержащая информацию о доменах, их именах и IP-адресах. Данная система делает возможным использование доменных имен вместо IP-адресов. Без нее мы вынуждены были бы, например, вместо доменного имени `oreilly.com` вводить IP-адрес `208.201.239.36`.

Используйте PHP для проверки доменов

PHP предоставляет функцию `checkdnsrr()` для проверки того, является ли переданная ей в качестве аргумента строка именем зарегистрированного домена. Этот метод даже лучше метода, основанного на использовании регулярного выражения для проверки соответствия доменного имени шаблону, так как если при использовании регулярного выражения мы проверяем, может ли данная строка быть именем домена, при использовании функции `checkdnsrr()` мы обращаемся к системе доменных имен с запросом, зарегистрирован ли домен с таким именем. Поэтому если проверка, основанная на использовании регулярных выражений, сообщит вам, что строка текста `lasdjkdfsalkjaf.com` может быть использована в качестве имени домена, то функция `checkdnsrr()` проследует далее и сообщит, что домен с таким именем не зарегистрирован, и вам, скорее всего, придется отвергнуть адрес электронной почты `sdfhfdskl@lasdjkdfsalkjaf.com`, если пользователь введет его в форму.

Синтаксис функции `checkdnsrr()` достаточно прост:

Возвращает 1, если домен с таким именем зарегистрирован, и 0, если нет.

Функция checkdnsrr() принимает в качестве аргумента строку, содержащую имя домена. Это часть адреса электронной почты, расположенная справа от символа @.

```
checkdnsrr('headfirstlabs.com')
```

Запомни!

Если вы используете PHP на сервере под управлением операционной системы Windows, эта функция работать не будет.

Вместо нее вы можете использовать следующий код:

Эта проблема возникает только в том случае, если ваш сервер работает под управлением операционной системы Windows. Если вы разрабатываете ваш сайт на компьютере, на котором установлена операционная система Windows, но затем отправляете ваше приложение для работы на сервере под управлением операционной системы Unix/Linux, тогда никаких проблем не возникает.

```
function win_checkdnsrr($domain,$recType='') {
 if (!empty($domain)) {
  if ($recType=='') $recType="MX";
  exec("nslookup -type=$recType $domain",$output);
  foreach($output as $line) {
   if (preg_match("/^$domain/", $line)) {
    return true;
   }
  }
  return false;
 }
 return false;
}
```

Функция exec() выполняет внешнюю программу на сервере для проверки имени домена.

Просто ради шутки попробуйте вывести содержание строки $line сразу после выхода из цикла foreach. Вы увидите что-то вроде этого: Server: 68.87.64.14Address: 68.87.64.14#53Non-authoritative answer: oreily.com mail exchanger = 20 smtp1.oreily.com.

Проверка адреса электронной почты: весь процесс полностью

Теперь мы знаем, как проверять части адреса электронной почты, расположенные по обе стороны от символа @: ту, что мы назвали «Локальное имя пользователя» — сопоставлением с регулярным выражением, а имя домена — вызовом функции `checkdnsrr()`. Давайте проследим поэтапно, как мы сможем собрать все это вместе, чтобы добавить полнофункциональную проверку адреса электронной почты при вводе его в регистрационную форму приложения «Рискованные работы».

1. Использование функции `preg_match()` для проверки отсутствия недопустимых символов в части адреса электронной почты под названием «Локальное имя пользователя», расположенной слева от символа @.

   Мы можем использовать для этого следующее регулярное выражение:

```
/^[a-zA-Z0-9][a-zA-Z0-9\._\-&!?=#]*/
```

Обратите внимание на отсутствие знака доллара в конце этого регулярного выражения. Его не должно быть потому, что после символа @ последует имя домена.

Адрес электронной почты должен начинаться с алфавитно-цифрового символа и далее содержать любое количество алфавитно-цифровых и некоторых специальных символов.

На этот раз мы также указываем символ @, чтобы быть уверенными, что он присутствует перед именем домена.

2. Если проверка части адреса электронной почты под названием «Локальное имя пользователя», расположенной слева от символа @, не проходит — вывод сообщения об ошибке и перезагрузка формы.

3. Если проверка части адреса электронной почты под названием «Локальное имя пользователя», расположенной слева от символа @, проходит — вызов функции `checkdnsrr()` с передачей ей в качестве аргумента расположенного справа от символа @ имени домена.

4. Если функция `checkdnsrr()` возвращает 0, то такое имя домена не зарегистрировано, поэтому необходимо вывести сообщение об ошибке и перезагрузить форму.

5. Если функция `checkdnsrr()` возвращает 1, то такое имя домена зарегистрировано и введенный пользователем адрес электронной почты может быть принят. Мы можем продолжать проверку остальных полей формы.

Ниже приведен PHP-код, обеспечивающий проверку адресов электронной почты, введенных пользователями, но некоторые фрагменты кода пропали. Заполните пропуски так, чтобы привести этот код в рабочее состояние.

```
if (!preg_match(' ..........', $email)) {
  // Значение переменной $email не соответствует регулярному выражению.
  // Неправильная часть адреса электронной почты под названием «Локальное имя пользователя»,
  // расположенная слева от символа @
  echo '<p class="error">Вы ввели неправильный адрес электронной почты.</p>';
  $output_form = 'yes';
}
else {
  // Удаление из адреса электронной почты всего, кроме имени домена, т. е. части,
  // расположенной справа от символа @
  $domain = preg_replace('..........', '...', ..........);
  // Теперь проверка того, зарегистрирован домен с таким именем или нет
  if (..........)) {
    echo '<p class="error"> Вы ввели неправильный адрес электронной почты.</p>';
    $output_form = 'yes';
  }
}
```

упражнение решение

Ниже приведен PHP-код, обеспечивающий проверку адресов электронной почты, введенных пользователями, но некоторые фрагменты кода пропали. Заполните пропуски так, чтобы привести этот код в рабочее состояние.

Это наше регулярное выражение для проверки части адреса электронной почты под названием «Локальное имя пользователя».

```
if (!preg_match('[a-zA-Z0-9][a-zA-Z0-9\._\-&!?=#]*@/', $email)) {
  // Значение переменной $email не соответствует регулярному выражению.
  // Неправильная часть адреса электронной почты под названием «Локальное имя пользователя»,
  // расположенная слева от символа @
  echo '<p class="error">Вы ввели неправильный адрес электронной почты.</p>';
  $output_form = 'yes';
}
else {
  // Удаление из адреса электронной почты всего, кроме имени домена, т. е. части,
  // расположенной справа от символа @
  $domain = preg_replace('/^[a-zA-Z0-9][a-zA-Z0-9\._\-&!?=#]*@/', '', $email);
  // Теперь проверка того, зарегистрирован домен с таким именем или нет
  if ( !checkdnsrr($domain) ) {
    echo '<p class="error">Вы ввели неправильный адрес электронной почты.</p>';
    $output_form = 'yes';
  }
}
```

Для того чтобы удалить часть адреса электронной почты под названием «Локальное имя пользователя» и символа @, указана пустая строка для замены.

Замена производится в строке $email!.

Функция с оператором отрицания !checkdnsrr() возвратит true, если домен не зарегистрирован.

Если вы работаете с сервером под управлением операционной системы Windows, не забудьте включить код для определения функции win_checkdnsrr() и ее вызова.

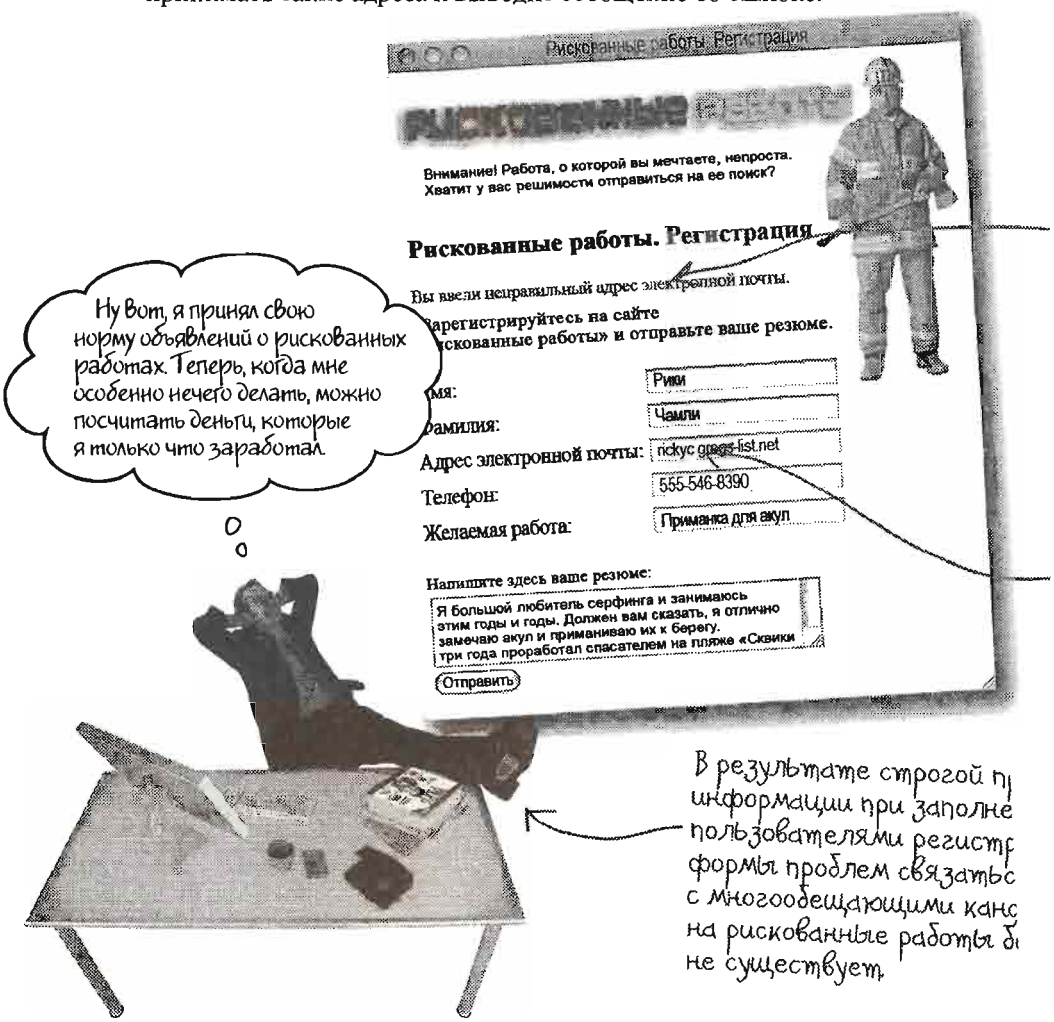
КЛЮЧЕВЫЕ МОМЕНТЫ

- `preg_match()` проверяет соответствие строки регулярному выражению.
- `preg_replace()` заменяет строку, соответствующую регулярному выражению, другой строкой.
- **Квантификатор** определяет, сколько раз символ или группа символов может повторяться в строке.
- Вы можете определить группу символов, которые должны появиться в вашей строке, с помощью **символьного класса**.
- В вашем регулярном выражении `\d`, `\w` и `\s` представляют цифру, алфавитный символ и пробельный символ соответственно.
- `checkdnsrr()` проверяет, зарегистрировано ли имя домена, переданное ей в качестве аргумента.

Тест-драйв

Добавьте проверку правильности адреса электронн в регистрационный сценарий приложения «Рискова

Используйте код, приведенный на предыдущей странице, для добавле в регистрационный сценарий проверки правильности адреса электрон Загрузите откорректированный сценарий на ваш веб-сервер и затем о в браузере. Попробуйте ввести несколько неправильных адресов элек и проверьте, как новый код с использованием регулярных выражений принимать такие адреса и выводит сообщение об ошибке.



Ваш инструментарий PHP и MySQL

Проверка строки текста на соответствие определенному шаблону может очень пригодиться в процессе определения правильности введенных пользователем данных. Ниже приведены используемые в PHP приемы такой проверки, осуществляемой с помощью регулярных выражений.

Регулярное выражение

Правила для нахождения соответствия строки текста определенному шаблону. В PHP имеются функции, которые позволяют вам использовать регулярные выражения как для проверки соответствия строки определенному шаблону, так и для нахождения и замещения подобных строк в тексте.

\d, \w, \s, ^, $...

Регулярное выражение создается с использованием метасимволов, которые представляют различные шаблоны, такие как три цифры (\d\d\d) или пробельный символ (\w).

Символьный класс

Набор правил, или шаблон для определения соответствия одному символу в регулярном выражении. Например, [А-Г] соответствует любому из символов А, Б, В или Г.

preg_match()

Эта PHP-функция проверяет строку текста на соответствие регулярному выражению. Функция возвращает true, если такое соответствие найдено, и false, если его нет.

checkdnsrr()

Эта PHP-функция проверяет, зарегистрирован ли домен с именем, переданным ей в качестве аргумента. Это очень удобно для проверки адреса электронной почты, потому что позволяет вам убедиться, что в качестве имени домена в адресе используется имя зарегистрированного домена.

preg_replace()

Используйте эту функцию для замещения строки в тексте на основании ее соответствия определенному регулярному выражению. Функция производит поиск строки, соответствующей регулярному выражению, и заменяет ее на строку, переданную ей в качестве одного из аргументов.

11 Визуализация ваших данных... и более!

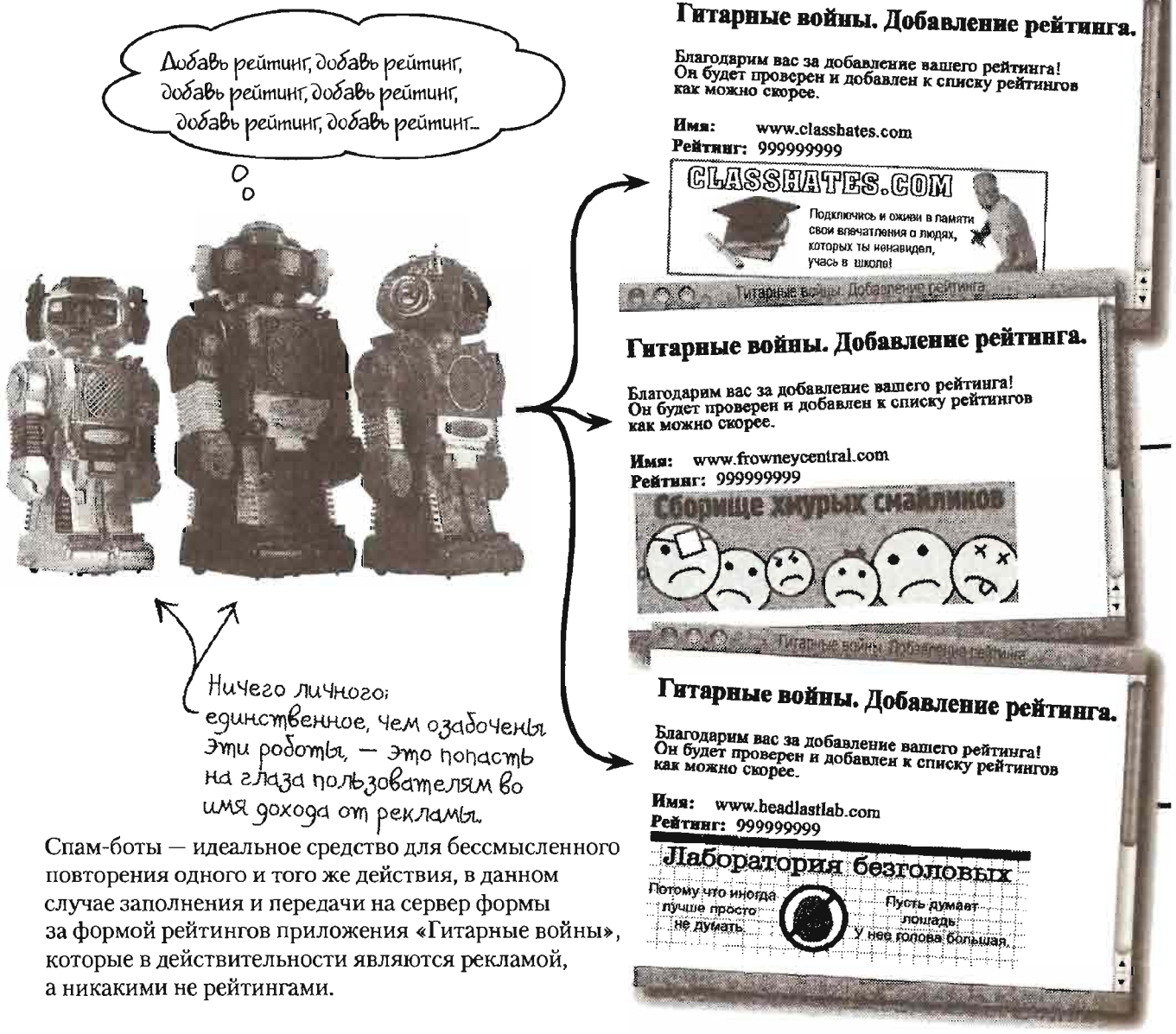
Динамическое создание изображений

Конечно, все мы знаем широкие возможности грамотно составленного запроса, предоставляющего великолепные результаты. Но результаты запросов не всегда достаточно удобны для восприятия. Иногда полезно показать данные в другом, более наглядном свете. PHP дает возможность представлять данные вашей базы в различных графических видах: в виде секторной диаграммы, столбчатой гистограммы, диаграммах Венна, тестах Роршаха… Вы можете продолжить этот перечень. Все, что дает пользователю возможность воспринимать данные в процессе работы вашего приложения, может рассматриваться как игра. Но в качестве источника данных для построения изображений в вашем приложении может выступать не только информация, сохраненная в базе данных. Например, знаете ли вы, что вполне возможно защитить форму от занесения в нее данных программой-роботом по распространению спама (спам-ботом) с помощью динамически генерируемых изображений?

Перезагрузка приложения «Гитарные войны»: восстание роботов

Будущее наступило. Роботы уже свободно перемещаются в виртуальном пространстве, и ничто не сможет остановить их шествия, кроме определенного бдительного PHP-кода. Это спам-боты, которые перехватывают веб-формы и распространяют через них различные рекламные объявления. Эти роботы хладнокровны и эффективны, и их совершенно не интересует, для какой цели предназначены атакуемые ими формы. Их главная и единственная цель — заменить оригинальное содержание своим в процессе жестокой борьбы за доход от рекламы для своих хозяев. На свое несчастье, приложение «Гитарные войны» пало жертвой в этой войне с роботами.

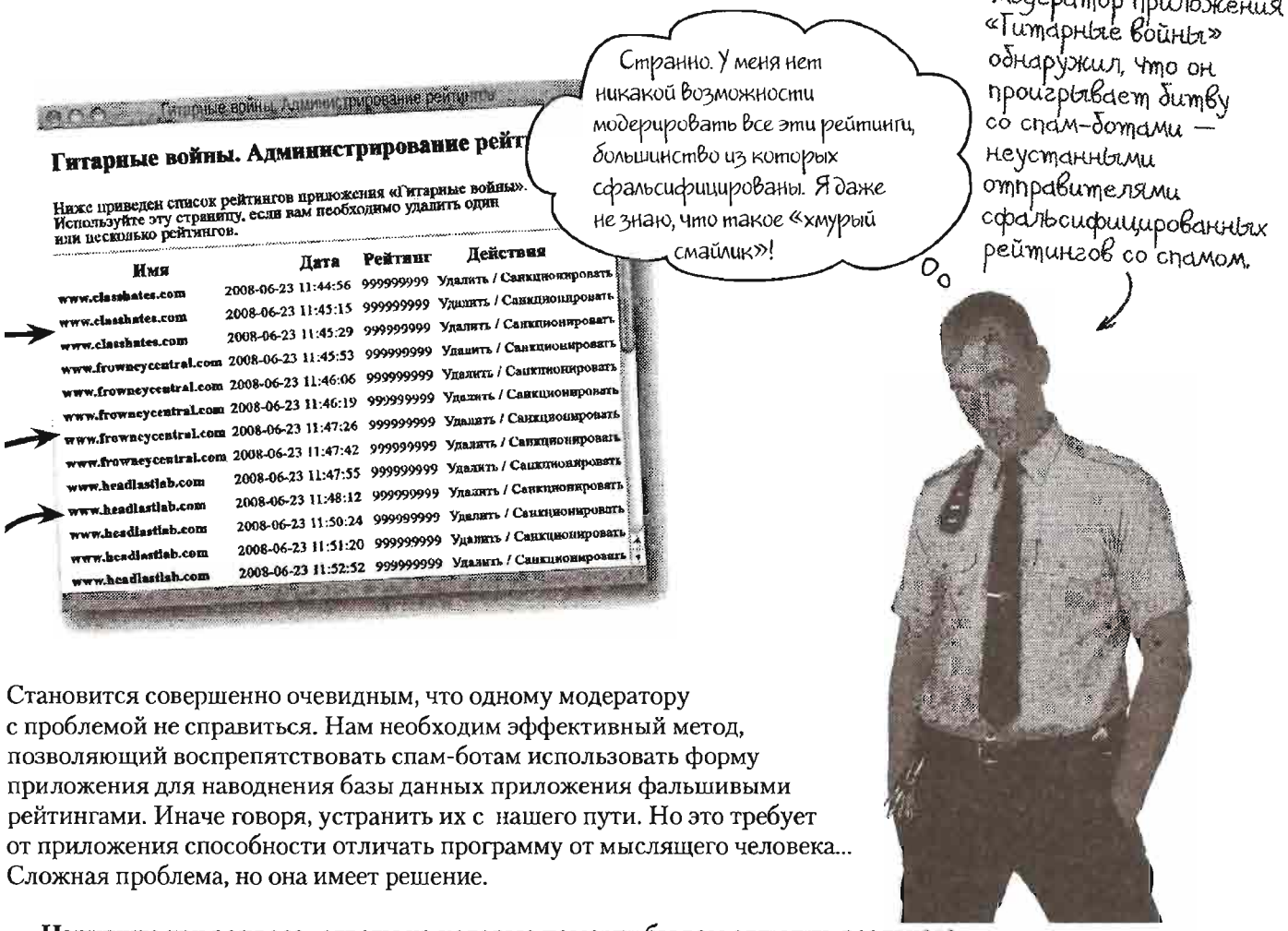
Любая веб-форма подвержена риску быть атакованной спам-ботами.



Спам-боты — идеальное средство для бессмысленного повторения одного и того же действия, в данном случае заполнения и передачи на сервер формы за формой рейтингов приложения «Гитарные войны», которые в действительности являются рекламой, а никакими не рейтингами.

Любая веб-форма подвержена риску

К счастью, в приложении «Гитарные войны» атаки спам-ботов видны модератору благодаря тем изменениям, которые мы внесли в код приложения, о чем говорилось в шестой главе книги. Тем не менее модератор совершенно ошеломлен количеством спама, посланного спам-ботами и в значительной степени затрудняющего просмотр и принятие решения о санкционировании или отклонении рейтингов гитарных воинов. Конечно, человек как модератор способен решить множество проблем, но ему трудно противостоять автоматическому генератору рекламы, который никогда не устает.

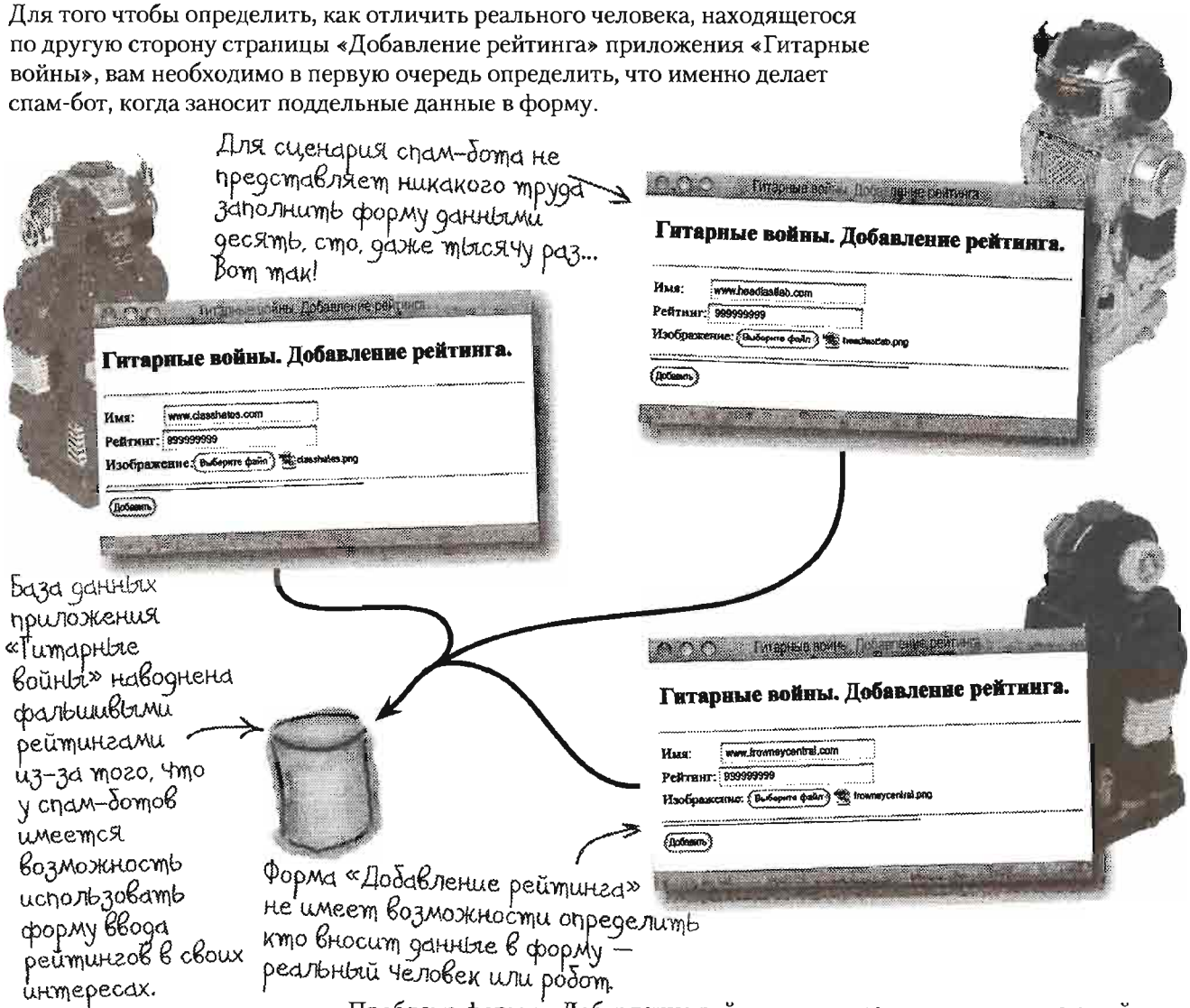


Становится совершенно очевидным, что одному модератору с проблемой не справиться. Нам необходим эффективный метод, позволяющий воспрепятствовать спам-ботам использовать форму приложения для наводнения базы данных приложения фальшивыми рейтингами. Иначе говоря, устранить их с нашего пути. Но это требует от приложения способности отличать программу от мыслящего человека... Сложная проблема, но она имеет решение.

Напишите три вопроса, ответы на которые помогли бы вам отличить реального, мыслящего человека от искусственного мозга робота:

Нам необходимо отличать людей от машин

Для того чтобы определить, как отличить реального человека, находящегося по другую сторону страницы «Добавление рейтинга» приложения «Гитарные войны», вам необходимо в первую очередь определить, что именно делает спам-бот, когда заносит поддельные данные в форму.



Форме «Добавление рейтинга» необходимо новое поле ввода данных, заполняемое человеком перед отправлением данных на сервер.

Проблема формы «Добавление рейтинга» заключается в том, что в ней не предусмотрено ничего, что могло бы предотвратить автоматическую отправку данных на сервер. В результате любой мало-мальски грамотный программист может создать программу-робота, которая будет многократно заносить данные в форму и отправлять их на сервер. Конечно, благодаря заботам модератора эти данные не попадут на главную страницу сайта, но это в значительной степени обесценивает его работу, так как ему приходится вручную удалять из базы сотни фальшивых данных.

В форме необходимо предусмотреть новое поле для ввода подтверждения, заполнение которого являлось бы необходимым условием для отправки данных на сервер. Но особенность этого поля должна заключаться в том, чтобы его заполнение было очень простым для **человека** и одновременно очень сложным для **программы**.

УПРАЖНЕНИЕ

Ниже перечислено несколько возможных полей ввода данных, которые потенциально могли бы использоваться, чтобы воспрепятствовать вводу в форму фальшивых данных спам-ботом. Укажите, использование каких из этих полей ввода, по вашему мнению, могло бы успешно и просто обеспечить передачу на сервер данных, введенных только человеком. Прокомментируйте ваши соображения.

Вы робот? ◯ **Да** ◯ **Нет**

Какое блюдо Элвис предпочитал всем остальным?

Сканирование радужной оболочки глаза: Посмотрите в веб-камеру и щелкните кнопкой мыши

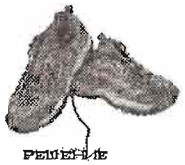
Введите символы, изображенные справа: kdyqmc

Сколько будет 7 + 5?

Что за животное изображено ниже?

Введите символы, изображенные справа: kdyqmc

Сканирование отпечатка большого пальца: Прижмите большой палец и щелкните кнопкой мыши



РЕШЕНИЕ К УПРАЖНЕНИЮ

Ниже перечислено несколько возможных полей ввода данных, которые потенциально могли бы использоваться, чтобы воспрепятствовать вводу в форму фальшивых данных спам-ботом. Укажите, использование каких из этих полей ввода, по вашему мнению, могло бы успешно и просто обеспечить передачу на сервер данных, введенных только человеком. Прокомментируйте ваши соображения.

Вы робот? ○ **Да** ○ **Нет**

Слишком легкий вопрос. Даже с вероятностью правильного ответа в 50 % (наугад) огромное количество фальшивых рейтингов будет принято.

Определенно сложный вопрос для робота, но не менее сложный и для многих людей. Далеко не все знают, что Элвис любил сэндвичи с ореховым маслом и бананом. Это также потребует достаточно объемной базы данных с подобными мелочными вопросами и ответами на них.

Какое блюдо Элвис предпочитал всем остальным?

Сканирование радужной болочки глаза: Посмотрите в веб-камеру и щелкните кнопкой мыши

Неплохое средство, если принять во внимание, что буквы идентификационной фразы выводятся в виде изображения, а не в виде текста, но потенциально преодолимое, если робот достаточно интеллектуален, чтобы использовать системы оптического распознавания текста.

Отличное средство для предотвращения атак спам-ботов, но технически трудновыполнимое и достаточно дорогое.

Введите символы, изображенные справа: kdyqmc

Сколько будет 7 + 5?

Просто и достаточно эффективно. Большинство роботов недостаточно интеллектуальны для того, чтобы проводить синтаксический анализ математических выражений. Остается надеяться, что большинство людей способны на это!

Что за животное изображено ниже?

Помните Фэнга, собаку, которая была похищена инопланетными пришельцами и о которой мы говорили ранее в этой книге?

Обманчиво-эффективное средство. Роботу, конечно, придется потрудиться, чтобы интерпретировать изображение, но использование этого метода потребует наличия базы данных изображений и ответов.

Искусное улучшение предыдущего варианта с вводом идентификационной фразы, но в этом случае буквы спрятаны за линиями и точками, чтобы затруднить работу систем автоматического распознавания текста.

Введите символы, изображенные справа: kdyqmc

Не настолько сложный метод, как сканирование сетчатки глаза, но все равно требующий специального оборудования и программного обеспечения.

Сканирование отпечатка большого пальца: Прижмите большой палец и щелкните кнопкой мыши

Мы можем победить автоматизацию с помощью автоматизации

Тест, позволяющий установить, что субъект, вводящий данные в форму, является живым человеком, известен под названием **CAPTCHA**, что является аббревиатурой английской фразы Completely Automated Public Turing Test to Tell Computers and Humans Apart (полностью автоматизированный открытый тест Тьюринга по распознаванию людей и машин). Можно много говорить по поводу любого теста, идеально соответствующего поставленной задаче. К настоящему времени разработано большое число интересных CAPTCHA-тестов, но один из наиболее широко используемых основан на генерации случайным образом идентификационной фразы, которую пользователь должен ввести в форму. Для того чтобы противостоять более интеллектуальным спам-ботам, снабженным системами оптического распознавания текста, символы идентификационной фразы искажаются точками и линиями, расположенными случайным образом, и частично прячутся за ними.

CAPTCHA — это программа, которая защищает сайт от роботов, используя определенный тест.

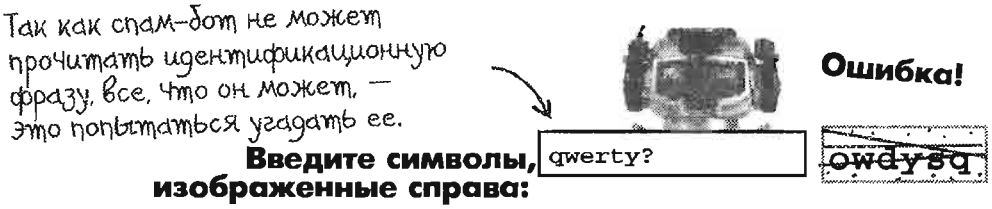
Так как буквы в идентификационной фразе генерируются случайным образом, при каждом вызове формы генерируется новая идентификационная фраза, отличная от всех прежних.

Для ввода идентификационной фразы CAPTCHA используется обычное поле для ввода текста.

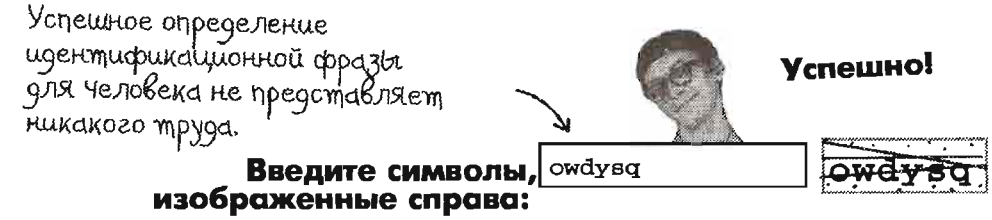
Линии и точки, генерируемые случайным образом, помогают сделать текст идентификационной фразы достаточно неразборчивым для систем оптического распознавания, в то же время позволяя людям понять его.

Поле ввода идентификационной фразы CAPTCHA точно такое же, как и любое другое поле ввода формы, с той лишь разницей, что его задачей является предотвращение передачи данных формы на сервер, если не введена правильная идентификационная фраза. Поэтому в отличие от других полей ввода данных, значения которых обычно передаются на сервер для обработки, значение поля идентификационной фразы CAPTCHA проверяется и используется для управления процессом передачи данных формы на сервер.

Так как спам-бот не может прочитать идентификационную фразу, все, что он может, — это попытаться угадать ее.



Успешное определение идентификационной фразы для человека не представляет никакого труда.



Очень важно, чтобы идентификационная фраза CAPTCHA выводилась в форме изображения, а не текста. В противном случае спам-боту не составит большого труда его прочитать.

Не бывает глупых вопросов

В: Мне понравилось это изображение собачки в CAPTCHA. Можно ли использовать его вместо идентификационной фразы CAPTCHA?

О: Вполне. Только имейте в виду, что в этом случае вам придется создать базу данных изображений и их описаний, потому что основная задача CAPTCHA — это проверка. Надежная проверка должна опираться на достаточно объемный выбор тестов, которые не будут часто повторяться при многократном вызове формы. В этом — преимущество CAPTCHA, основанного на проверке идентификационной фразы, так как она в данном случае генерируется как совокупность случайно выбранных букв. Вероятность того, что один и тот же текст будет выведен одному и тому же пользователю дважды, достаточно мала даже при большом количестве попыток.

В: Насколько велика зависимость от CAPTCHA людей с ослабленным зрением? Что если они не смогут пройти этот тест?

О: Визуальный CAPTCHA — не лучший способ проверки для людей с ослабленным зрением. Идеальным решением в этих случаях могло бы быть создание звукового CAPTCHA, при котором приложение зачитывало бы несколько цифр, а пользователь должен был бы ввести их в форму, чтобы пройти тест. Но здесь опять возникает проблема, связанная с тем, что достаточно интеллектуальный робот, используя системы распознавания речи, сможет пройти тест и преодолеть защиту. Поэтому в некоторых из таких систем используется умышленное искажение звука, чтобы затруднить его распознавание. Подобно визуальным CAPTCHA, звуковые его версии должны располагать объемной базой звуковых клипов с соответствующими их описаниями. Существуют сервисы CAPTCHA, которые предлагают гибкие тесты, использующие как возможности изображений, так и звука, например www.captcha.net. Такие сервисы обладают тем преимуществом, что используют новейшие технологии, но встраивание их в приложение обычно проходит не так гладко, как встраивание тестов, разработанных специально для вашего веб-приложения.

В: Но ведь есть также люди, у которых ослаблено не только зрение, но и слух. Как помочь им?

О: В конечном счете CAPTCHA всегда приходится находить компромисс: с одной стороны, не дать пройти тест роботу, с другой — не ужесточить этот тест в такой степени, что его будет трудно пройти человеку. Подобно компьютерным вирусам и антивирусным программам, спам-боты и CAPTCHA ведут друг с другом постоянную борьбу, напоминающую игру в кошки-мышки, в которой спам-бот усовершенствуется, чтобы преодолеть CAPTCHA, что вызывает необходимость усовершенствовать CAPTCHA, и т. д. Под этот перекрестный огонь попадает пользователь, который может не получить доступа к приложению из-за ограниченных возможностей некоторых CAPTCHA. Поэтому разработчику веб-приложения нужно очень тщательно взвешивать и сопоставлять риск успешной атаки спам-бота и возможность потери пользователя, который не смог получить доступа к части сайта, защищенной с помощью CAPTCHA. В качестве утешения может служить тот факт, что большинство мощных спам-ботов обычно нацелены на крупные сайты, так как именно с такими сайтами они связывают наибольшие доходы от рекламы. Поэтому вы, возможно, и не столкнетесь с действительно злостным спам-ботом до тех пор, пока ваш сайт не станет настолько большим, чтобы превратиться в желанную цель для вооруженных до зубов спам-ботов.

В PHP имеется возможность динамически генерировать изображения, которые затем могут выводиться на экран средствами HTML.

С помощью графической библиотеки, называемой GD (Graphic Draw, что может быть переведено на русский язык как «Рисование графических элементов»), наш сценарий может динамически создавать изображения в популярных форматах GIF, JPEG и PNG и либо выводить их в окне приложения, либо сохранять в файле на сервере. Эта возможность PHP исключительно важна, так как нельзя рисовать изображения средствами чистого HTML. PHP позволяет вам нарисовать часть страницы с использованием функций графической библиотеки и затем вывести это изображение с помощью уже знакомого вам тега `<img>`.

Создание текста идентификационной фразы CAPTCHA

Прежде чем мы начнем даже думать о графической стороне идентификационной фразы CAPTCHA, нам нужно решить, как мы будем создавать саму идентификационную фразу в виде последовательности случайно выбранных символов в текстовой форме. Идентификационная фраза может состоять из любого количества символов, но обычно считается, что последовательности из шести-восьми символов вполне достаточно. Мы можем определить длину идентификационной фразы как константу, что позволит нам легко изменить ее значение, если в этом возникнет необходимость.

```
define('CAPTCHA_NUMCHARS', 6);
```

Длины идентификационной фразы CAPTCHA в шесть символов вполне достаточно, чтобы поставить преграду спам-боту и не слишком раздражать пользователей.

Итак, как именно мы сможем сгенерировать последовательность, состоящую из шести случайным образом выбранных символов? Здесь на сцену выходят еще две встроенные функции PHP: `rand()` и `chr()`. Функция `rand()` возвращает случайное число, находящееся в диапазоне, границы которого определены двумя аргументами, переданными функции при ее вызове. Функция `chr()` преобразует числовой ASCII в соответствующий ему символ (American Standard Code for Information Interchange — американский стандартный код обмена информацией —7-битный код для представления десятичных цифр, латинского алфавита, знаков препинания и управляющих символов). Нас интересует ASCII-коды только в диапазоне 97–122, которые представляют строчные буквы латинского алфавита от «a» до «z». Если мы вызовем функцию `rand()` в этом диапазоне шесть раз, мы выберем случайным образом ASCII-коды шести строчных букв латинского алфавита.

`$pass_phrase`

Проход цикла для генерации каждого символа идентификационной фразы.

```
// Создание идентификационной фразы, состоящей из шести
// случайно выбранных строчных букв латинского алфавита
$pass_phrase = "";
for ($i = 0; $i < CAPTCHA_NUMCHARS; $i++) {
   $pass_phrase .= chr(rand(97, 122));
}
```

Идентификационная фраза создается по одному случайно выбираемому символу за один проход цикла.

Этот код в конечном итоге перейдет в свой собственный многократно используемый сценарий captcha.php.

`rand()`

Эта встроенная функция возвращает случайное целое число со значением, находящимся или в определенном диапазоне, или в диапазоне от нуля до значения встроенной константы `RAND_MAX` (оно зависит от сервера). Чтобы получить случайное число внутри определенного диапазона, просто передайте функции в виде аргументов значения минимальной и максимальной границ диапазона.

`chr()`

Эта встроенная функция рассматривает целое число, переданное ей в качестве аргумента, как ASCII-код символа и возвращает этот символ. Например, число 97 является ASCII-кодом латинской строчной буквы «a». В результате вызова функции **`chr(97)`** будет возвращена латинская прописная буква «A».

Функция rand() возвращает случайное целое число со значением, находящимся в определенном диапазоне.

Вывод изображения CAPTCHA

Разобравшись с текстом идентификационной фразы, мы можем двигаться дальше и генерировать изображение, состоящее из этого текста на фоне линий и точек, задача которых состоит в том, чтобы сделать идентификационную фразу неразборчивой для спам-бота. Но с чего начать? Первое, что мы должны сделать, — это определить, каких размеров должно быть изображение CAPTCHA. Зная, что это изображение должно располагаться возле поля ввода идентификационной фразы, имеет смысл сделать его небольшим. Давайте примем значения ширины и высоты этого изображения равными 100 и 25 и сохраним их в виде констант, чтобы определить только в одном месте и при необходимости их изменения это не вызывало бы особых проблем.

Создание динамических изображений в PHP может быть осуществлено с использованием функций библиотеки GD.

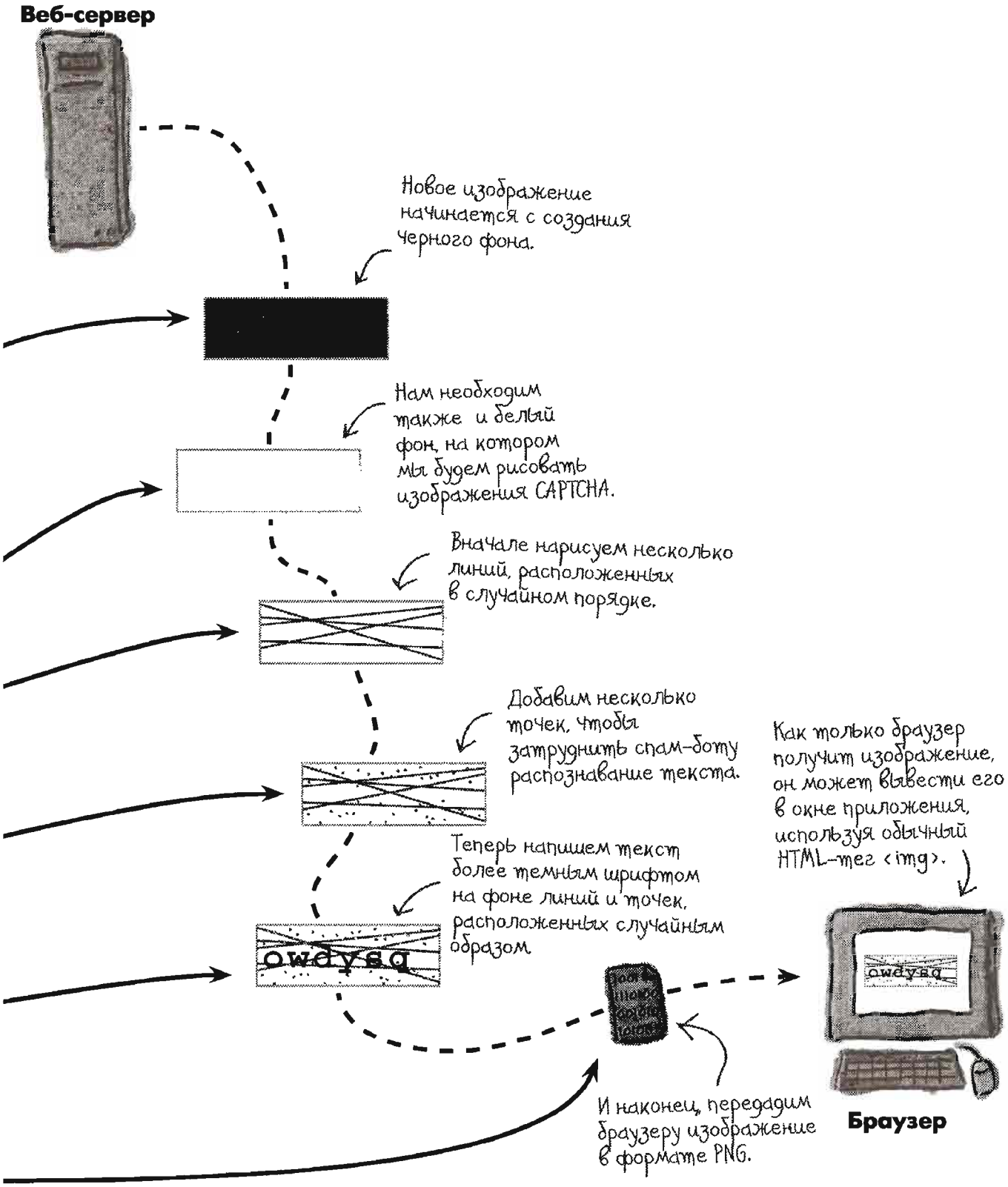
```
define('CAPTCHA_WIDTH', 100);
define('CAPTCHA_HEIGHT', 25);
```

Размеры изображения CAPTCHA сохранены в константах для того, чтобы в случае необходимости их изменение не вызывало проблем.

Создание изображения CAPTCHA требует вызова нескольких функций библиотеки GD, при этом все они работают с изображением в оперативной памяти компьютера. Иначе говоря, вы создаете изображение в оперативной памяти компьютера, затем рисуете в нем необходимые элементы и после того, как все закончено, передаете его браузеру для вывода в окне приложения.

В результате выполнения этого кода с помощью других функций библиотеки GD создаются цвета для различных элементов изображения.

```
// Создание изображения
$img = imagecreatetruecolor(CAPTCHA_WIDTH, CAPTCHA_HEIGHT);

// Установка цветов: белого для фона, черного для текста и серого для графики
$bg_color = imagecolorallocate($img, 255, 255, 255);     // белый цвет
$text_color = imagecolorallocate($img, 0, 0, 0);         // черный цвет
$graphic_color = imagecolorallocate($img, 64, 64, 64);   // серый цвет

// Заполнение фона
imagefilledrectangle($img, 0, 0, CAPTCHA_WIDTH, CAPTCHA_HEIGHT, $bg_color);

// Рисование нескольких линий, расположенных случайным образом
for ($i = 0; $i < 5; $i++) {
  imageline($img, 0, rand() % CAPTCHA_HEIGHT, CAPTCHA_WIDTH,
    rand() % CAPTCHA_HEIGHT, $graphic_color);
}

// Рисование нескольких точек, расположенных случайным образом
for ($i = 0; $i < 50; $i++) {
  imagesetpixel($img, rand() % CAPTCHA_WIDTH,
    rand() % CAPTCHA_HEIGHT, $graphic_color);
}

// Написание текста идентификационной фразы
imagettftext($img, 18, 0, 5, CAPTCHA_HEIGHT - 5, $text_color,
  'Courier New Bold.ttf', $pass_phrase);

// Вывод изображения в PNG-формате с использованием заголовка
header("Content-type: image/png");
imagepng($img);
```


Веб-сервер
Новое изображение начинается с создания черного фона.
Нам необходим также и белый фон, на котором мы будем рисовать изображения CAPTCHA.
Вначале нарисуем несколько линий, расположенных в случайном порядке.
Добавим несколько точек, чтобы затруднить спам-боту распознавание текста.
Теперь напишем текст более темным шрифтом на фоне линий и точек, расположенных случайным образом.
owdysq
Как только браузер получит изображение, он может вывести его в окне приложения, используя обычный HTML-тег <img>.
owdysq
И наконец, передадим браузеру изображение в формате PNG.
Браузер

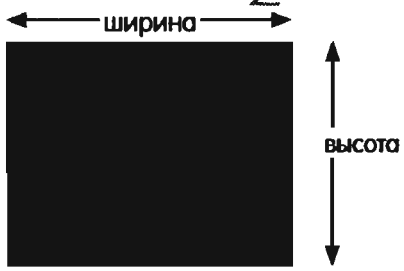
Внутри функций графической библиотеки GD

Секрет создания изображения CAPTCHA кроется в возможностях графической библиотеки GD, которая, как вы уже узнали, содержит функции, позволяющие рисовать различные элементы изображения в коде PHP. Давайте рассмотрим некоторые из этих функций более подробно, чтобы понять, как с их помощью генерируется изображение CAPTCHA.

imagecreatetruecolor()

Эта функция создает в памяти пустое изображение, готовое к тому, чтобы в него помещались различные элементы с помощью других функций графической библиотеки GD. Два аргумента, передаваемые этой функции при ее вызове, — это ширина и высота изображения. Изображение создается с черным фоном, поэтому обычно приходится менять цвет фона, например на белый, прежде чем вы начнете рисование элементов изображения. Вы можете добиться этого вызовом функции `imagefilledrectangle()`.Функция `imagecreatetruecolor()` возвращает идентификатор изображения. Его необходимо передавать большинству функций графической библиотеки GD в качестве первого аргумента, чтобы определить изображение, в котором эти функции должны рисовать различные элементы.

Новое изображение первоначально создается с фоном черного цвета.



Ширина нового изображения в пикселах.

Высота нового изображения в пикселах.

```
$img = imagecreatetruecolor(CAPTCHA_WIDTH, CAPTCHA_HEIGHT);
```

Функция возвращает идентификатор изображения, который необходим другим функциям графической библиотеки GD для того, чтобы рисовать различные элементы.

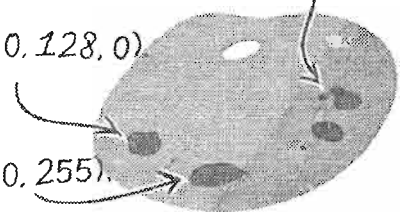
В результате выполнения этого кода будет создано изображение размером 100x25 в соответствии со значениями наших констант.

imagecolorallocate()

Используйте эту функцию для того, чтобы определить цвета, которые вы будете применять при вызове других функций графической библиотеки GD. В качестве первого аргумента функции передается идентификатор изображения. Три последующих аргумента — это значения трех компонентов аддитивной цветовой модели RGB (наименование этой цветовой модели образовано из начальных букв трех английских слов Red, Green и Blue, которые переводятся на русский язык как «красный», «зеленый» и «синий»). Каждый из компонентов может принимать значение в диапазоне от 0 до 255. Функция возвращает идентификатор цвета, который может быть использован для указания цвета (часто в качестве последнего аргумента) при вызове других функций графической библиотеки GD.

Красный цвет: (255, 0, 0).

Зеленый цвет: (0, 128, 0).



Синий цвет: (0, 0, 255).

Возвращаемое значение — это идентификатор цвета, который может быть использован для указания цвета при вызове других функций графической библиотеки GD, например для указания цвета текста идентификационной фразы CAPTCHA.

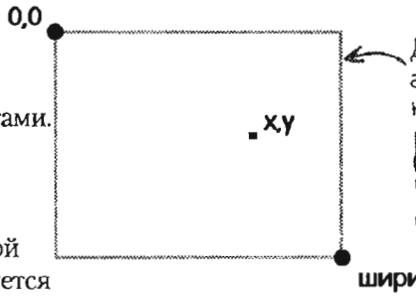
Идентификатор изображения, с которым будет использоваться данный цвет.

```
$text_color = imagecolorallocate($img, 0, 0, 0);
```

Значения красного, зеленого и синего компонентов аддитивной цветовой модели RGB. В данном случае совокупность этих компонентов определяет черный цвет.

imagesetpixel()

Эта функция рисует один пиксел внутри изображения в точке с определенными координатами. Начало координатной сетки (0,0) располагается в левом верхнем углу изображения. Координаты увеличиваются при движении вправо и вниз. Как и в большинстве других функций графической библиотеки GD, для рисования пиксела используется цвет, переданный функции с последним аргументом.



Для большинства функций графической библиотеки GD начало координатной сетки располагается в левом верхнем углу изображения. Координаты увеличиваются при движении вправо и вниз.

```
imagesetpixel($img, rand() % CAPTCHA_WIDTH, rand() % CAPTCHA_HEIGHT, $graphic_color);
```

Идентификатор изображения, в котором будет нарисован пиксел.

Координаты XY пиксела относительно левого верхнего угла изображения. В данном случае эти координаты выбраны случайным образом в диапазоне значений ширины и высоты изображения CAPTCHA.

Цвет (идентификатор) пиксела.

imageline()

Вызывайте эту функцию для того, чтобы нарисовать линию между точками с координатами X1,Y1 и X2,Y2. Координаты указываются относительно левого верхнего угла изображения, а цвет линии определяется значением последнего аргумента, переданного функции при вызове.

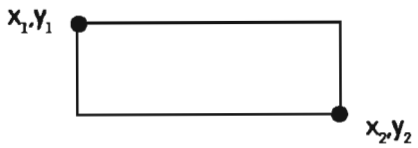
Координаты XY начальной точки линии в этом случае соответствуют координатам левого верхнего угла изображения CAPTCHA.

```
imageline($img, 0, rand() % CAPTCHA_HEIGHT,
  CAPTCHA_WIDTH, rand() % CAPTCHA_HEIGHT, $graphic_color);
```

Координаты XY конечной точки линии соответствуют координатам правого нижнего угла изображения CAPTCHA.

imagerectangle()

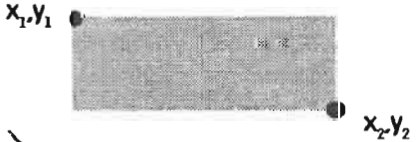
Рисует прямоугольник определенного цвета с координатами левого верхнего угла X1,Y1 и правого нижнего — X2,Y2. Эти координаты передаются функции в числе шести аргументов сразу после аргумента, определяющего идентификатор изображения.



imagefilledrectangle()

Аналогично функции `imagerectangle()` эта функция рисует прямоугольник с цветом фона, определенного значением последнего аргумента.

Функция imagerectangle() принимает те же самые аргументы, что и функция imagefilledrectangle().




```
imagefilledrectangle($img, 0, 0, CAPTCHA_WIDTH, CAPTCHA_HEIGHT, $bg_color);
```

Координаты XY точек левого верхнего и правого нижнего углов прямоугольника, в котором располагается изображение CAPTCHA.

Функции графической библиотеки GD. Продолжение...

imageellipse()

Для рисования окружностей и эллипсов этой функции необходимо передать в качестве аргументов координаты центра, а также ширину и высоту фигуры. Истинная окружность — это всего лишь эллипс с одинаковыми значениями ширины и высоты. Цвет окружности/эллипса определяется значением последнего аргумента, переданного функции при вызове.

Эллипсы не используются при создании изображений CAPTCHA, но, несмотря на это, они очень удобны.

imagefilledellipse()

Хотите эллипс с цветным фоном? Просто вызовите функцию `imagefilledellipse()`, которая работает точно так же, как и функция `imageellipse()`, с той лишь разницей, что цвет, который вы ей передаете в качестве аргумента, используется при прорисовке фона, а не контура фигуры.

```
imagefilledellipse($img, 0, 0, 320, 240, $color);
```

Функции `imageellipse()` и `imagefilledellipse()` принимают одни и те же аргументы.

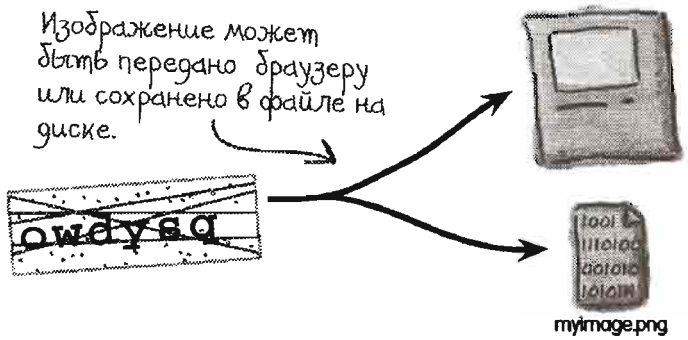
Это ширина и высота эллипса. Задайте их равными друг другу, если вам необходимо нарисовать окружность.

Это координаты XY центра эллипса.

imagepng()

Как только вы закончили рисовать все элементы изображения, вы можете передать его браузеру или сохранить в файле на диске, вызвав эту функцию. В любом случае в результате этих действий вы получаете изображение, которое может быть использовано в HTML-теге `<img>` для вывода в окно приложения. Если вы выберете вариант с созданием изображения в формате PNG непосредственно в памяти (то есть у вас не будет никакого имени файла, связанного с этим изображением), то вам необходимо предварительно вызвать функцию `header()`, чтобы передать это изображение браузеру через HTTP-заголовок.

Изображение может быть передано браузеру или сохранено в файле на диске.



Это идентификатор изображения, который вы использовали в других функциях графической библиотеки GD.

```
imagepng($img);
```

Функция возвращает `true` или `false` в зависимости от того, насколько успешно прошла операция.

Вы можете передать функции в качестве второго, необязательного аргумента имя файла. Без этого аргумента функция генерирует изображение в памяти, и оно может быть передано браузеру через HTTP-заголовок.

Удаление изображения и освобождение ресурсов сервера после того, как у вас отпала надобность в этом изображении, — это хороший стиль программирования.

imagedestroy()

Для работы с изображениями с использованием функций графической библиотеки GD необходимы системные ресурсы. Вызов этой функции после того, как вы закончили работу с изображением, приводит к освобождению этих ресурсов. Вызывайте ее всякий раз после того, как вы передали изображение с помощью функции `imagepng()`.

Подобно функции imagepng() эта функция возвращает true или false в зависимости от того, насколько успешно прошла операция.

```
imagedestroy($img);
```

Идентификатор изображения, которое вы хотите удалить.

Старайтесь всегда использовать эту функцию при работе с каждым из изображений, которые вы создаете, чтобы все изображения были удалены после использования.

Всегда удаляйте изображения из памяти с помощью функции imagedestroy() после того, как вы закончили с ними работу.

imagestring()

Эта функция выводит строку текста в указанном цвете, используя встроенный в PHP шрифт. В дополнение к идентификатору изображения вы передаете функции размер шрифта в виде числа (1–5) вместе с координатами левого верхнего угла строки текста, самой строкой и цветом.

Число в диапазоне от 1 до 5 устанавливает размер шрифта, используемого для написания текста. Число 5 соответствует наибольшему размеру.

Встроенный шрифт соответствует базовому шрифту, но ограничен в своих размерах.

Размер шрифта строки текста в диапазоне от 1 до 5.

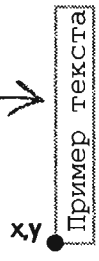
```
imagestring($img, 3, 75, 75, 'Пример текста', $color);
```

Это координаты XY левого верхнего угла строки текста..

Это строка текста, которая должна быть выведена на экран

Цвет текста.

Текст, выведенный с помощью функции imagestringup(), повернут на 90 градусов против часовой стрелки, то есть расположен вертикально.



imagestringup()

Аналогично функции `imagestring()` эта функция выводит строку текста в указанном цвете, используя встроенный в PHP шрифт, но текст выводится вертикально, как будто повернутый на 90 градусов против часовой стрелки. Функция вызывается с теми же самыми аргументами, что и функция `imagestring()`.

Использование в тексте шрифтов True Type

Функция `imagestring()` проста в использовании, но она очень ограничена в возможностях управления параметрами шрифтов. Для того чтобы разнообразить виды шрифтов, вам необходимо использовать шрифты True Type. Идентификационная фраза CAPTCHA — хороший пример того, где это действительно необходимо, потому что ее символы должны быть достаточно крупными и выводиться жирным шрифтом. Чтобы достигнуть этого эффекта, вам необходима еще одна функция графической библиотеки GD, которая выводит текст, используя шрифт True Type, доступный на вашем сервере.

`imagettftext()`

Для того чтобы иметь возможность выводить текст различного вида и гибко управлять им, установите на вашем веб-сервере шрифт True Type и затем вызовите эту функцию. Вы получите возможность не только выбирать вид шрифта по вашему усмотрению, но также гибко управлять его размерами и даже углом наклона, под которым шрифт будет выводиться. В отличие от функции `imagestring()` координаты, передаваемые этой функции в качестве аргументов, определяют «базовую точку» первого символа выводимого текста, которая приблизительно соответствует координатам его левого нижнего угла.

Эта функция требует обязательного присутствия на сервере файла шрифта True Type, имя которого вы должны передать функции в качестве ее предпоследнего аргумента. Файлы шрифтов True Type обычно имеют расширение .TTF.

Для вывода текста в нестандартном формате используются шрифты True Type и функция imagettftext().

x,y Sample text

В отличие от функции imagestring() начало координат при выводе текста с помощью функции imagettftext() соответствует нижнему левому углу текста.

Размер шрифта, обычно указываемый в «пунктах».

Угол наклона шрифта в направлении против часовой стрелки (0 означает нормальный текст).

Координаты XY левого нижнего угла текста.

```
imagettftext($img, 18, 0, 5, CAPTCHA_HEIGHT - 5, $text_color,
  'Courier New Bold.ttf', $pass_phrase);
```

Выводимый текст

Вы должны обязательно установить шрифт True Type на вашем веб-сервере так, чтобы графическая библиотека имела к нему доступ.

Courier New Bold.ttf

Чокнутые биты

Если у вас есть желание заняться созданием своего собственного шрифта True Type, чтобы еще тоньше настраивать свой CAPTCHA, посетите сайт `www.fontstruct.com`. Этот сайт интернет-сообщества создателей шрифтов включает веб-инструментарий для создания собственных шрифтов.

Используйте функцию imagettftext() для гибкого управления параметрами выводимого текста с использованием шрифтов True Type.

РИСУЙ ЧТО К ЧЕМУ

imagecolorallocate($img, 128, 128, 128);

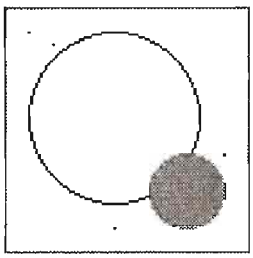
Найдите соответствие фрагментов PHP-кода, приведенных слева, изображениям, расположенным справа. Считайте, что изображение (`$img`) и цвета (`$black_color`, `$white_color` и `$gray_color`) уже созданы.

imagecolorallocate($img, 0, 0, 0);

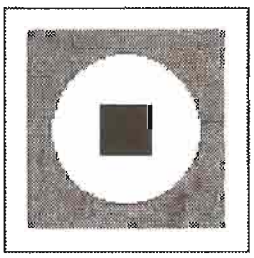
imagecolorallocate($img, 255, 255, 255);

```
imagefilledrectangle($img, 10, 10, 90, 90, $gray_color);
imagefilledellipse($img, 50, 50, 60, 60, $white_color);
imagefilledrectangle($img, 40, 40, 60, 60, $black_color);
```

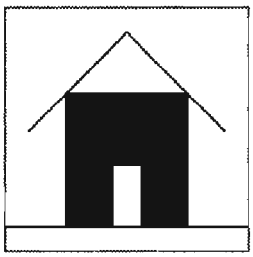
```
imageline($img, 15, 15, 50, 50, $black_color);
imageline($img, 15, 85, 50, 50, $black_color);
imageline($img, 50, 50, 85, 50, $black_color);
imagefilledellipse($img, 15, 15, 20, 20, $gray_color);
imagefilledellipse($img, 15, 85, 20, 20, $gray_color);
imagefilledellipse($img, 50, 50, 20, 20, $gray_color);
imagefilledellipse($img, 85, 50, 20, 20, $gray_color);
```



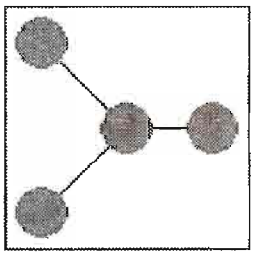
```
imagefilledrectangle($img, 10, 10, 90, 60, $gray_color);
imagesetpixel($img, 30, 25, $black_color);
imagesetpixel($img, 70, 25, $black_color);
imageline($img, 35, 45, 65, 45, $black_color);
imagefilledrectangle($img, 45, 50, 55, 90, $gray_color);
```



```
imageellipse($img, 45, 45, 70, 70, $black_color);
imagefilledellipse($img, 75, 75, 30, 30, $gray_color);
imagesetpixel($img, 10, 10, $black_color);
imagesetpixel($img, 80, 15, $black_color);
imagesetpixel($img, 20, 15, $black_color);
imagesetpixel($img, 90, 60, $black_color);
imagesetpixel($img, 20, 80, $black_color);
imagesetpixel($img, 45, 90, $black_color);
```

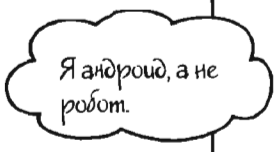


```
imagefilledrectangle($img, 25, 35, 75, 90, $black_color);
imageline($img, 10, 50, 50, 10, $black_color);
imageline($img, 50, 10, 90, 50, $black_color);
imagefilledrectangle($img, 45, 65, 55, 90, $white_color);
imageline($img, 0, 90, 100, 90, $black_color);
```



РИСУЕТ ЧТО К ЧЕМУ — РЕШЕНИЕ

Найдите соответствие фрагментов PHP-кода, приведенных слева, изображениям, расположенным справа. Считайте, что изображение (`$img`) и цвета (`$black_color`, `$white_color` и `$gray_color`) уже созданы.



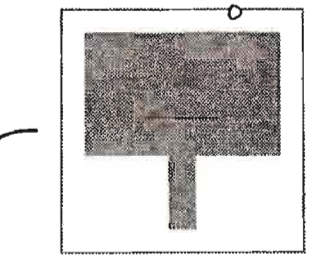

```
imagefilledrectangle($img, 10, 10, 90, 90, $gray_color);
imagefilledellipse($img, 50, 50, 60, 60, $white_color);
imagefilledrectangle($img, 40, 40, 60, 60, $black_color);
```

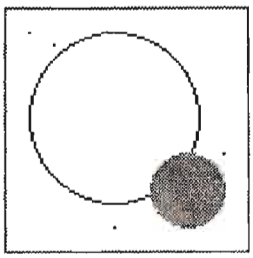
```
imageline($img, 15, 15, 50, 50, $black_color);
imageline($img, 15, 85, 50, 50, $black_color);
imageline($img, 50, 50, 85, 50, $black_color);
imagefilledellipse($img, 15, 15, 20, 20, $gray_color);
imagefilledellipse($img, 15, 85, 20, 20, $gray_color);
imagefilledellipse($img, 50, 50, 20, 20, $gray_color);
imagefilledellipse($img, 85, 50, 20, 20, $gray_color);
```

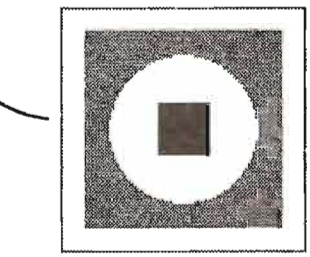
```
imagefilledrectangle($img, 10, 10, 90, 60, $gray_color);
imagesetpixel($img, 30, 25, $black_color);
imagesetpixel($img, 70, 25, $black_color);
imageline($img, 35, 45, 65, 45, $black_color);
imagefilledrectangle($img, 45, 50, 55, 90, $gray_color);
```

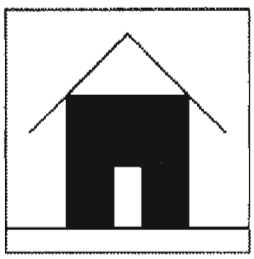
```
imageellipse($img, 45, 45, 70, 70, $black_color);
imagefilledellipse($img, 75, 75, 30, 30, $gray_color);
imagesetpixel($img, 10, 10, $black_color);
imagesetpixel($img, 80, 15, $black_color);
imagesetpixel($img, 20, 15, $black_color);
imagesetpixel($img, 90, 60, $black_color);
imagesetpixel($img, 20, 80, $black_color);
imagesetpixel($img, 45, 90, $black_color);
```

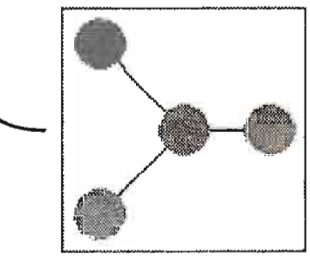
```
imagefilledrectangle($img, 25, 35, 75, 90, $black_color);
imageline($img, 10, 50, 50, 10, $black_color);
imageline($img, 50, 10, 90, 50, $black_color);
imagefilledrectangle($img, 45, 65, 55, 90, $white_color);
imageline($img, 0, 90, 100, 90, $black_color);
```











Создание случайного изображения CAPTCHA

В результате сборки всего кода CAPTCHA мы получаем новый сценарий `captcha.php`, в результате выполнения которого случайным образом генерируется идентификационная фраза. Вместе с дополнительными элементами она передается браузеру в виде PNG-изображения.

Сценарий captcha.php совершен самостоятелен. Вы можете открыть его в браузере и наблюдать генерируемое им изображение.

```
<?php
 session_start();

 // Определение нескольких важных констант CAPTCHA
 define('CAPTCHA_NUMCHARS', 6); // количество символов в идентификационной фразе
 define('CAPTCHA_WIDTH', 100);  // ширина изображения
 define('CAPTCHA_HEIGHT', 25);  // высота изображения

 // Создание идентификационной фразы случайным образом
 $pass_phrase = "";
 for ($i = 0; $i < CAPTCHA_NUMCHARS; $i++) {
  $pass_phrase .= chr(rand(97, 122));
 }

 // Сохранение идентификационной фразы в переменной сессии в зашифрованном виде
 $_SESSION['pass_phrase'] = sha1($pass_phrase);

 // Создание изображения
 $img = imagecreatetruecolor(CAPTCHA_WIDTH, CAPTCHA_HEIGHT);

 // Установка цветов: белого для фона, черного для текста и серого для графических элементов
 $bg_color = imagecolorallocate($img, 255, 255, 255);    // белый цвет
 $text_color = imagecolorallocate($img, 0, 0, 0);        // черный цвет
 $graphic_color = imagecolorallocate($img, 64, 64, 64);  // темно-серый цвет

 // Заполнение фона
 imagefilledrectangle($img, 0, 0, CAPTCHA_WIDTH, CAPTCHA_HEIGHT, $bg_color);

 // Рисование линий, расположенных случайным образом
 for ($i = 0; $i < 5; $i++) {
  imageline($img, 0, rand() % CAPTCHA_HEIGHT, CAPTCHA_WIDTH, rand() % CAPTCHA_HEIGHT, $graphic_color);
 }

 // Рисование точек, расположенных случайным образом
 for ($i = 0; $i < 50; $i++) {
  imagesetpixel($img, rand() % CAPTCHA_WIDTH, rand() % CAPTCHA_HEIGHT, $graphic_color);
 }

 // Написание строки, содержащей идентификационную фразу
 imagettftext($img, 18, 0, 5, CAPTCHA_HEIGHT - 5, $text_color, "Courier New Bold.ttf", $pass_phrase);
 // Вывод изображения в формате PNG с помощью HTTP-заголовка
 header("Content-type: image/png");
 imagepng($img);

 // Удаление изображения
 imagedestroy($img);
?>
```

Создание констант, содержащих значения количества символов в идентификационной фразе, ширины и высоты изображения CAPTCHA.

Хотя вы могли бы сохранить идентификационную фразу в базе данных в зашифрованном виде, проще сохранить ее в переменной сессии. Мы должны сохранить ее в любом виде для того, чтобы сценарий «Добавление рейтинга» мог получить к ней доступ.

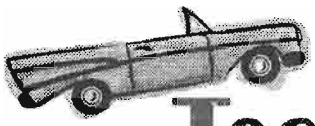
Некоторые версии графической библиотеки GD требуют указания относительного пути к файлу шрифта True Type, например ./Courier New Bold.ttf.

Изображение в формате PNG передается браузеру через HTTP-заголовок.

Создание PNG-изображения, использующего все нарисованные прежде элементы.

После окончания работы с изображением (отправки его браузеру) необходимо удалить его из памяти и освободить все связанные с ним ресурсы.

captcha.php

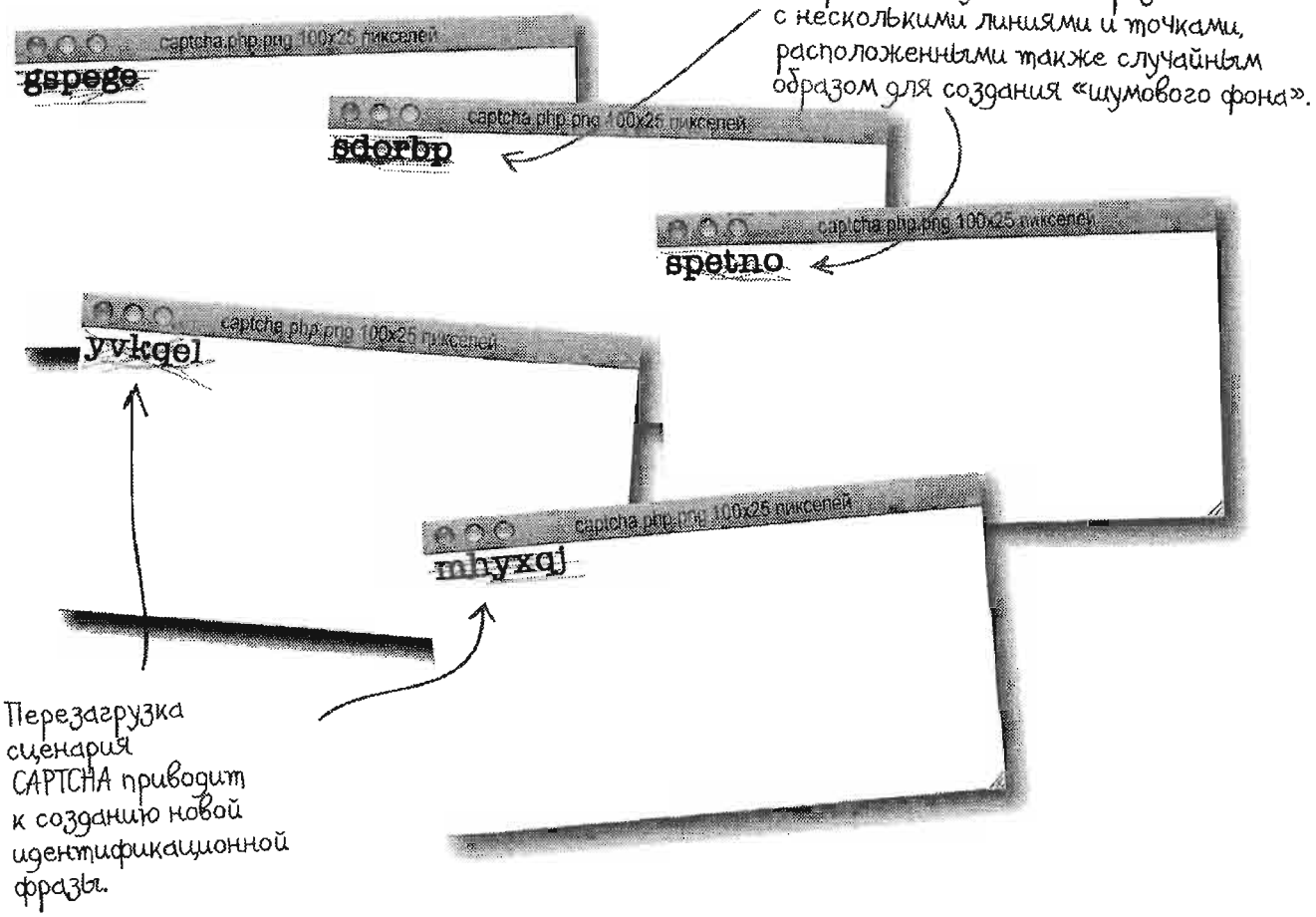


Тест-драйв

Создайте сценарий CAPTCHA и проверьте его работу.

Создайте новый текстовый файл с именем `captcha.php` и поместите в него код, приведенный на предыдущей странице (или загрузите сценарий с сайта по адресу `www.headfirstlabs.com/books/hfphp`).

Загрузите сценарий на ваш веб-сервер и откройте его в браузере. Вы немедленно увидите изображение CAPTCHA с идентификационной фразой, составленной случайным образом. Для того чтобы создать новую идентификационную фразу, обновите страницу.



Приведение в порядок приложения «Гитарные войны»

Теперь, когда мы узнали о возможностях некоторых функций графической библиотеки GD в создании изображения CAPTCHA, наступило время использовать изображение CAPTCHA для защиты модератора приложения «Гитарные войны» от атак спам-ботов. Решение этой проблемы с помощью проверки идентификационной фразы CAPTCHA включает несколько этапов. К счастью, два из них мы уже выполнили. Это генерация идентификационной фразы и создание изображения CAPTCHA. Давайте выполним и оставшиеся два этапа, чтобы освободить приложение «Гитарные войны» от засилья спам-ботов!

Модератор приложения «Гитарные войны» доведен до состояния нервного срыва, он пытается даже угрожать воображаемым роботам палкой. Ему необходима срочная помощь!

Уже сделано!

1. ~~**Генерация идентификационной фразы, составленной из случайно выбранных букв латинского алфавита.**~~

Изображение создано!

2. ~~**Создание изображения CAPTCHA с использованием идентификационной фразы.**~~

3. **Вывод изображения CAPTCHA в форме «Добавление рейтинга» приложения «Гитарные войны» с предложением пользователю ввести в поле формы идентификационную фразу.**

4. **Проверка правильности идентификационной фразы, введенной пользователем.**

УПРАЖНЕНИЕ

Выполните этап 3 создания CAPTCHA в форме «Добавление рейтинга» приложения «Гитарные войны» и составьте HTML-код создания нового поля ввода идентификационной фразы для проверки. Не забудьте про статический текст, предлагающий пользователю ввести идентификационную фразу, которую он видит в изображении CAPTCHA, созданном в результате вызова сценария `capthca.php` и выведенном в форме с помощью тега `<img>`.

упражнение решение

РЕШЕНИЕ К УПРАЖНЕНИЮ

Выполните этап 3 создания CAPTCHA в форме «Добавление рейтинга» приложения «Гитарные войны» и составьте HTML-код создания нового поля ввода идентификационной фразы для проверки. Не забудьте про статический текст, предлагающий пользователю ввести идентификационную фразу, которую он видит в изображении CAPTCHA, созданном в результате вызова сценария `capthca.php` и выведенном в форме с помощью тега `<img>`.

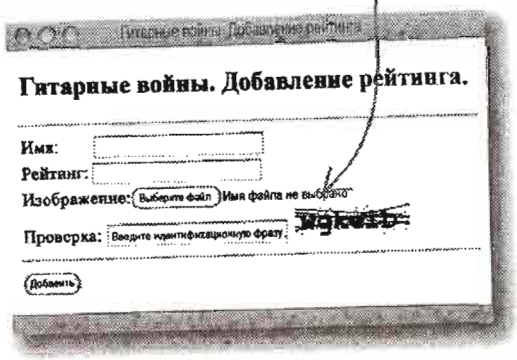
Тег <label> используется для того, чтобы создать ярлык для нового поля ввода «Проверка».

В это текстовое поле пользователь должен ввести идентификационную фразу, которую он видит в изображении CAPTCHA.

```
<label for="verify">Проверка:</label>
<input type="text" id="verify" name="verify" value="Введите идентификационную фразу." />
<img src="captcha.php" alt=" Проверка идентификационной фразы " />
```

В качестве значения атрибута src тега <img> используется имя PHP-сценария, который динамически генерирует изображение CAPTCHA. Это работает потому, что сценарий captcha.php передает изображение непосредственно браузеру, используя функцию imagepng() и HTTP-заголовок.

Изображение CAPTCHA выводится в форме справа от поля ввода идентификационной фразы.



Выполнено! Остался один этап.

~~**3 Вывод изображения CAPTCHA в форме «Добавление рейтинга» приложения «Гитарные войны» с предложением пользователю ввести в поле формы идентификационную фразу.**~~

Включение CAPTCHA в сценарий «Добавление рейтинга»

На клиентской стороне этого диалога сценарий `addscore.php` предоставляет новое поле ввода идентификационной фразы вместе с изображением CAPTCHA. Наиболее значительное изменение сценари связано с появлением управляющей конструкции `if`, в которой производится проверка соответствия идентификационной фразы, выведенной пользователю в изображении CAPTCHA, той идентификационной фразе, которую он ввел в текстовое поле «Проверка».

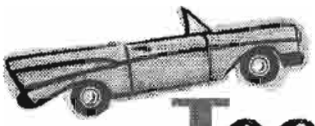
4 Проверка правильности идентификационной фразы, введенной пользователем.

Мы все сделали!

```
<?php
 session_start();
?>

<html>
<head>
 <title> Гитарные войны. Добавление рейтинга </title>
 <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
 <h2> Гитарные войны. Добавление рейтинга </h2>

<?php
 require_once('appvars.php');
 require_once('connectvars.php');

 if (isset($_POST['submit'])) {
  // Соединение с базой данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  // Извлечение данных из суперглобального массива $_POST
  $name = mysqli_real_escape_string($dbc, trim($_POST['name']));
  $score = mysqli_real_escape_string($dbc, trim($_POST['score']));
  $screenshot = mysqli_real_escape_string($dbc, trim($_FILES['screenshot']['name']));
  $screenshot_type = $_FILES['screenshot']['type'];
  $screenshot_size = $_FILES['screenshot']['size'];

  // Проверка соответствия идентификационной фразы, введенной пользователем,
  // идентификационной фразе, выведенной в изображении CAPTCHA
  $user_pass_phrase = sha1($_POST['verify']);
4 if ($_SESSION['pass_phrase'] == $user_pass_phrase) {
   ...
  else {
   echo '<p class="error">Введите идентификационную фразу.</p>';
  }
 }
?>

 <hr />
 <form enctype="multipart/form-data" method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
  <input type="hidden" name="MAX_FILE_SIZE" value="<?php echo GW_MAXFILESIZE; ?>" />
  <label for="name">Имя: </label>
  <input type="text" id="name" name="name" value="<?php if (!empty($name)) echo $name; ?>" /><br />
  <label for="score">Рейтинг: </label>
  <input type="text" id="score" name="score" value="<?php if (!empty($score)) echo $score; ?>" /><br />
  <label for="screenshot">Изображение: </label>
  <input type="file" id="screenshot" name="screenshot" /><br />
  <label for="verify">Проверка: </label>
  <input type="text" id="verify" name="verify" value="Введите идентификационную фразу." />
  <img src="captcha.php" alt="Проверка идентификационной фразы" />
  <hr />
  <input type="submit" value="Добавить" name="submit" />
 </form>
</body>
</html>
```

Здесь зашифрованная идентификационная фраза извлекается из переменной сессии и производится ее сравнение с идентификационной фразой, которую пользователь ввел в поле формы.

Здесь сценарий CAPTCHA подключается к сценарию «Добавление рейтинга» во время выполнения этапа 3. В результате на странице выводится изображение CAPTCHA.

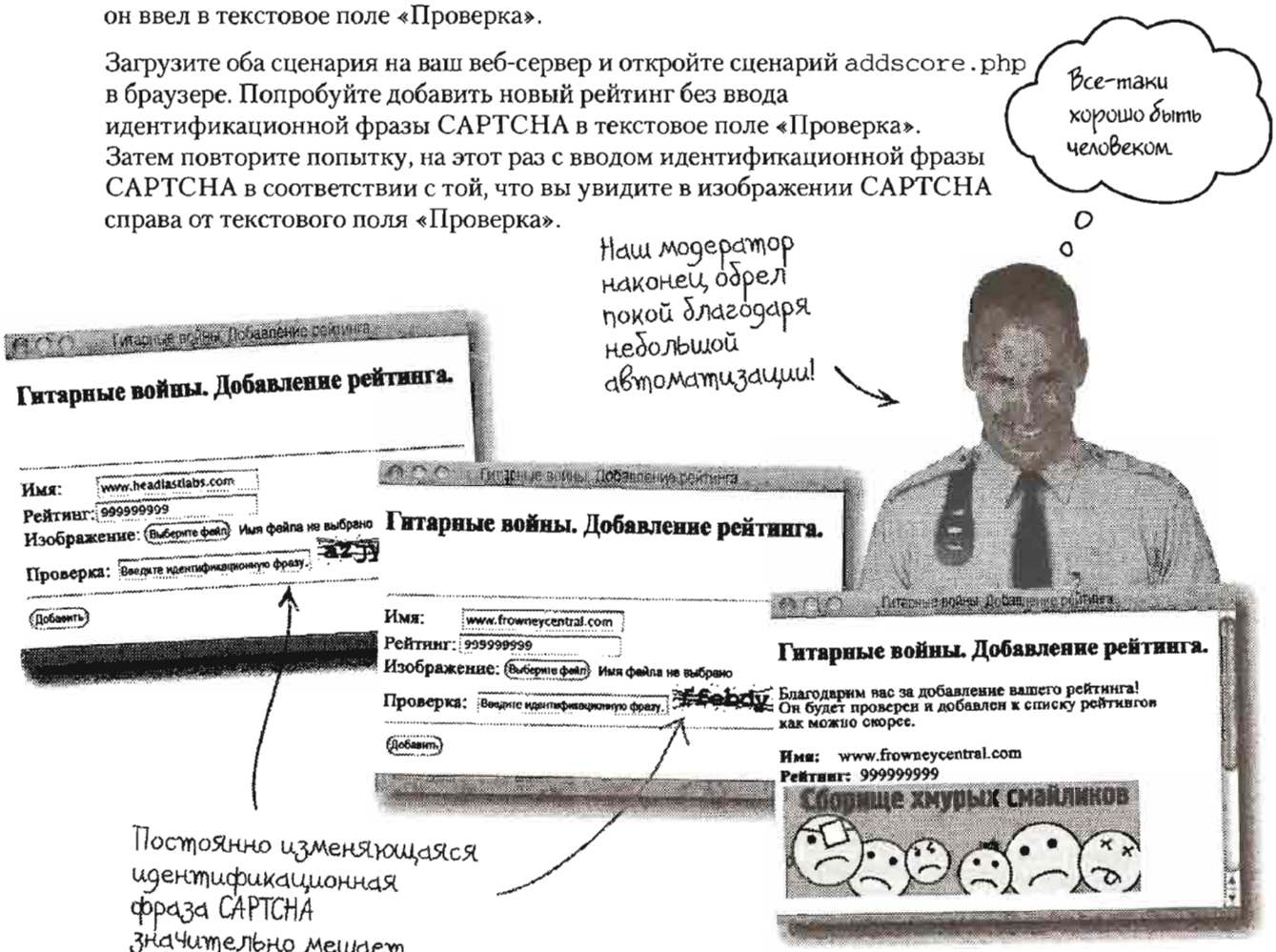


Тест-драйв

Внесите в сценарий «Добавление рейтинга» изменения, обеспечивающие поддержку CAPTCHA.

Внесите в сценарий `addscore.php` изменения, обеспечивающие добавление нового текстового поля «Проверка», а также использование сценария `captcha.php` для вывода в окне изображения CAPTCHA. Добавьте также код, обеспечивающий проверку соответствия идентификационной фразы, выведенной пользователю в изображении CAPTCHA, той фразе, которую он ввел в текстовое поле «Проверка».

Загрузите оба сценария на ваш веб-сервер и откройте сценарий `addscore.php` в браузере. Попробуйте добавить новый рейтинг без ввода идентификационной фразы CAPTCHA в текстовое поле «Проверка». Затем повторите попытку, на этот раз с вводом идентификационной фразы CAPTCHA в соответствии с той, что вы увидите в изображении CAPTCHA справа от текстового поля «Проверка».



Не бывает глупых вопросов

В: Могу ли я, используя функции графической библиотеки GD, создавать изображения в форматах, отличных от PNG?

О: Да. Функции `imagegif()` и `imagejpeg()` действуют аналогично функции `imagepng()`, но создают изображения в форматах GIF и JPEG соответственно.

В: Могут ли эти функции создавать прозрачные изображения?

О: Да! В графической библиотеке GD имеется функция с именем `imagecolortransparent()`, в которой можно установить цвету атрибут прозрачности. Это обязательно должен быть цвет, который вы уже создали вызовом функции `imagecolorallocate()`. После определения цвета как прозрачного любой элемент, нарисованный с использованием этого цвета, будет выглядеть прозрачным. Для созданий прозрачных изображений вы можете вызывать только функции `imagegif()` и `imagepng()` и не можете — `imagejpeg()`, потому что формат JPEG не поддерживает прозрачности.

В: При передаче браузеру изображения PNG где сохраняется файл изображения .png и с каким именем?

О: Никакого файла .png для сохранения изображения не создается. Вместо этого функция `imagepng()` генерирует бинарный образ изображения в оперативной памяти на сервере и передает его непосредственно браузеру, используя HTTP-заголовок. Раз данные изображения после их создания передаются браузеру напрямую, нет никакой необходимости создавать для них файл.

В: Поэтому я присваиваю атрибуту src тега <img> имя сценария CAPTCHA в качестве значения?

О: Правильно. Ссылка на PHP-сценарий в атрибуте `src` тега `<img>`, как это сделано в сценарии `captcha.php` приложения «Гитарные войны», приводит к тому, что изображение передается браузеру как результат выполнения этого сценария. Это отличается от того, как тег `<img>` используется обычно, когда его атрибут `src` ссылается на имя файла изображения. Раз сценарий передает изображение браузеру непосредственно через HTTP-заголовок, необходимость в файле изображения отпадает. А браузер связывает это изображение, полученное через HTTP-заголовок, с конкретным тегом `<img>`, поскольку его атрибут `src` содержит имя этого сценария.

КЛЮЧЕВЫЕ МОМЕНТЫ

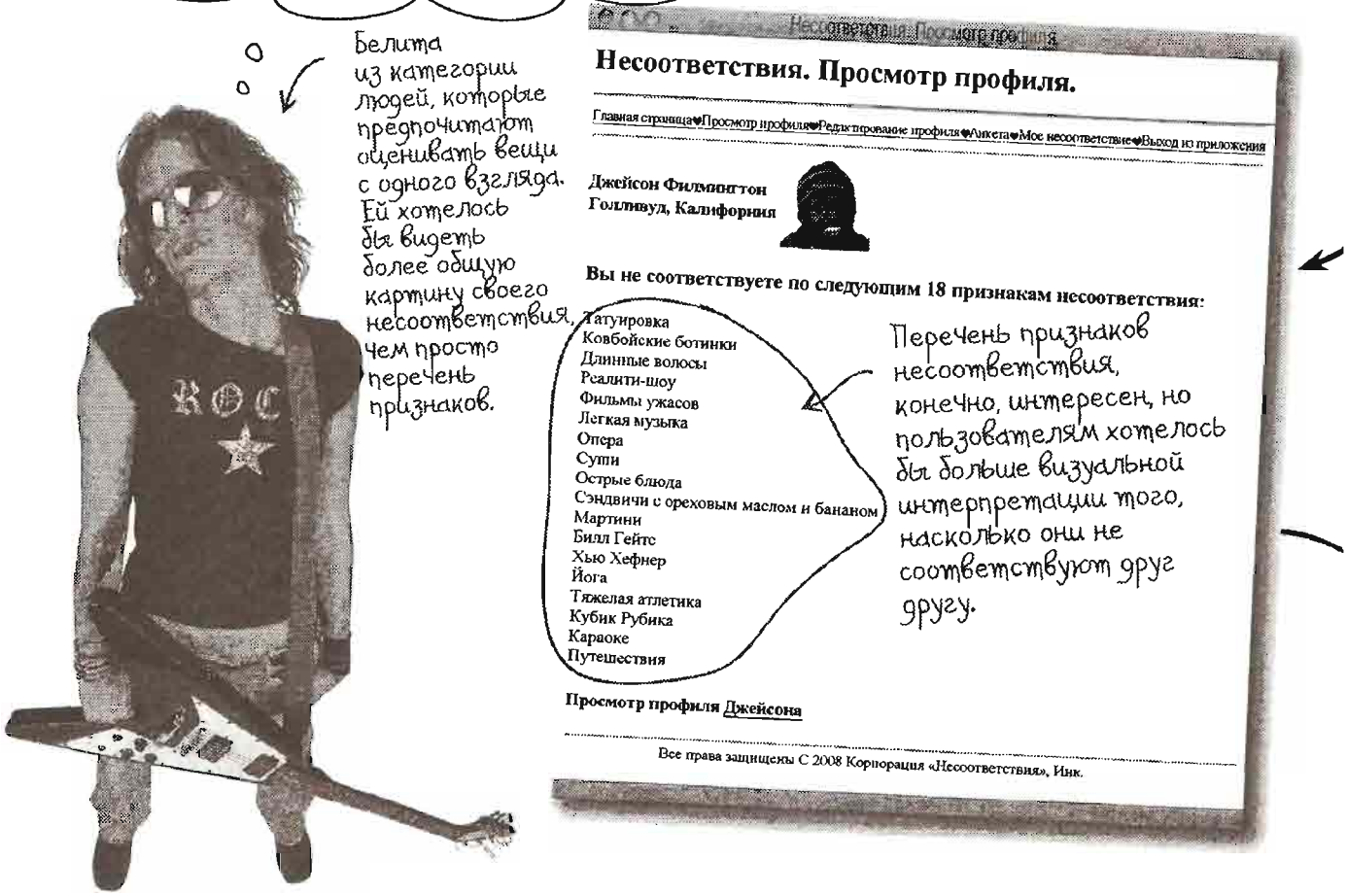
- Все веб-формы рискуют подвергнуться атаке спам-ботов, но все спам-боты рискуют быть остановленными грамотными программистами, использующими эффективные средства защиты, такие, например, как CAPTCHA.
- GD — это стандартная графическая библиотека, которая позволяет вам динамически создавать изображения и рисовать в них различные графические элементы и тексты.
- Функция графической библиотеки GD `createtruecolorimage()` используется для создания пустого изображения, чтобы рисовать в нем различные элементы.
- Для того чтобы передать изображение PNG браузеру или сохранить в файле, вызовите функцию графической библиотеки GD `imagepng()`.
- После того как вы закончили работу с изображением, вызовите функцию `imagedestroy()`, чтобы удалить изображение и освободить все связанные с ним ресурсы.

Пять уровней противопоставления

Так как приложением «Несоответствия» пользуются зарегистрированные члены сообщества, спам-боты не являются для него проблемой. Тем не менее пользователи стремятся получить более подробные оценки несоответствий с применением для начала пяти уровней противопоставления, о которых они недавно узнали. Пользователи приложения «Несоответствия» хотят больше, чем просто перечень признаков несоответствий своего идеального несоответствующего. Они желают видеть в определенном визуальном контексте, как все эти признаки несоответствий распределяются по категориям.

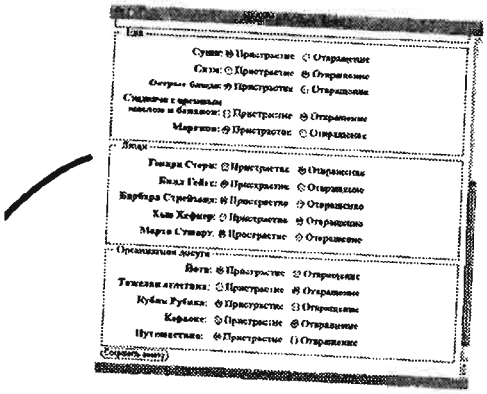
«Пять уровней противопоставления» приложения «Несоответствия» включают измерение несоответствий по категориям.

Я вижу перечень признаков несоответствия, но я не вижу того, как они распределены по разным категориям. Мне говорили о каких-то пяти уровнях противопоставления, но я не имею ни малейшего представления, какое отношение они могут иметь к моим несоответствиям. В чем дело?



Составление гистограммы распределения несоответствий по категориям

Если вы помните, приложение «Несоответствия» включает анкету, в которой пользователь дает свою оценку (пристрастие/отвращение) по различным признакам несоответствия, разбитым на категории. На основании этих оценок разыскивается идеальное несоответствие. Представляя пользователю идеальное несоответствие, сценарий «Мое несоответствие» выводит перечень несоответствий, составленный на основе данных, имеющихся в базе приложения «Несоответствия». Но пользователь теперь хочет больше, чем просто перечень признаков несоответствия... Он хочет видеть визуальное распределение несоответствий по категориям, возможно, в виде гистограммы.



УПРАЖНЕНИЕ

Создайте гистограмму распределения признаков несоответствий по категориям, визуально демонстрирующую пять уровней противопоставления, для Белиты и Джейсона. Объясните значение данных на гистограмме.

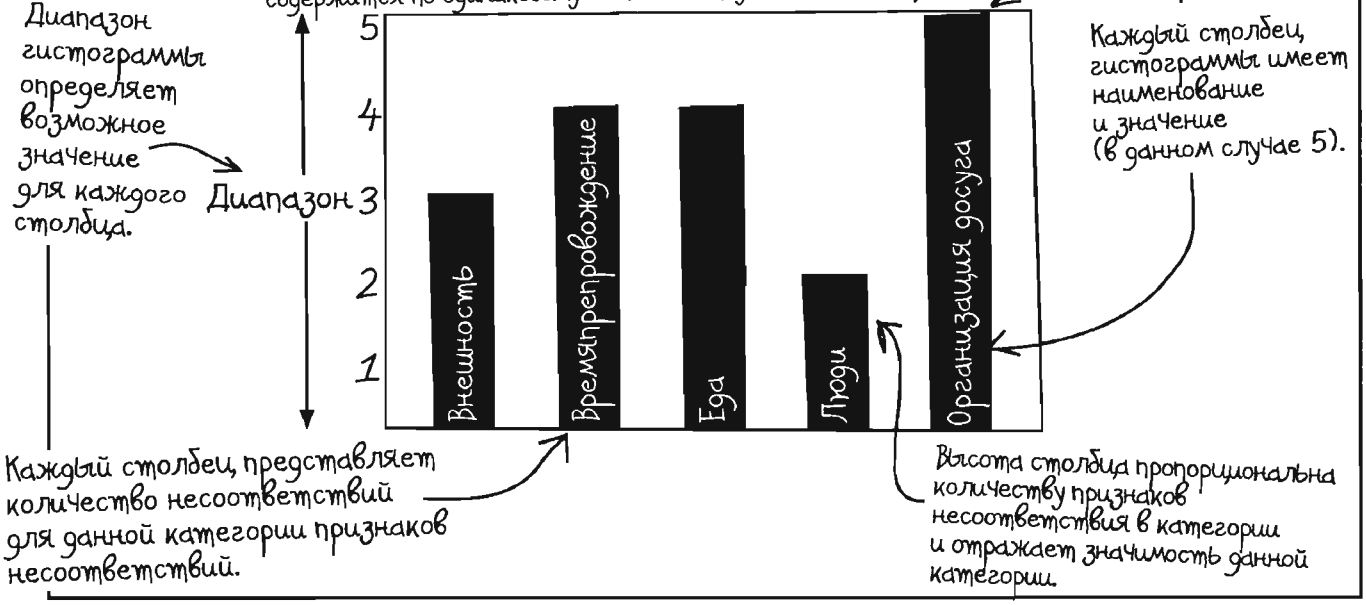
- Татуировка
- Золотые цепочки
- Пирсинг
- Ковбойские ботинки
- Длинные волосы
- Реалити-шоу
- Профессиональная борьба
- Фильмы ужасов
- Легкая музыка
- Опера
- Суши
- Спэм
- Острые блюда
- Сэндвичи с ореховым маслом и бананом
- Мортини
- Говард Стерн
- Билл Гейтс
- Барбара Стрейзонд
- Хью Хефнер
- Марта Стюарт
- Йога
- Тяжелая атлетика
- Кубик Рубика
- Караоке
- Путешествия

Нам необходимо как-то преобразовать этот перечень признаков несоответствия в гистограмму их распределения по категориям.

РЕШЕНИЕ К УПРАЖНЕНИЮ

Создайте гистограмму распределения признаков несоответствия по категориям, визуально демонстрирующую пять уровней противопоставления, для Белиты и Джейсона. Объясните значение данных на гистограмме.

Хотя существует множество способов, которые вы могли бы использовать для визуализации данных приложения «Несоответствия», гистограмма — совсем неплохой выбор, так как в каждой категории признаков несоответствий содержится по одинаковому их количеству.



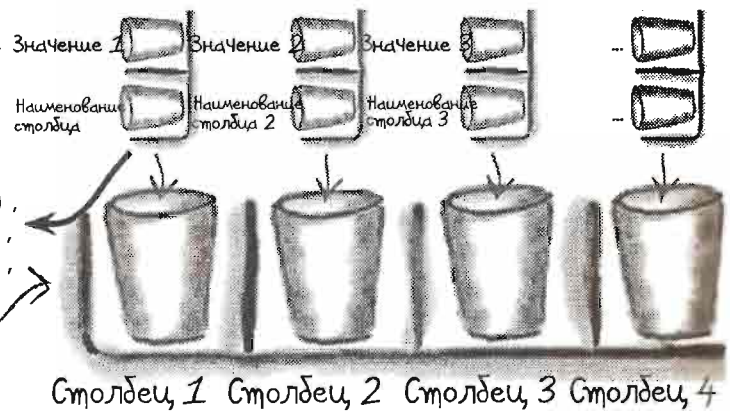
Сохранение данных гистограммы

При ближайшем рассмотрении становится понятным, что данные гистограммы, пожалуй, важнее, чем их графическая интерпретация. Зная, что гистограмма в действительности не что иное, как набор наименований столбцов и их величины, мы можем рассматривать данные для гистограммы как двухмерный массив, в котором в массиве первого уровня сохранены столбцы гистограммы, а в каждом из массивов второго уровня — пары «наименование столбца/его величина» для каждого столбца гистограммы.

В каждом массиве второго уровня сохранены наименования столбцов и их величины.

```
$graph_data = array(
  array("Наименование столбца 1", $value1),
  array("Наименование столбца 2", $value2),
  array("Наименование столбца 3", $value3),
  ...);
```

Каждому элементу массива первого уровня соответствует один столбец гистограммы.



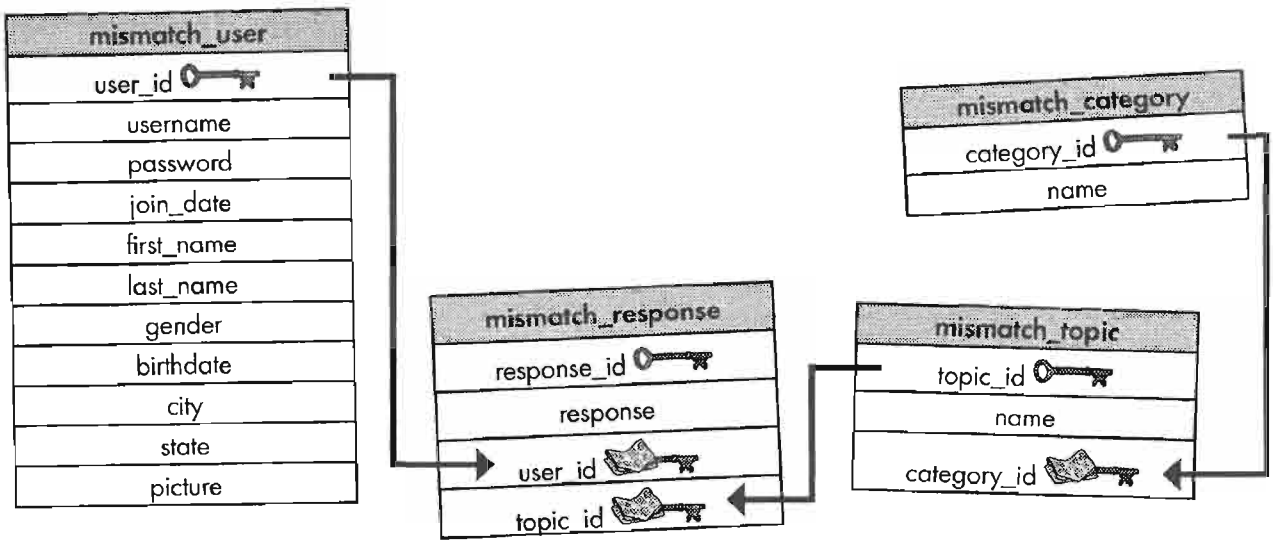
Не бывает глупых вопросов

В: Данные для гистограммы должны поступать из двухмерного массива?

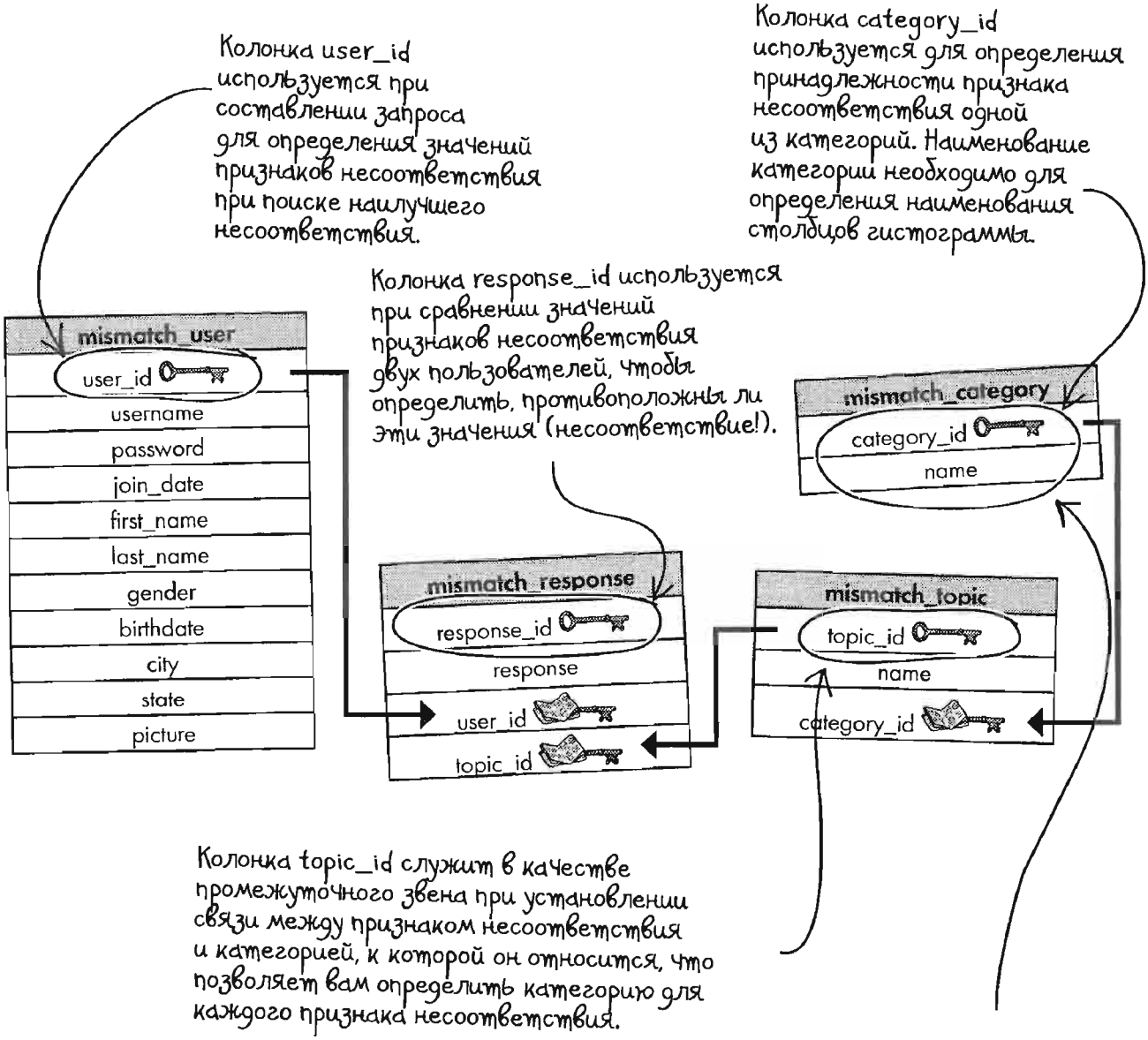
О: Совсем необязательно. Но имейте в виду, что гистограмма обычно имеет дело с парой связанных данных: наименование столбца и его величина. А каждая гистограмма состоит из множества столбцов, поэтому двухмерный массив — это логичный и эффективный способ сохранять данные для построения гистограммы. Есть достаточно эксцентричное высказывание: «Если вы видите только одно решение, значит вы не понимаете проблемы». В данном случае проблем заключается в том, как наилучшим образом сохранить данные, необходимые для построения гистограммы, и одно из конкретных решений, которое работает очень даже неплох заключается в использовании двухмерного массива. Конечно, остается нерешенным вопрос, как создать этот двухмерный массив, содержащий значения признаков несоответствий, сгруппированных по категориям. Первый шаг на этом пути состоит в том, чтобы определить, какая информация в базе данных необходима для решения этой задачи.

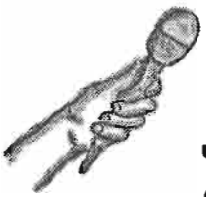
УПРАЖНЕНИЕ

Структура базы данных приложения «Несоответствия» показана ниже. Отметьте все данные, которые необходимы для динамического создания гистограммы «Пять уровней противопоставления», не забыв при этом прокомментировать, как эти данные будут использоваться в процессе создания гистограммы.



Структура базы данных приложения «Несоответствия» показана ниже. Отметьте все данные, которые необходимы для динамического создания гистограммы «Пять уровней противопоставления», не забыв при этом прокомментировать, как эти данные будут использоваться в процессе создания гистограммы.





Гистограмма: близкий взгляд

Интервью этой недели: Читая между строк вместе с мастером создания диаграмм

Корреспондент редакции Head First: Итак, вы тот самый человек, которого вызывают, когда возникает необходимость в визуальном представлении некоторых данных. Да?

Гистограмма: О, да. Я эксперт во всех аспектах визуализации данных, особенно в форме прямоугольных столбцов.

Корреспондент редакции Head First: То есть ваши изобразительные возможности ограничиваются только прямоугольниками?

Гистограмма: Мне кажется, слово «ограничиваются» слишком жесткое для данного случая. Во многих ситуациях действует формула: чем проще, тем лучше. Людям часто удобнее сравнивать данные, когда их величины представлены прямоугольными столбцами разных размеров, возможно, потому что они часто сталкиваются с такими способами представления информации. Посмотрите на измеритель уровня сигнала на экране вашего мобильного телефона. «Вы меня слышите?» Мне это нравится.

Корреспондент редакции Head First: Хорошо. Но я также видел множество достаточно наглядных способов визуализации. Они ассоциируются с приятными мыслями... например, с яблочным пирогом. Вы понимаете, о чем я говорю?

Гистограмма: Я понимаю, к чему вы клоните, и я вполне осведомлена о секторной диаграмме. Послушайте, есть два разных способа рассматривать одни и те же вещи. Секторная диаграмма рассматривает мир как состоящий из кривых линий, я же смотрю на него немного прямее, вот и все.

Корреспондент редакции Head First: Но разве люди по своей природе не тяготеют больше к круглому пирогу, чем к набору каких-то прямоугольных столбцов?

Гистограмма: Нет. По крайней мере, те из них, которые не испытывают чувство голода. Видите ли, секторная диаграмма действительно очень удобна в случаях, когда нужно показать часть целого, когда данные нужно сравнить с чем-то, что соответствует 100 %: 32 команды или 50 штатов. У нас же 50 штатов?

Корреспондент редакции Head First: Да. При условии, что вы рассматриваете Вашингтон, федеральный округ Колумбия как столичный район, а такие местности, как Пуэрто-Рико и Гуам как территории. Но, как я вижу, вы считаете, что секторная диаграмма более удобна для представления различных частей целого, но разве вы не делаете то же самое?

Гистограмма: Это так. Но имейте в виду, что я проявляю большую гибкость. Вы можете добавить ко мне столько столбцов, сколько вам необходимо, при этом не возникает никаких проблем с их изображением. Чем больше элементов вы добавляете к секторной диаграмме, тем меньше размеры отдельных секторов. Начиная с определенного момента они становятся настолько малы, что их трудно отличить друг от друга. Для меня важно только, чтобы высоты всех моих столбцов можно было выполнить в одном масштабе.

Корреспондент редакции Head First: Что вы имеете в виду?

Гистограмма: Видите ли, мне достаточно трудно изображать элементы, значения которых сильно отличаются друг от друга. Если, конечно, вы не возражаете против того, чтобы высоты моих прямоугольных элементов различались в значительной степени. Где мне действительно нет равных, так это в тех случаях, когда нужно показать разницу в значениях, лежащих в определенном диапазоне. Предположим, вы хотите показать колебания цены на бензин в течение года. В этом случае все значения будут находиться внутри определенного диапазона цен, скажем, в несколько долларов.

Корреспондент редакции Head First: Вы в этом уверены?

Гистограмма: Я знаю, что цена бензина — вещь трудно предсказуемая и может колебаться в широком диапазоне, но, по крайней мере, не более широком, чем тот, с которым я имею дело.

Корреспондент редакции Head First: Можно предположить, что вам приходится иметь дело с достаточно необычными вещами, да?

Гистограмма: Вы, возможно, не поверите. Однажды ко мне обратился парень, создавший веб-приложение, измеряющее расстояние, которое пробегает мышь в течение месяца. Все это он постоянно сообщал в своем блоге и обратился ко мне с просьбой составить гистограмму, демонстрирующую статистику этого «путешествия». Достаточно сумасбродно, но людям нравятся такие вещи.

Корреспондент редакции Head First: Итак, можем ли мы сказать, что ваша роль в общей картине, которую веб-приложение рисует в браузере, — это дать пользователям визуальное представление их данных?

Гистограмма: Да, я надеюсь. Каждый раз, когда я попадаю на веб-страницу и показываю данные в более привлекательном для глаз виде, а иначе они выглядели бы унылыми или вообще непонятными, я считаю такой день удачным.

Корреспондент редакции Head First: Рад слышать это. Что ж, я благодарен вам за то, что вы поделились с нами вашими мыслями, и надеюсь на дальнейшие встречи.

Гистограмма: Буду очень рада. И не волнуйтесь, вы еще не раз столкнетесь со мной повсюду.

От одного массива к другому

Когда мы в последний раз имели дело с приложением «Несоответствия», у нас был список признаков несоответствия, по которым у двух пользователей было найдено несоответствие. Точнее, у нас был **массив** признаков несоответствия. Проблема заключается в том, что гистограмма, которую мы пытаемся построить, не использует признаки несоответствия как таковые: она использует **категории**, к которым принадлежат признаки несоответствия. Таким образом, у нас отсутствует один из необходимых уровней. Похоже, что нам придется сделать еще один запрос к базе данных. Нам необходим не только массив признаков несоответствия, но и еще один, параллельный массив категорий, к которым относятся эти признаки несоответствия.

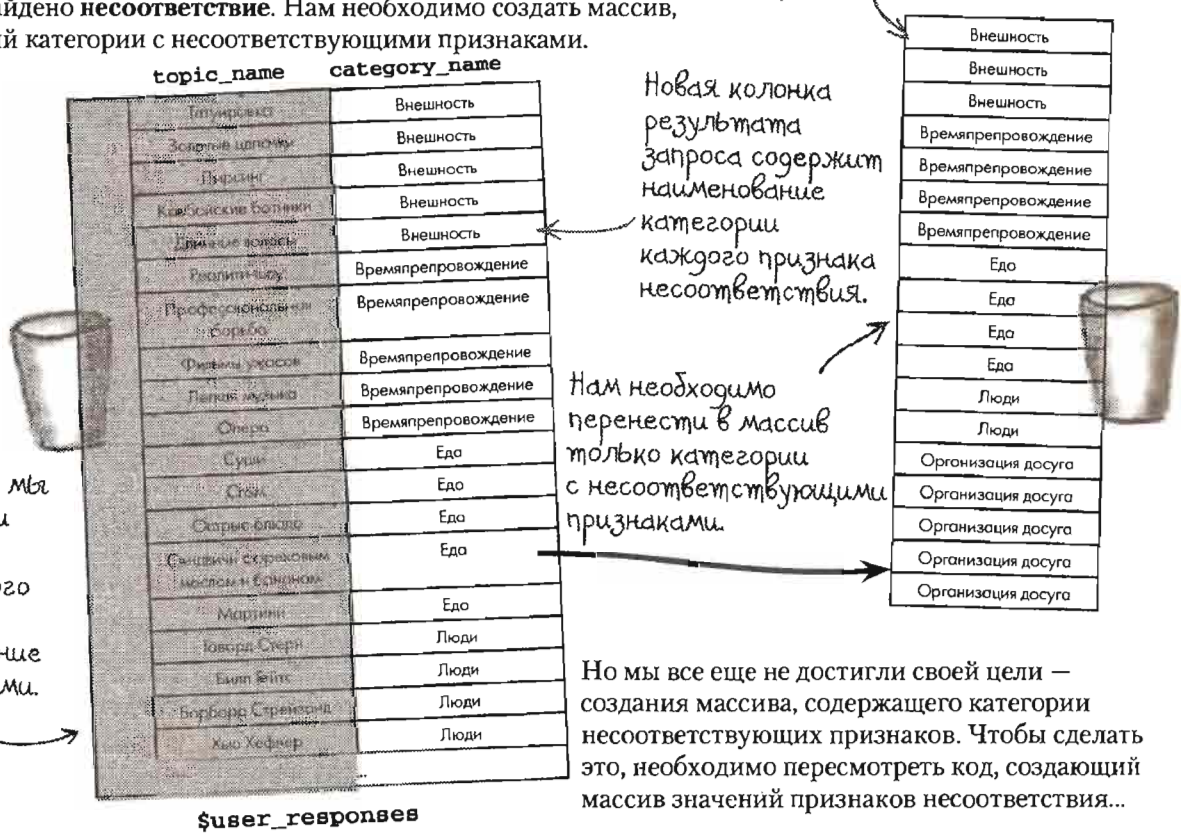
Во избежание конфликта наименований колонок для них используются альтернативные имена.

Выражения JOIN объединяют таблицу, содержащую значения признаков несоответствия, с таблицей, содержащей категории, к которым эти признаки несоответствия относятся, что дает возможность извлечь соответствующие наименования категорий.

```
$query = "SELECT mr.response_id, mr.topic_id, mr.response,
  mt.name AS topic_name, mc.name AS category_name " .
  "FROM mismatch_response AS mr " .
  "INNER JOIN mismatch_topic AS mt USING (topic_id) " .
  "INNER JOIN mismatch_category AS mc USING (category_id) " .
  "WHERE mr.user_id = '" . $_SESSION['user_id'] . "'";
```

Дополнительное объединение в этом запросе приводит к тому, что наименования категорий для каждого признака несоответствия могут непосредственно извлекаться в массив `$user_responses`.
Но не забывайте, что нам необходимы не все категории, а только те, для которых найдено **несоответствие**. Нам необходимо создать массив, содержащий категории с несоответствующими признаками.

Нам необходим массив, содержащий только категории с несоответствующими признаками.



Новая колонка результата запроса содержит наименование категории каждого признака несоответствия.

Нам необходимо перенести в массив только категории с несоответствующими признаками.

В главе 8 мы описывали создание двухмерного массива и заполнение его данными.

$user_responses

Но мы все еще не достигли своей цели — создания массива, содержащего категории несоответствующих признаков. Чтобы сделать это, необходимо пересмотреть код, создающий массив значений признаков несоответствия...

Тест-драйв

Проверьте работу нового запроса для извлечения признаков несоответствия и категорий, к которым они относятся.

Используя инструментальную программу MySQL, выполните следующий запрос SELECT, чтобы извлечь признаки несоответствия и категории, к которым они относятся, для определенного пользователя. Не забудьте указать идентификатор пользователя, который не только существует в базе данных, но и заполнил анкету приложения «Несоответствия»:

Это должен быть идентификатор пользователя, который не только существует в базе данных, но и заполнил анкету приложения «Несоответствия».

```
SELECT mr.response_id, mr.topic_id, mr.response,
  mt.name AS topic_name, mc.name AS category_name
  FROM mismatch_response AS mr
  INNER JOIN mismatch_topic AS mt USING (topic_id)
  INNER JOIN mismatch_category AS mc USING (category_id)
  WHERE mr.user_id = 3;
```

```
Файл Правка Окно Помощь Противоположности
mysql> SELECT mr.response_id, mr.topic_id, mr.response,
  mt.name AS topic_name, mc.name AS category_name
  FROM mismatch_response AS mr
  INNER JOIN mismatch_topic AS mt USING (topic_id)
  INNER JOIN mismatch_category AS mc USING (category_id)
  WHERE mr.user_id = 3;

+-------------+----------+----------+-------------------------+---------------+
| response_id | topic_id | response | topic_name              | category_name |
+-------------+----------+----------+-------------------------+---------------+
|          26 |        1 |        1 | Татуировка              | Внешность     |
|          27 |        2 |        2 | Золотые цепочки         | Внешность     |
|          28 |        3 |        1 | Пирсинг                 | Внешность     |
|          29 |        4 |        2 | Ковбойские ботинки      | Внешность     |
|          30 |        5 |        1 | Длинные волосы          | Внешность     |
|          31 |        6 |        2 | Реалити-шоу             | Времяпрепровождение |
|          32 |        7 |        1 | Профессиональная борьба | Времяпрепровождение |
|          33 |        8 |        1 | Фильмы ужасов           | Времяпрепровождение |
|          34 |        9 |        2 | Легкая музыка           | Времяпрепровождение |
```

Обратите внимание на то, что данные результата запроса согласуются с данными массива $user_responses, показанными на предыдущей странице, а это как раз то, чего мы добиваемся.

Очень полезно проверять запрос в инструментальной программе MySQL, прежде чем вставлять его в PHP-код.

Создание массива для признаков несоответствия

Теперь у нас имеется объединенный запрос, в результате выполнения которого извлекаются наименования и значения признаков несоответствия, а также категории, к которым эти признаки несоответствия относятся. Извлеченные данные заносятся в массив `$user_responses`. Вы, конечно, помните, что в результате выполнения другого аналогичного запроса извлекаются данные, имеющиеся в базе для каждого пользователя для поиска несоответствий. Итак, массив `$user_responses` содержит данные для пользователя, вошедшего в приложение «Несоответствия», в то время как массив `$mismatch_responses` содержит данные для остальных пользователей — кандидатов на идеальное несоответствие.

Мы уже использовали оба этих массива для сохранения значений несоответствий и создания массива, содержащего перечень несоответствий. Сейчас мы сможем добавить еще несколько строк кода для того, чтобы создать массив категорий несоответствующих признаков. **Этот массив будет содержать категории каждого из несоответствующих признаков для двух пользователей**.

Это тот же код, который мы использовали в предыдущей версии, за исключением того, что на этот раз в дополнение к массиву признаков несоответствий мы также создаем массив категорий, к которым эти признаки принадлежат.

```
$categories = array();
for ($i = 0; $i < count($user_responses); $i++) {
  if ($user_responses[$i]['response'] + $mismatch_responses[$i]['response'] == 3) {
    $score += 1;
    array_push($topics, $user_responses[$i]['topic_name']);
    array_push($categories, $user_responses[$i]['category_name']);
  }
}
```

В результате выполнения этого кода создается массив, содержащий категории несоответствующих признаков.

После того как все будет сделано, массив `$categories` станет содержать по одной категории на каждое несоответствие.

Каждый элемент массива категорий несоответствующих признаков создается как результат каждого найденного несоответствия.

Не бывает глупых вопросов

В: Я запутался. Чем отличается результат, возвращаемый после выполнения MySQL-запроса, от PHP-массива?

О: Одно существенное отличие — это доступ. В результате запроса вы можете получить доступ к одной записи за раз, в то время как массив может содержать несколько записей благодаря своей многомерности. Размещение данных результата запроса в двухмерном массиве дает нам возможность свободно и эффективно перемещаться между данными без необходимости делать повторные запросы. Это не будет работать с большими объемами данных, так как вынудит вас создавать огромные массивы, но в случае приложения «Несоответствия» это вполне приемлемый вариант, так как размеры массивов не превышают общего количества признаков несоответствия.

В: Разве нам не нужно подсчитать, сколько раз повторяется одна и та же категория несоответствующего признака, для построения гистограммы?

О: Нужно. Одного массива категорий несоответствующих признаков недостаточно. Вспомните: основная мысль создания гистограммы заключалась в том, что каждый прямоугольный ее элемент представляет категорию несоответствующих признаков, в то время как его высота представляет количество этих признаков. Поэтому нам необходимо подсчитать количество несоответствующих признаков для каждой категории. Но, наверно, стоит вернуться немного назад, к разработке плана наступления...

Разработка плана операции по созданию гистограммы

Имея массив категорий несоответствующих признаков и множество идей по поводу того, как использовать эти данные для создания гистограммы в приложении «Несоответствия», мы все еще не составили план действий. Как выясняется, гистограмма может быть динамически сгенерирована в три этапа, и один из них мы уже прошли.

На этом этапе мы получаем общий перечень категорий несоответствующих признаков.

1. ~~**Выполнение запроса к базе данных приложения «Несоответствия» для извлечения категорий несоответствующих признаков.**~~

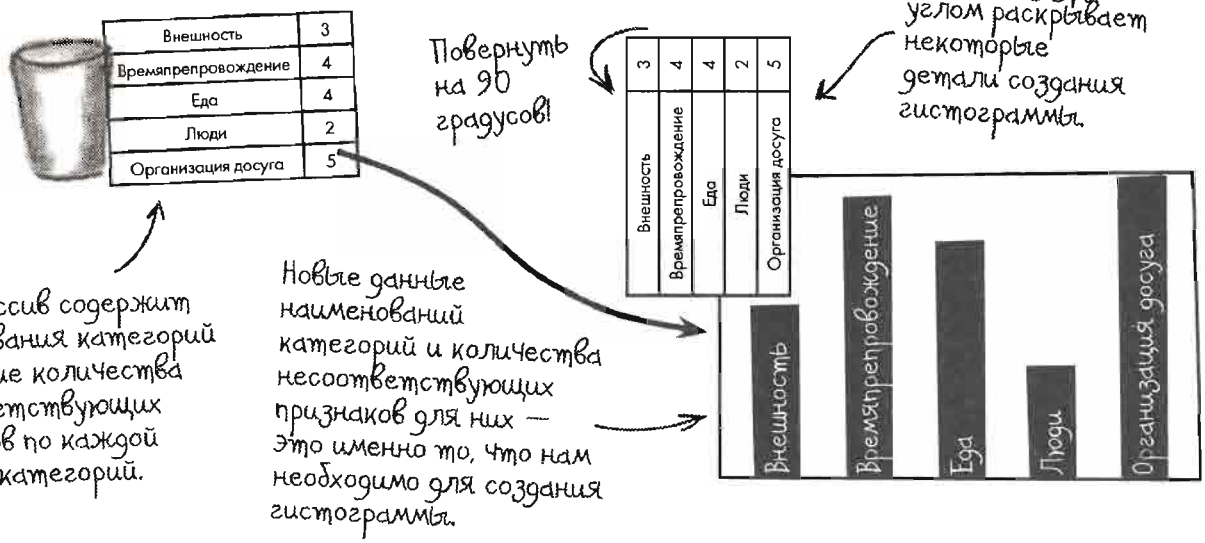
На основании общего перечня категорий несоответствующих признаков необходимо создать перечень категорий вместе с количеством несоответствий для каждой из них.

2. **Подсчет количества несоответствий для каждой категории.**

Имея перечень категорий вместе с количеством несоответствий для каждой из них, мы можем перейти к самому интересному этапу — созданию гистограммы с использованием функций графической библиотеки GD.

3. **Создание гистограммы с использованием данных по наименованиям категорий и количеству несоответствий для каждой из них.**

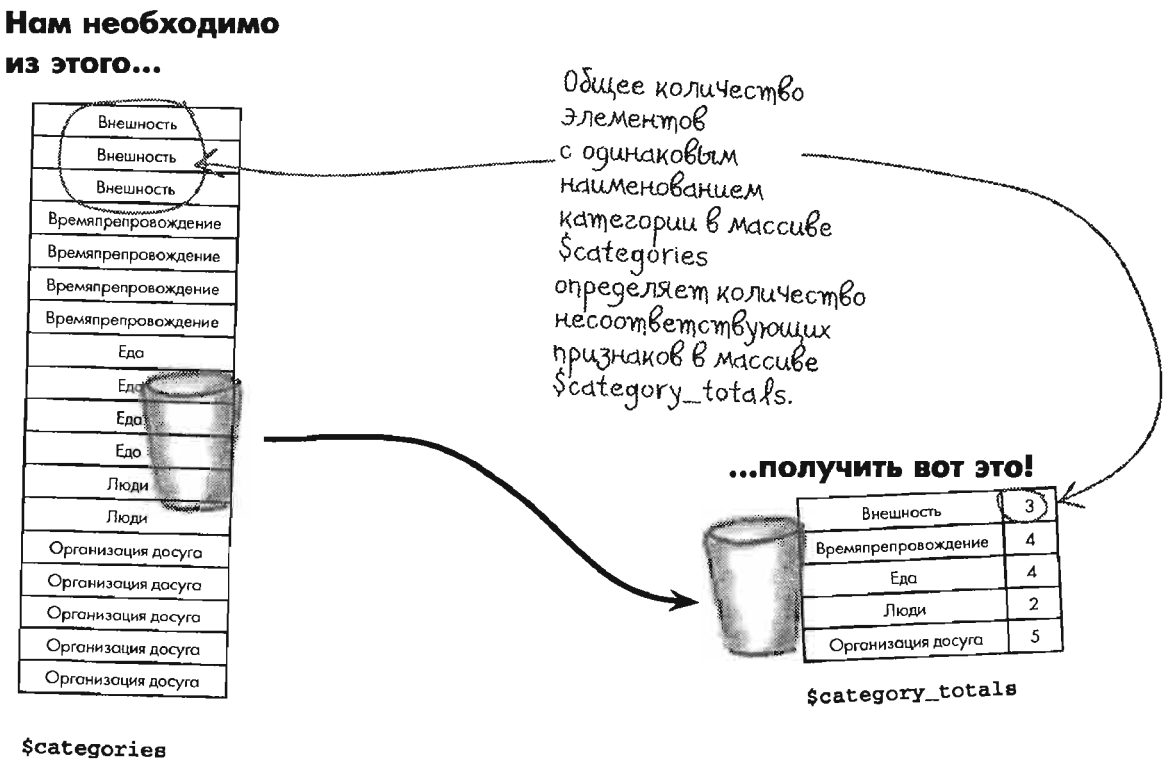
Для того чтобы выполнить задание второго этапа, мы должны, используя данные массива, содержащего общий перечень категорий несоответствующих признаков, создать массив, содержащий наименования категорий вместе с количеством несоответствий для каждой из них. Если вы не забыли, именно такие данные нужны нам для построения гистограммы, в которой наименования категорий будут выступать в роли наименований столбцов, а высота столбцов будет пропорциональна количеству несоответствующих признаков для каждого из этих столбцов. Нам необходимо создать двухмерный массив для объединения в единую структуру наименований категорий и количества несоответствующих признаков по каждой из этих категорий.



Раз массив, содержащий наименования категорий и количество несоответствующих признаков по каждой из этих категорий, создан, мы можем переходить к третьему этапу и, используя функции графической библиотеки GD, запускать процесс создания гистограммы.

Перемалывание категорий

Наша задача теперь заключается в том, чтобы, взяв массив с наименованиями категорий и количеством несоответствующих признаков по каждой из них, создать двухмерный массив наименований столбцов гистограммы и их высоты. Наименования категорий и количество несоответствующих признаков по каждой из этих категорий содержатся в массиве `$categories`. Мы должны создать новый массив, который назовем `$category_totals` и который будет содержать по одному элементу для каждой категории вместе с количеством несоответствующих признаков для каждой из этих категорий.



Как бы вы создали массив `$category_totals`, содержащий наименования категорий и количество несоответствующих признаков для каждой из них, исходя из данных массива `$categories`?

Проведение математической обработки категорий

Создание двухмерного массива с наименованиями столбцов гистограммы и указанием их высоты из одномерного массива, содержащего общий перечень категорий несоответствующих признаков, является задачей не такой уж и простой, как может показаться на первый взгляд. Исходя из этого, полезно составить решение задачи, используя **псевдокод**, прежде чем приступать к разработке реального PHP-кода. Псевдокод освобождает вас от соблюдения строгих синтаксических правил реального кода, позволяя сосредоточиться на разработке алгоритма решения проблемы.

Создание нового двухмерного массива для сохранения наименований категорий вместе с количеством несоответствующих признаков для каждой из этих категорий. Присвоение первому элементу массива значения наименования первой категории с найденным несоответствием, а количеству несоответствующих признаков — нуля.

Прохождение в цикле через все элементы массива, содержащего общий перечень категорі несоответствующих признаков...

Проверка условия: отличается ли наименование категории последнего элемента двухмерного массива от наименования категории текущего элемента одномерного массива?

- **Да! Это новая категория, поэтому добавить в двухмерный массив новый элемент с этим наименованием категории и инициализировать нулем количество несоответствующих признаков.**
- **Нет. Это еще один несоответствующий признак в текущей категории, поэт увеличить на единицу количество несоответствующих признаков последне элемента в двухмерном массиве.**

В результате выполнения такого псевдокода будет создан двухмерный массив, в котором каждый элемент представлен как массив второго уровня, содержащий наименование категории и количество несоответствующих признаков в ней.

Разработайте на базе приведенного псевдокода реальный PHP-код, в результате выполнения которого будет создан двухмерный массив с именем `$category_totals`, содержащий наименования категорий вместе с количеством несоответствующих признаков для каждой из этих категорий.

```
$category_totals = array(array($mismatch_categories[0], 0));

foreach ($mismatch_categories as $category) {

.....

.....

.....

.....

.....

.....

}
```

Разработайте на базе приведенного псевдокода реальный PHP-код, в результате выполнения которого будет создан двухмерный массив с именем `$category_totals`, содержащий наименования категорий вместе с количеством несоответствующих признаков для каждой из этих категорий.

Счет элементов массивов начинается с нуля, поэтому последний элемент массива всегда имеет индекс count() минус единица.

```
$category_totals = array(array($mismatch_categories[0], 0));

foreach ($mismatch_categories as $category) {
  if ($category_totals[count($category_totals) - 1][0] != $category) {
    array_push($category_totals, array($category, 1));
  }
  else {
    $category_totals[count($category_totals) - 1][1]++;
  }
}
```

Это новая категория, поэтому она добавляется в двухмерный массив как новый массив второго уровня с наименованием категории и единицей в качестве начального значения количества несоответствий.

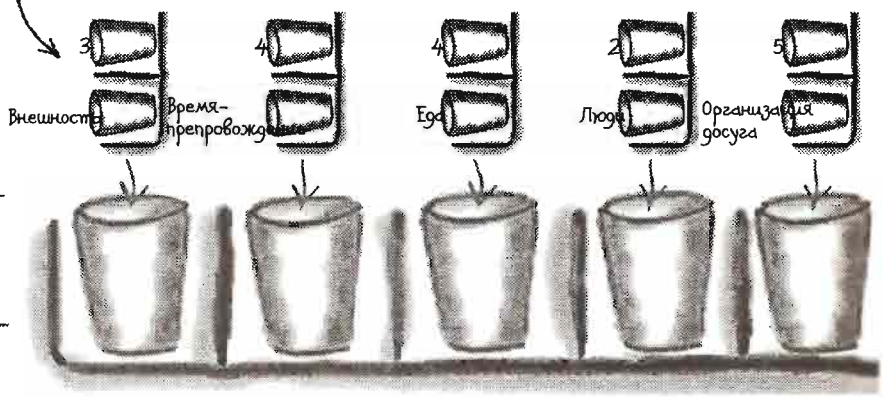
Количество несоответствий для данной категории увеличивается на единицу применением инкрементного оператора (++).

$category_totals

Переменная $category_totals теперь содержит именно те данные, которые необходимы нам, чтобы создать гистограмму распределения несоответствий по категориям.

Внешность	3
Времяпрепровождение	4
Еда	4
Люди	2
Организация досуга	5

Это конечный результат выполнения данного кода.



Теперь мы можем спокойно вычеркнуть этот этап из нашего списка, оставив только последний пункт — создание гистограммы.

~~2 Подсчет количества несоответствий для каждой категории.~~

не бывает глупых вопросов

В: Не возникнет ли проблем при выполнении кода создания двухмерного массива, содержащего наименования категорий вместе с количеством несоответствующих признаков для каждой из этих категорий, если наименования категорий в массиве $mismatch_categories расположены в произвольном порядке?

О: Это серьезная проблема. Этот код будет выполняться, как ожидается, только при условии, что наименования категорий в массиве `$mismatch_categories` расположены строго по порядку. Алгоритм основан на предположении, что каждое изменение наименования категории является ее началом, что справедливо только при том условии, что все категории сгруппированы по наименованиям. К счастью, запрос в сценарии «Анкета», который первоначально извлекает данные для добавления их в массив `$mismatch_categories`, составлен с соблюдением этого условия.

```
SELECT topic_id FROM mismatch_topic ORDER BY category_id, topic_id
```

Этот запрос используется для извлечения признаков несоответствия из базы данных, чтобы затем они были добавлены как признаки несоответствия с пустыми значениями для данного пользователя. Это гарантирует, что признаки несоответствия для каждого пользователя сохраняются в базе данных сгруппированными по категориям. Это и позволяет получать правильные результаты при выполнении кода по созданию двухмерного массива, содержащего наименования категорий вместе с количеством несоответствующих признаков для каждой из этих категорий.

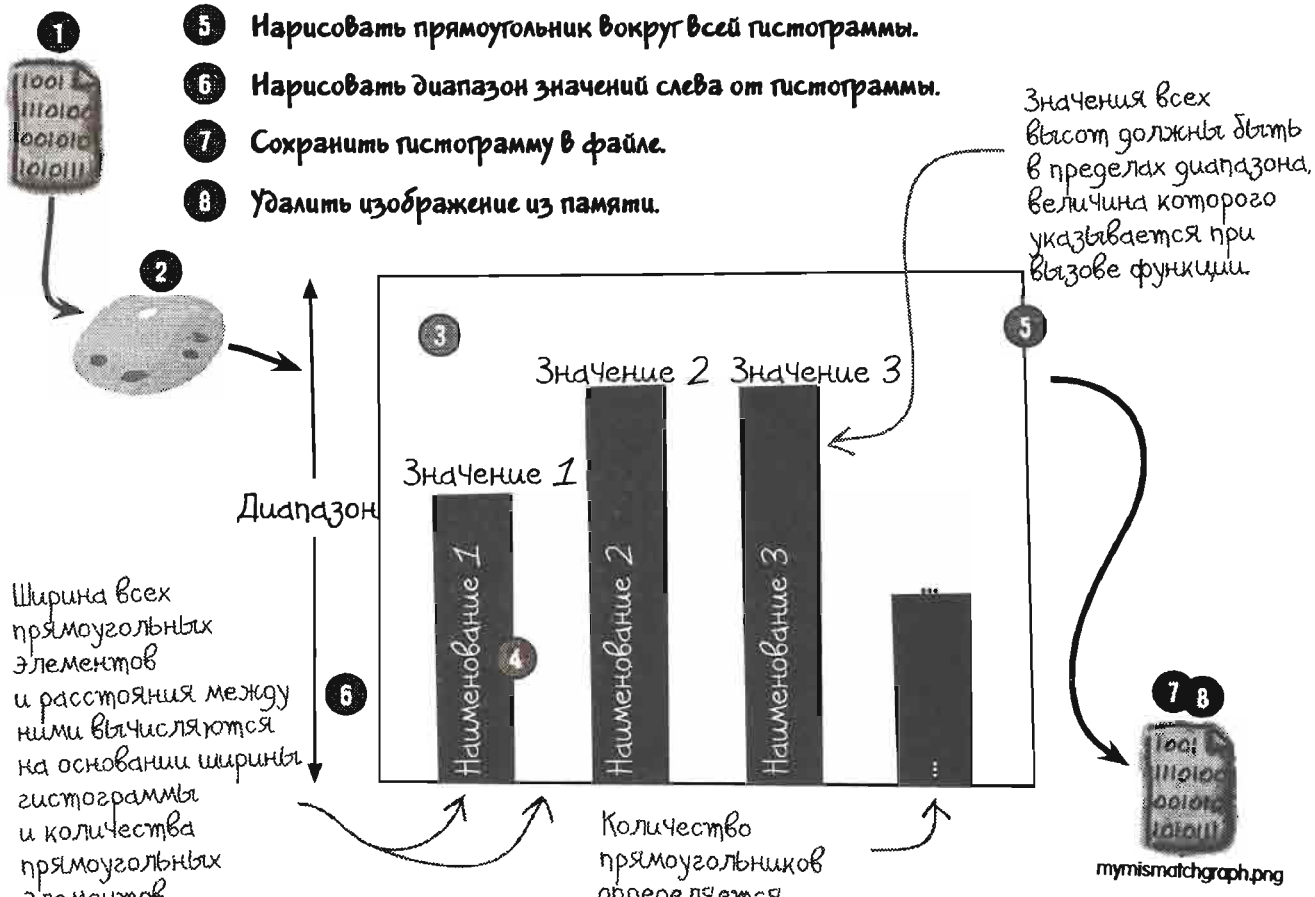
В: Но разве это не рискованно — разрабатывать код, зависимый от порядка сохранения данных в базе?

О: И да, и нет. Не забывайте, что эта база данных полностью управляется кодом сценария, который пишете вы, поэтому порядок сохранения данных в базе может измениться только в случае, если вы измените сценарий. С учетом сказанного в сценарии «Мое несоответствие» запрос должен быть составлен так, чтобы данные в результате его выполнения были сгруппированы по категориям признаков несоответствий.

Основы процесса построения гистограмм

Вы чувствуете, как двухмерный массив, содержащий наименования категорий и количество несоответствующих признаков по каждой из этих категорий, начинает прожигать дыру в вашем кармане. Это значит, что настало время приступать к построению гистограммы на основе его данных. Но вместо того чтобы сосредоточиться на изображении конкретной гистограммы распределения несоответствий по категориям, почему бы не начать с более общего подхода к решению задачи? Если вы разработаете функцию, способную создавать любую гистограмму, вы сможете использовать ее не только для создания гистограммы для приложения «Несоответствия», но и для построения любой другой гистограммы, если такая необходимость возникнет в будущем. Иначе говоря, сделать ее многократно используемой. Эта новая функция в процессе создания гистограммы из двухмерного массива с наименованиями столбцов гистограммы и указанием их высоты должна проходить серию этапов:

1. **Создать изображение.**
2. **Создать цвета, которые вы будете использовать при рисовании графических элементов и текста.**
3. **Заполнить фон цветом.**
4. **Нарисовать столбцы гистограммы и указать их наименования.**
5. **Нарисовать прямоугольник вокруг всей гистограммы.**
6. **Нарисовать диапазон значений слева от гистограммы.**
7. **Сохранить гистограмму в файле.**
8. **Удалить изображение из памяти.**



Магниты PHP

Сценарий «Мое несоответствие» содержит новую функцию `draw_bar_graph()`, которая создает гистограмму, получив в качестве аргументов ее ширину, высоту, диапазон значений, двухмерный массив с данными и имя файла, в котором будет сохранено изображение. Используйте магниты, чтобы добавить отсутствующие вызовы функций графической библиотеки GD.

```
function draw_bar_graph($width, $height, $data, $max_value, $filename) {
1 // Создание пустого изображения

  $img = ..... ($width, $height);

2 // Установка цветов: белого для фона, белого и черного для текста и серого для графики

  $bg_color = .....($img, 255, 255, 255);     // белый цвет

  $text_color = .....($img, 255, 255, 255);    // белый цвет

  $bar_color = .....($img, 0, 0, 0);        // черный цвет

  $border_color = .....($img, 192, 192, 192);  // светло-серый цвет

3 // Заполнение фона

  ..... ($img, 0, 0, $width, $height, $bg_color);

4 // Рисование столбцов гистограммы
  $bar_width = $width / ((count($data) * 2) + 1);
  for ($i = 0; $i < count($data); $i++) {

    .....($img, ($i * $bar_width * 2) + $bar_width, $height,

   ($i * $bar_width * 2) + ($bar_width * 2), $height - (($height / $max_value) * $data[$i][1]), $bar_color);

    ..... ($img, 5, ($i * $bar_width * 2) + ($bar_width), $height - 5, $data[$i][0],

   $text_color);
  }
5 // Рисование: прямоугольник вокруг всей гистограммы

  .....($img, 0, 0, $width - 1, $height - 1, $border_color);

6 // Рисование: диапазон значений слева от гистограммы
  for ($i = 1; $i <= $max_value; $i++) {

    .....($img, 5, 0, $height - ($i * ($height / $max_value)), $i, $bar_color);
7 }

  // Сохранение гистограммы в файле

8 .....($img, $filename, 5);

  .....($img);
}
```

imagecreatetruecolor
imagestringup
imagerectangle
imagedestroy
imagecolorallocate
imagefilledrectangle
imagecolorallocate
imagepng
imagestring
imagefilledrectangle
imagecolorallocate
imagecolorallocate

Магниты PHP. Решение

Сценарий «Мое несоответствие» содержит новую функцию `draw_bar_graph()`, которая создает гистограмму, получив в качестве аргументов ее ширину, высоту, диапазон значений, двухмерный массив с данными и имя файла, в котором будет сохранено изображение. Используйте магниты, чтобы добавить отсутствующие вызовы функций графической библиотеки GD.

```
function draw_bar_graph($width, $height, $data, $max_value, $filename) {
1  // Создание пустого изображения
   $img = imagecreatetruecolor($width, $height);

2  // Установка цветов: белого для фона, белого и черного для текста и серого для графики
   $bg_color = imagecolorallocate($img, 255, 255, 255);    // белый цвет
   $text_color = imagecolorallocate($img, 255, 255, 255);  // белый цвет
   $bar_color = imagecolorallocate($img, 0, 0, 0);         // черный цвет
   $border_color = imagecolorallocate($img, 192, 192, 192); // светло-серый цвет

3  // Заполнение фона
   imagefilledrectangle($img, 0, 0, $width, $height, $bg_color);

4  // Рисование столбцов гистограммы
   $bar_width = $width / ((count($data) * 2) + 1);
   for ($i = 0; $i < count($data); $i++) {
     imagefilledrectangle($img, ($i * $bar_width * 2) + $bar_width, $height,
     ($i * $bar_width * 2) + ($bar_width * 2), $height - (($height / $max_value) * $data[$i][1]), $bar_color);
     imagestringup($img, 5, ($i * $bar_width * 2) + ($bar_width), $height - 5, $data[$i][0],
     $text_color);
   }

5  // Рисование: прямоугольник вокруг всей гистограммы
   imagerectangle($img, 0, 0, $width - 1, $height - 1, $border_color);

6  // Рисование: диапазон значений слева от гистограммы
   for ($i = 1; $i <= $max_value; $i++) {
     imagestring($img, 5, 0, $height - ($i * ($height / $max_value)), $i, $bar_color);
7  }

   // Сохранение гистограммы в файле
8  imagepng($img, $filename, 5);
   imagedestroy($img);
}
```

Вначале создаем новое пустое изображение, в котором будут нарисованы все элементы гистограммы.

Создаем несколько цветов, которые будут использованы для рисования элементов гистограммы.

Расчищаем фон, готовим его для рисования элементов гистограммы.

Рисуем столбцы гистограммы в виде прямоугольников, заполненных черным цветом.

Рисуем наименования столбцов гистограммы в виде вертикальных строк.

Рисуем прямоугольник вокруг всей гистограммы.

Рисуем диапазон значений слева от гистограммы в виде горизонтальной строки.

Сохраняем изображение в файл с заданным именем в PNG-формате и степенью сжатия 5 (средний уровень).

Удаляем изображение из памяти.

Каталог на сервере, в котором будет сохранен файл, должен быть доступен для записи.

Не бывает глупых вопросов

В: Почему функция draw_bar_graph() сохраняет изображение в файле, а не передает его непосредственно браузеру?

О: Потому что эта функция не имеет своего собственного сценария, который мог бы передать изображение браузеру через HTTP-заголовок. Не забывайте, что единственный способ передать динамически созданное изображение непосредственно браузеру — это использование HTTP-заголовка. Но это возможно только в том случае, если единственной целью сценария является создание изображения.

В: Тогда почему бы не поместить функцию draw_bar_graph() в ее собственный сценарий, чтобы он смог передать браузеру изображение гистограммы непосредственно?

О: Хотя это и хорошая мысль — поместить функцию в ее собственный сценарий, чтобы иметь возможность использовать ее многократно, — все же проблема передачи изображения через HTTP-заголовок остается. Она связана с тем, как вы используете код внешнего сценария. Если этот код включается с использованием директив `include`, `include_once`, `require` или `require_once`, он просто помещается на место этой директивы, как будто он был здесь изначально. Это отлично работает, пока не связано с какими-либо манипуляциями браузером. Но отправка HTTP-заголовка влияет на ту информацию, которую сценарий сам отправляет браузеру помимо динамически созданного изображения, что может осложнить дело. Вопрос не стоит так, что вы не можете отправлять заголовок из добавленного внешнего кода: вы уже делали это ранее. Проблема заключается в том, что вы должны быть исключительно внимательными, и в некоторых случаях достаточно небезопасно считать, что заголовок еще не отправлен. Сценарий «Мое несоответствие», например, не может оправить браузеру изображение гистограммы с использованием HTTP-заголовка, потому что его задача — отправить браузеру HTML-код, содержащий результаты анализа несоответствий. Включение внешнего сценария по динамическому созданию изображения гистограммы и отправке его браузеру вызовет конфликт HTTP-заголовков.

В: Хорошо. Тогда могу ли я просто сослаться на код создания гистограммы, как мы ссылались на сценарий captcha.php в приложении «Гитарные войны». Похоже, это не должно вызывать проблем, подобных тем, которые возникают при включении внешнего кода, так ведь?

О: Да, мы действительно ссылались на сценарий `captcha.php` непосредственно через атрибут `src` тега `<img>`, и это работало. Проблема же в данном случае связана с тем, что изображение гистограммы имеет довольно большой объем, и это может вызвать затруднения при передаче его через суперглобальные массивы `$_POST` или `$_GET`.

Создание и вывод гистограммы распределения несоответствий по категориям

Функция `draw_bar_graph()` дает возможность динамически создавать изображение гистограммы при условии, что вы передали ей необходимую информацию. В случае создания такой гистограммы для приложения «Несоответствия» это включает передачу значений ширины и высоты, которые будут приемлемыми для страницы «Мое несоответствие» (480 × 240), двухмерного массива, содержащего наименования категорий и количество несоответствующих признаков по каждой из них, максимального значения диапазона, равного пяти (5 — это максимальное количество наименований несоответствий, принадлежащих одной категории), и имени файла, в котором это изображение должно быть сохранено. В результате вызова этой функции генерируется изображение, готовое для вывода с использованием HTML-тега `<img>`.

Файл изображения, созданный в результате вызова этой функции, имеет имя mymismatchgraph.png и сохранен на сервере, в каталоге с именем, содержащемся в константе MM_UPLOADPATH.

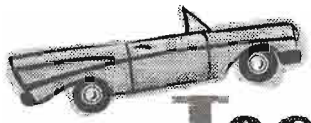
```
echo '<h4>Гистограмма распределения несоответствий по категориям</h4>';

draw_bar_graph(480, 240, $category_totals, 5, MM_UPLOADPATH . 'mymismatchgraph.png');

echo '<img src="' . MM_UPLOADPATH . 'mymismatchgraph.png" alt="Гистограмма распределения
    несоответствий по категориям" /><br />';" /><br />';
```

С помощью многократно используемой функции появилась возможность создать изображение гистограммы и сохранить его в файле, используя данные из базы.

В качестве значения атрибута src тега <img> взято то же самое имя файла изображения и каталога, в котором он сохранен.

Внешность	3
Времяпрепровождение	4
Еда	4
Люди	2
Организация досуга	5

mymismatchgraph.png

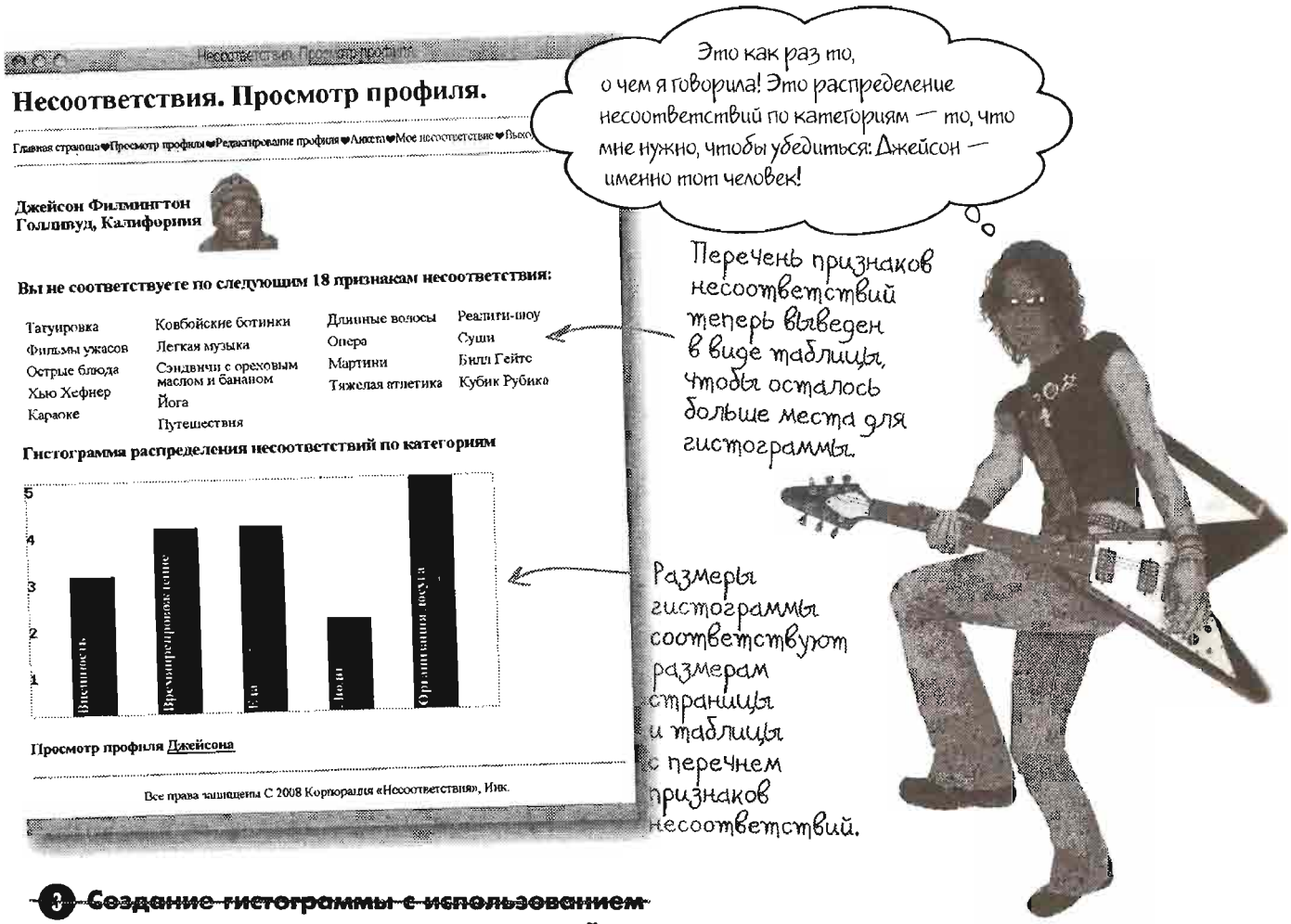


Тест-драйв

Создайте сценарий «Мое несоответствие» и проверьте его работу.

Создайте новый текстовый файл с именем `mymismatch.php` и введите в него код сценария «Мое несоответствие» (или загрузите его с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Добавьте также новую гиперссылку на сценарий «Мое несоответствие» в навигационное меню (сценарий `navmenu.php`).

Загрузите сценарий на ваш веб-сервер и откройте главную страницу приложения «Несоответствия» (`index.php`) в браузере. Войдите в приложение и щелкните кнопкой мыши на гиперссылке «Мое несоответствие» в главном навигационном меню. Поздравляем, это ваше идеальное несоответствие!



3 Создание гистограммы с использованием данных по наименованиям категорий и количеству несоответствий для каждой из них.

Интересно, как это мы можем сохранять гистограммы распределения несоответствий по группам для разных пользователей в одном и том же файле, в то время как для каждого пользователя создается свое уникальное изображение гистограммы?

Это всего лишь вопрос случая

Дело в том, что в каждый определенный момент времени может существовать изображение гистограммы распределения несоответствий по группам только для одного пользователя независимо от их общего количества. Это может повлечь за собой проблему, если два пользователя откроют страницу «Мое несоответствие» в один и тот же момент времени. При этом мы подвергаемся риску создания двух различных изображений с попыткой записи их в один и тот же файл.

Вероятность возникновения такой проблемы в реальной жизни, скорее всего, исключительно мала, но в связи со значительным ростом популярности приложения «Несоответствия», число пользователей которого измеряется тысячами, пренебрегать ею нельзя. Тот факт, что каждый пользователь рассматривает изображение гистограммы распределения несоответствий по категориям как свое собственное, указывает на слабость такого подхода.

Здесь у нас три разных изображения гистограмм распределений несоответствий по категориям, но мы знаем, что только одно изображение может быть сохранено в файле.

mymismatchgraph.png

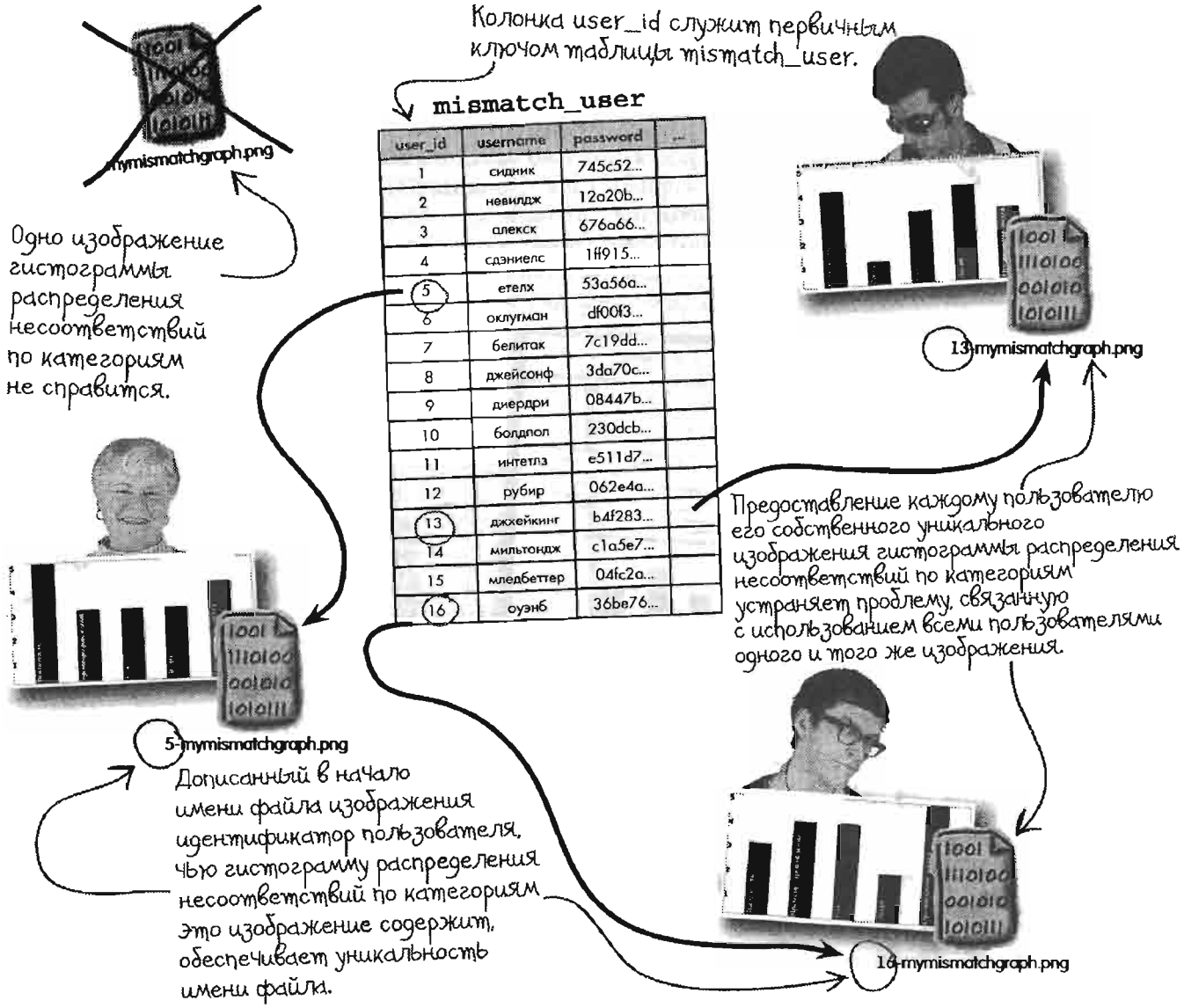
Единственное изображение постоянно генерируется для каждой гистограммы распределения несоответствий по категориям.

СИЛА МЫСЛИ

Как бы вы изменили код приложения «Несоответствия», чтобы пользователи не получали одно и то же изображение гистограммы, отражающей распределение несоответствий по категориям?

Индивидуальные изображения гистограмм распределения несоответствий по категориям для всех

Решение проблемы общего изображения гистограммы, показывающей распределение несоответствий по категориям, заключается в генерировании множества изображений — фактически по одному на каждого пользователя. Но нам необходимо быть уверенными, что это конкретное изображение относится к этому конкретному пользователю и ни к какому другому. Вот здесь-то и выходит на сцену знакомый объект базы данных — первичный ключ! Первичный ключ `user_id` для таблицы `mismatch_user` является уникальным идентификатором пользователя и поэтому предоставляет отличную возможность дать уникальное имя файлу изображения гистограммы для каждого пользователя. Все, что нам для этого необходимо, — это дописать в начало имени файла изображения идентификатор пользователя, чью гистограмму распределения несоответствий по категориям это изображение содержит.



Не бывает глупых вопросов

В: **Имеются ли какие-либо преимущества формата PNG перед форматами GIF или JPEG для генерирования динамических изображений?**

О: Нет, кроме тех преимуществ, которые имеют одни форматы перед другими для статических изображений. Например, форматы GIF и PNG более предпочтительны для векторной графики, в то время как JPEG лучше передает изображения растровой графики, например фотографий. В случае приложения «Несоответствия» мы имеем дело с векторной графикой, поэтому и GIF, и PNG будут работать хорошо. PNG — это более современный формат, поэтому и используется здесь, но и GIF работал бы не хуже. Для вывода изображения в форматах GIF или JPEG вызывайте функции `imagegif()` или `imagejpeg()` соответственно.

В: **Как мне определить наилучший уровень сжатия при выводе изображения PNG в файл?**

О: Заданное значение уровня сжатия для функции `imagepng()` оказывает влияние на качество изображения при выводе его в файл и на размер этого файла. Значение уровня сжатия может быть в пределах от нуля (никакого сжатия) до девяти (максимальное сжатие). Не существует жестких правил по поводу того, какой уровень сжатия лучше и в каких случаях, поэтому вам, возможно, придется поэкспериментировать с разными значениями. В приложении «Несоответствия» при сохранении изображения гистограммы используется уровень 5, который является вполне оправданным компромиссом между качеством изображения и эффективностью (размером файла).

В: **Не будут ли возникать проблемы, связанные с нехваткой дискового пространства для сохранения файлов изображений гистограмм для каждого пользователя?**

О: Не должно. Этот вопрос имеет некоторую связь с предыдущим вопросом о степенях сжатия изображений при сохранении их в файлах. Не похоже, что вы перегрузите сервер слишком объемными файлами или слишком большим их количеством, если только вас не угораздит создавать объемные файлы тысячами. В качестве примера представьте себе, что в приложении «Несоответствия» средний размер файла изображения гистограммы составляет 2 Кбайт. Тогда, даже если в приложении зарегистрируются 50 000 пользователей, для сохранения этих файлов понадобится всего 100 Мбайт дискового пространства. Это не так уж и много по сравнению с общим дисковым пространством, предоставляемым хостинговыми компаниями, зато 50 000 пользователей гарантируют вам достаточные средства для покрытия таких затрат с лихвой.

УПРАЖНЕНИЕ

Ниже приведен код, в результате выполнения которого генерируется изображение гистограммы, показывающей распределение признаков несоответствий по категориям (затем оно выводится на веб-страницу). Перепишите код так, чтобы имена файлов изображений были разными для всех пользователей.

Совет: используйте глобальную переменную `$_SESSION['user_id']` для создания уникальных имен файлов.

```
echo '<h4>Гистограмма распределения несоответствий по категориям</h4>';

draw_bar_graph(480, 240, $category_totals, 5, MM_UPLOADPATH . 'mymismatchgraph.png');

echo '<img src="' . MM_UPLOADPATH . 'mymismatchgraph.png" alt="Гистограмма распределения
несоответствий по категориям" /><br />';
```

Ниже приведен код, в результате выполнения которого генерируется изображение гистограммы, показывающей распределение признаков несоответствий по категориям (затем оно выводится на веб-страницу). Перепишите код так, чтобы имена файлов изображений были разными для всех пользователей.

Совет: используйте глобальную переменную `$_SESSION['user_id']` для создания уникальных имен файлов.

```
echo '<h4>Гистограмма распределения несоответствий по категориям</h4>';

draw_bar_graph(480, 240, $category_totals, 5, MM_UPLOADPATH . 'mymismatchgraph.png');

echo '<img src="' . MM_UPLOADPATH . 'mymismatchgraph.png" alt="Гистограмма распределения
несоответствий по категориям" /><br />';
```

Здесь используем имя стандартного каталога для загружаемых пользовательских файлов, чтобы быть уверенным, что необходимые права имеются.

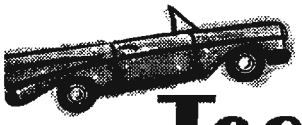
Убедитесь, что вы имеете права на запись в этот каталог на сервере.

Уникальное имя файла имеет общее наименование X-mymismatchgraph.png, где X — это значение идентификатора пользователя.

```
echo '<h4>Гистограмма распределения несоответствий по категориям</h4>';
draw_bar_graph(480, 240, $category_totals, 5,
MM_UPLOADPATH . $_SESSION['user_id'] . '-mymismatchgraph.png');
echo '<img src="' . MM_UPLOADPATH . $_SESSION['user_id'] . '-mymismatchgraph.png" ' .
'alt="Гистограмма распределения несоответствий по категориям" /><br />';
```

Мы используем идентификатор пользователя, значение которого мы уже сохранили в переменной сессии.

То же самое имя файла используется для присвоения значения атрибуту src тега <img> для HTML-кода вывода изображения гистограммы.



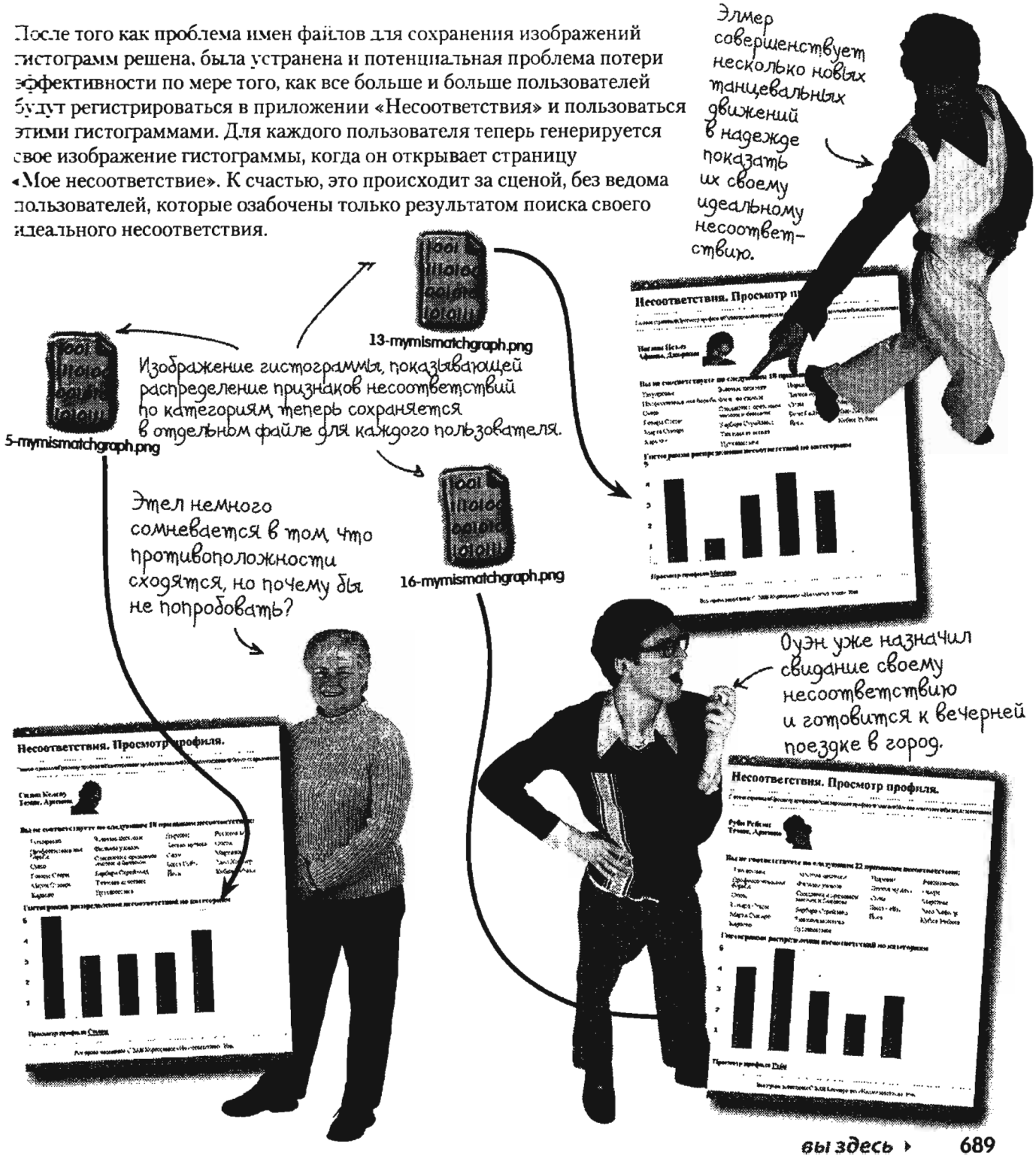
Тест-драйв

Внесите в сценарий «Мое несоответствие» изменения, обеспечивающие создание уникальных имен файлов для изображений гистограмм, которые показывают распределение признаков несоответствий по категориям.

Внесите в сценарий «Мое несоответствие» изменения, обеспечивающие создание уникальных имен файлов для изображений гистограмм, которые показывают распределение признаков несоответствий по категориям. Загрузите сценарий `mymismatch.php` на ваш веб-сервер и откройте в браузере. Внешний вид страницы не изменится, но вы можете просмотреть ее исходный текст, чтобы убедиться, что теперь имена файлов изображений уникальны.

Пользователи приложения «Несоответствия» углубились в изучение гистограмм распределений признаков несоответствий по категориям

После того как проблема имен файлов для сохранения изображений гистограмм решена, была устранена и потенциальная проблема потери эффективности по мере того, как все больше и больше пользователей будут регистрироваться в приложении «Несоответствия» и пользоваться этими гистограммами. Для каждого пользователя теперь генерируется свое изображение гистограммы, когда он открывает страницу «Мое несоответствие». К счастью, это происходит за сценой, без ведома пользователей, которые озабочены только результатом поиска своего идеального несоответствия.



Ваш инструментарий PHP и MySQL

Динамически создаваемые графические объекты открывают новые интересные возможности в создании PHP-сценариев с генерацией изображений на лету. Давайте подведем итоги: что дает возможность делать это?

CAPTCHA

Программа, которая защищает сайт от атак автоматических спам-ботов путем использования определенных тестов. Например, CAPTCHA может включать определение искаженных букв, составляющих идентификационную фразу, определение содержания изображения или анализ уравнения с несложными математическими вычислениями.

Графическая библиотека GD

Набор PHP-функций, которые используются для того, чтобы рисовать различные графические объекты в изображении. Графическая библиотека GD позволяет вам динамически рисовать в изображениях и затем или передавать их непосредственно браузеру, или сохранять в файле на сервере.

imagecreatetruecolor()

Эта функция включена в состав графической библиотеки GD и используется для создания новых цветов при рисовании графических объектов. Изображение первоначально создается в оперативной памяти и остается там до тех пор, пока не будет вызвана специальная функция, например imagepng().

imagestring(), imagestringup(), imagettftext()

Графическая библиотека GD позволяет также писать текст либо с помощью встроенного шрифта, либо с использованием шрифтов True Type на ваш выбор.

imageline(), imagerectangle(), ...

Графическая библиотека GD включает множество функций для построения различных графических примитивов, таких как линии, прямоугольники и даже отдельные пикселы. Каждая функция оперирует с изображением, которое уже создано вызовом функции imagecreatetruecolor().

imagepng()

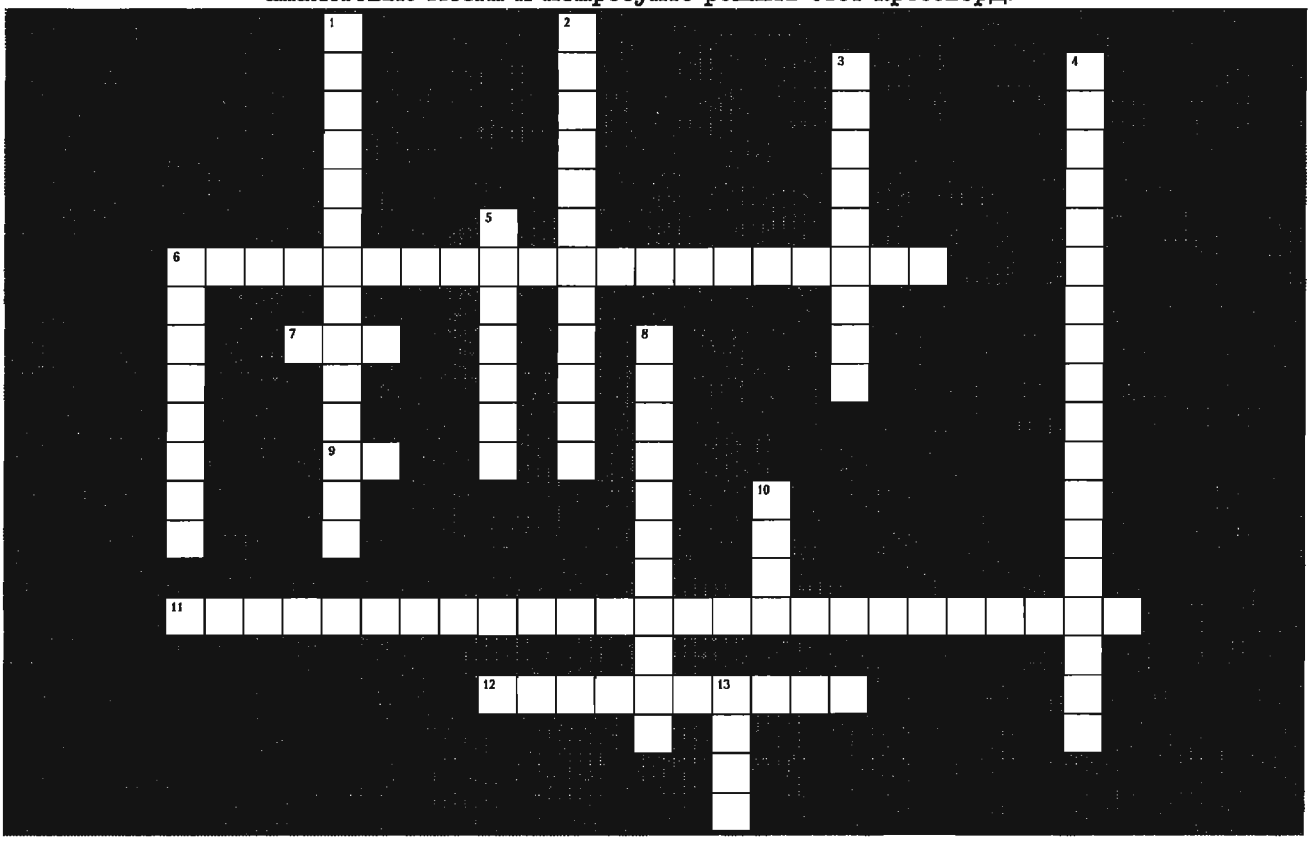
После того как вы закончили создание изображения с использованием графической библиотеки GD, вызов этой функции позволяет вывести изображение, чтобы его можно было показать в приложении. Вы можете вывести изображение непосредственно в браузер или сохранить в файле на сервере.

imagedestroy()

После окончания рисования изображения и вывода его туда, куда вам необходимо, очень важно освободить все ресурсы, связанные с законченным изображением, вызовом этой функции.

Кроссворд PHP и MySQL

Конечно, вы могли бы использовать робота, но что-то их не видно поблизости, так что напрягите свои аналоговые мозги и попробуйте решить этот кроссворд.



По горизонтали

6. Вызовите эту функцию графической библиотеки GD для того, чтобы создать новое изображение.

7. Если вы хотите писать текст с использованием определенного шрифта, вызовите функцию image...text().

9. Сокращенное наименование графической библиотеки PHP.

11. Для создания изображения гистограммы отдельно для каждого пользователя данное значение используется как часть имени файла, в котором сохраняется это изображение.

12. В приложении «Несоответствия» в массиве такого типа сохраняются данные, необходимые для построения изображения гистограммы, показывающей распределение признаков несоответствий по категориям.

По вертикали

1. В результате вызова этой функции с двумя аргументами (со значениями координат двух точек) будет нарисован прямоугольник.

2. Всегда вызывайте эту функцию для удаления изображения из памяти и освобождения всех связанных с ним ресурсов, после того как вы закончили работу с этим изображением.

3. Эта функция графической библиотеки GD рисует прямую линию.

4. Приложение «Несоответствия» использует гистограмму распределения признаков несоответствий по категориям для сравнения пользователей на основании пяти уровней ...

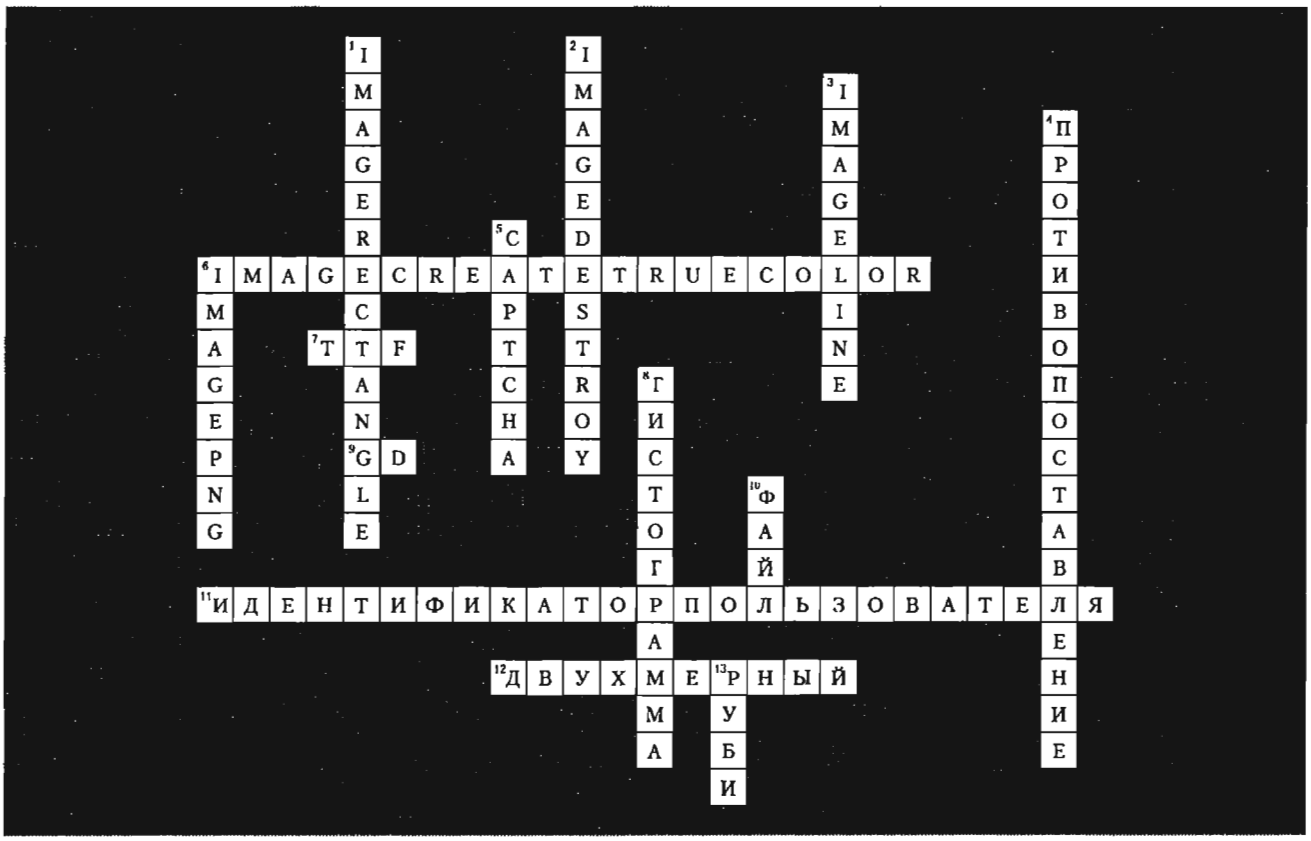
5. Тест, используемый для того, чтобы отличить человека от автоматизированного спам-бота.

8. Визуальное представление распределения признаков несоответствий по категориям.

10. Когда готовое изображение выводится средствами PHP, оно может быть либо передано непосредственно браузеру, либо сохранено в ...

13. Идеальное несоответствие Оуэна.

Кроссворд PHP и MySQL. Решение



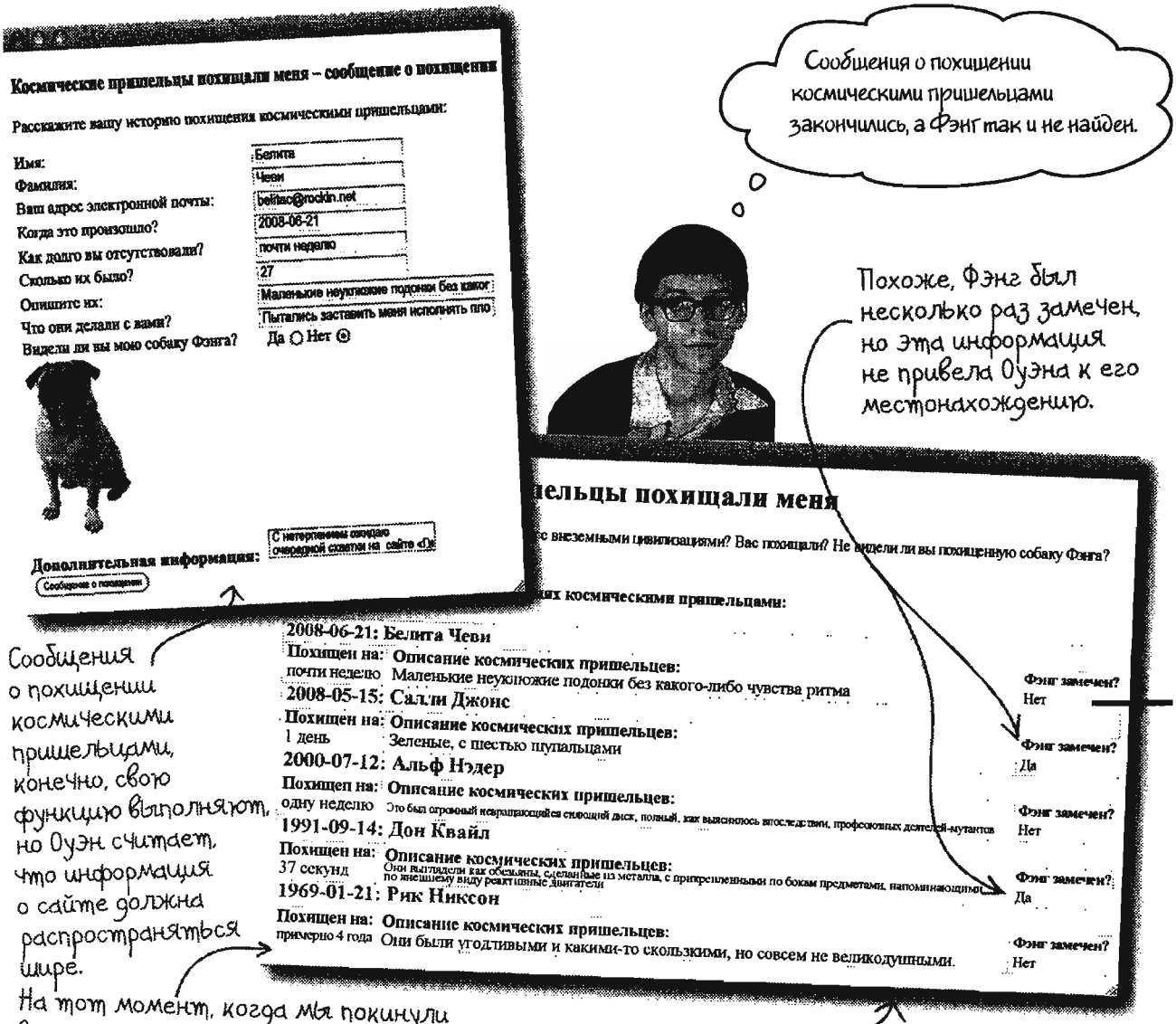
12 распространение информации и веб-сервисы

Связь с миром

Вокруг нас огромный мир, и наше веб-приложение не может позволить себе игнорировать его. Хотя, может быть, правильнее сказать, для вас было бы лучше, если окружающий вас мир не игнорировал бы ваше веб-приложение. Один из исключительно эффективных способов привлечь внимание мира к вашему веб-приложению — это сделать ваши данные доступными для распространения, что означает предоставить пользователям возможность получать на свой компьютер новости вашего сайта, вместо того чтобы они посещали его непосредственно в надежде узнать эти новости. И не только это. Ваше приложение может взаимодействовать с другими приложениями через различные веб-сервисы и использовать информацию этих приложений.

Оуэну необходимо поведать миру о Фэнге

Одна из серьезных проблем, с которой сталкиваются разработчики сайтов, заключается в том, как сделать так, чтобы посетители сайта возвращались на него снова и снова. Одно дело — завлечь посетителя, но совсем другое — пробудить в нем желание вернуться на сайт. Даже сайты с привлекательным содержанием могут терять пользователей просто потому, что не всегда удается помнить о регулярном посещении того или другого ресурса. Принимая это во внимание, Оуэн решает предложить альтернативный способ просмотра сообщений о похищении космическими пришельцами. Вместо того чтобы дожидаться, когда пользователи начнут регулярно посещать его сайт, он хочет сам отправлять им «запросы» на сообщения о похищении космическими пришельцами.

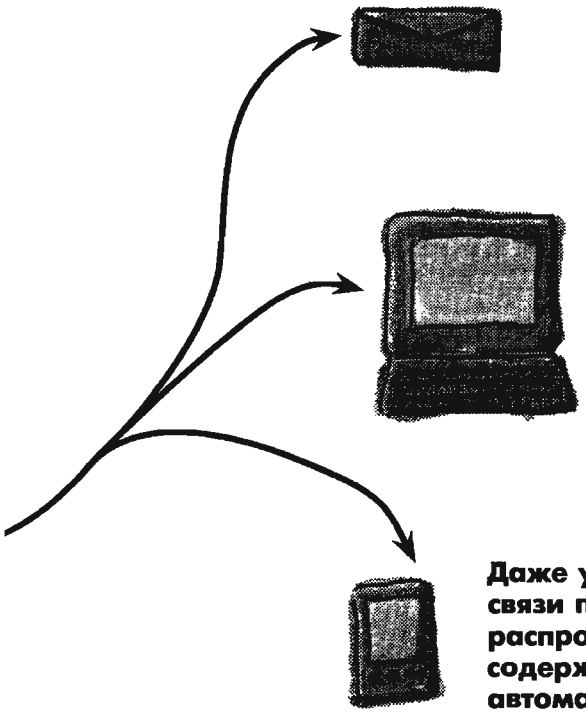


Распространение сообщений о похищении космическими пришельцами

Распространяя сообщения о похищениях космическими пришельцами среди пользователей, Оуэн создает виртуальную команду людей, которые могут помочь ему в мониторинге этих сообщений. Чем больше людей будет вовлечено в этот процесс, тем выше вероятность того, что местонахождение Фэнга будет зафиксировано и он наконец-то вернется домой.

Некоторые клиенты электронной почты поддерживают распространение содержания сайтов, позволяя вам получать обновление этого содержания точно так же, как вы получаете электронную почту.

Распространение содержания сайта среди пользователей — это эффективный способ привлечь к нему больше внимания.



Многие обычные браузеры также предоставляют вам возможность просматривать распространяемое содержание, в результате чего вы постоянно находитесь в курсе всех последних изменений на сайте.

Даже устройства мобильной связи поддерживают распространение изменений содержания сайта, которые автоматически передаются на эти устройства по мере появления на сайте.

Оуэн пока не совсем понимает, как распространять содержание сайта среди пользователей, но сама идея ему очень нравится.

Оуэн надеется, что деятельность его виртуальной команды анализаторов содержания сообщений о похищении космическими пришельцами увеличит шансы вернуть Фэнга домой.

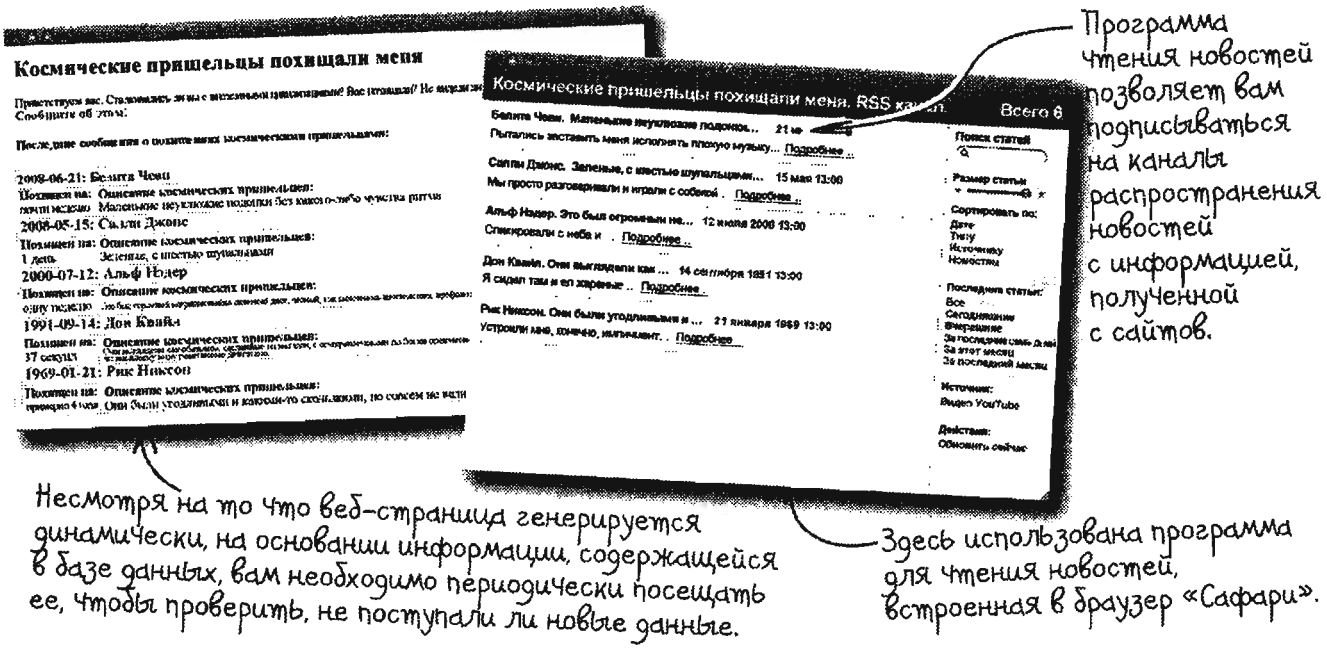
RSS используется для распространения содержания сайта

Идея, лежащая в основе размещения информации в HTML-формате в Интернете, заключается в том, чтобы предоставить людям возможность просматривать эту информацию, посещая сайты. Но что если мы хотим, чтобы пользователи получали данную информацию, не запрашивая ее? Такую возможность предоставляет RSS — формат данных, позволяющий пользователям узнавать о содержании сайта без его посещения.

HTML обеспечивает возможность просмотра информации. RSS — возможность ее распространения.

RSS — это что-то вроде веб-эквивалента записывающего DVD-видеопроигрывателя. Такой проигрыватель позволяет вам «подписаться» на определенную телевизионную программу и автоматически записывать ее, когда она выходит в эфир. Зачем переключать каналы в поисках любимой программы, если благодаря DVD-видеопроигрывателю у вас имеется возможность получить ее на экран своего телевизора автоматически? Хотя RSS и ничего не записывает, как видеопроигрыватель, он подобен ему в том смысле, что присылает информацию с сайтов, освобождая вас от необходимости разыскивать ее в Интернете.

Создавая RSS-канал для своих данных о похищениях космическими пришельцами, Оуэн хочет извещать пользователей о том, что поступил новый отчет о таком похищении. Это поддерживает интерес людей к информации, вовлекая большее количество пользователей в ее анализ. При этом данные одной и той же базы могут использоваться как для формирования содержания сайта, так и для RSS-канала.



RSS позволяет просматривать данные, которые поступают к пользователю в автоматическом режиме, сразу же, как только они становятся доступными. Такой интернет-ресурс называется RSS-каналом. Пользователям, подписавшимся на какой-либо из RSS-**каналов** и получающим соответствующую информацию, нет необходимости посещать интересующие их сайты.

Для просмотра RSS-канала пользователю необходима **программа чтения новостей**. Большинство популярных браузеров и клиентских программ электронной почты позволяют подписываться на RSS-каналы. Все, что вам нужно, — это указать URL, остальное программа чтения новостей сделает самостоятельно.

RSS — это не что иное, как XML (eXtensible Markup Language, расширяемый язык разметки (гипертекста))

RSS похож на HTML в том, что оба они являются обычными **языками разметки** гипертекста, которые используют теги и атрибуты для описания формы представления содержания. RSS базируется на XML, который является базовым языком разметки и может быть использован для описания любых видов данных. Высокая эффективность XML определяется его гибкостью. Он не определяет специальные теги и атрибуты, а только устанавливает правила, по которым любые теги и атрибуты создаются и используются. Дело производных языков, таких как HTML или RSS, — устанавливать, какие теги и атрибуты могут использоваться и как.

RSS — это язык разметки, используемый для описания содержания, предназначенного для распространения.

Для того чтобы хорошо разобраться в RSS, вы должны вначале понять основные правила XML. Они являются общими для всех языков разметки, производных от XML, включая RSS и современную версию HTML, известную под названием XHTML. Эти правила просты, но очень важны. Ваш XML(RSS)-код не будет работать, если вы нарушите их! Вот они.

Неправильно! Нет закрывающего тега, соответствующего открывающему тегу.

```
<p>Номер домашнего телефона
```

```
<p>Номер домашнего телефона</p>
```

Правильно.

- ☑ **Тегу, включающему какое-либо содержание, должен соответствовать тег, закрывающий это содержание.**

Неправильно! В пустом теге должен быть пробел и прямая наклонная черта перед закрывающей угловой скобкой (>).

```
<br>
```

```
<br />
```

Правильно.

- ☑ **Пустой тег, то есть тег, который не включает никакого содержания, должен кодироваться с пробелом и прямой наклонной чертой перед закрывающей угловой скобкой.**

Неправильно! Значение атрибута должно быть заключено в двойные кавычки.

```
<img src=alien.gif />
```

```
<img src="alien.gif" />
```

Правильно.

- ☑ **Все значения атрибутов должны заключаться в двойные кавычки.**

В отличие от PHP, в котором могут использоваться как двойные, так и одинарные кавычки, в XML для значений атрибутов допустимы только двойные кавычки.

XML — это язык разметки, используемый для описания данных любого вида.

Не бывает глупых вопросов

В: Чем же использование RSS лучше простого регулярного посещения моего сайта?

О: Если люди регулярно посещают ваш сайт в поисках последних обновлений, тогда RSS не имеет никаких преимуществ. Но большинство людей просто забывают о сайтах, которые представляли для них интерес при прежних посещениях. В этих случаях RSS демонстрирует свои преимущества, продвигая информацию вашего сайта пользователям вместо того, чтобы вынуждать их искать ее самостоятельно.

В: Откуда произошла эта аббревиатура — RSS?

О: Сейчас она рассматривается как сокращение от Really Simple Syndication, что может быть переведено на русский язык как «исключительно простой способ распространения информации». В процессе развития существовало несколько версий RSS (и толкований происхождения этой аббревиатуры), но наименование самого последнего варианта (версии 2.0) происходит от английской фразы Really Simple Syndication, и это все, что может вас волновать.

В: А из чего состоит RSS?

О: RSS — это формат данных. Аналогично тому, как HTML можно рассматривать как формат данных, который позволяет вам описывать информацию для просмотра ее в браузере, RSS — это формат данных, который описывает информацию для просмотра ее с помощью программ чтения новостей. Так же как и HTML, RSS — это текстовый формат, содержащий теги, описывающие способы вывода информации в программах чтения новостей.

В: Где мне получить программу чтения новостей в формате RSS?

О: В большинстве браузеров имеется встроенная программа чтения новостей в формате RSS. Даже некоторые клиенты электронной почты поддерживают возможность читать новости в этом формате в специальном каталоге новостей. Существуют также самостоятельные программы для этих целей.

УПРАЖНЕНИЕ

Ниже приведен код для RSS-канала приложения «Космические пришельцы похищали меня». Прокомментируйте, какую, по вашему мнению, функцию выполняет каждый выделенный тег.

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
  <channel>
    <title>Космические пришельцы похищали меня. Канал RSS</title>
    <link>http://aliensabductedme.com/</link>
    <description>Сообщения о похищении космическими пришельцами, собранные со всего
мира благодаря Оуэну и его похищенной собаке Фэнгу.</description>
    <language>ru-ru</language>

    <item>
      <title>Белита Чеви. Маленькие неуклюжие подонки без какого-…</title>
      <link>http://www.aliensabductedme.com/index.php?abduction_id=7</link>
      <pubDate>Суббота, 21 января 00:00:00</pubDate>
      <description>Пытались заставить меня исполнять плохую музыку.</description>
    </item>

    <item>
      <title>Салли Джонс. Зеленые, с шестью щупальцами…</title>
      <link>http://www.aliensabductedme.com/index.php?abduction_id=8</link>
      <pubDate>Воскресенье, 11 мая 00:00:00</pubDate>
      <description>Мы просто разговаривали и играли с собакой.</description>
    </item>

  </channel>
</rss>
```

RSS-код с комментариями

Ниже приведен код для RSS-канала приложения «Космические пришельцы похищали меня». Прокомментируйте, какую, по вашему мнению, функцию выполняет каждый выделенный тег.

Эта строка не является тегом. Это XML-директива, которая декларирует этот текст как XML-документ.

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
  <channel>
    <title>Космические пришельцы похищали меня. Канал RSS</title>
    <link>http://aliensabductedme.com/</link>
    <description>Сообщения о похищении космическими пришельцами, собранные со всего мира
благодаря Оуэну и его похищенной собаке Фэнгу.</description>
    <language>ru-ru</language>

    <item>
      <title>Велита Чеви. Маленькие неуклюжие подонки без какого-…</title>
      <link>http://www.aliensabductedme.com/index.php?abduction_id=7</link>
      <pubDate>Суббота, 21 января 00:00:00</pubDate>
      <description>Пытались заставить меня исполнять плохую музыку.</description>
    </item>

    <item>
      <title>Салли Джонс. Зеленые, с шестью щупальцами…</title>
      <link>http://www.aliensabductedme.com/index.php?abduction_id=8</link>
      <pubDate>Воскресенье, 11 мая 00:00:00</pubDate>
      <description>Мы просто разговаривали и играли с собакой.</description>
    </item>

  </channel>
</rss>
```

Этот тег <title> относится ко всему документу.

Ссылка на канал обычно указывает на сайт, ассоциируемый с этим каналом.

Для каждого канала необходимо описание, объясняющее, какого рода новости он предлагает.

Каналы могут содержать информацию на разных языках. Этот тег устанавливает язык, на котором будет распространяться информация.

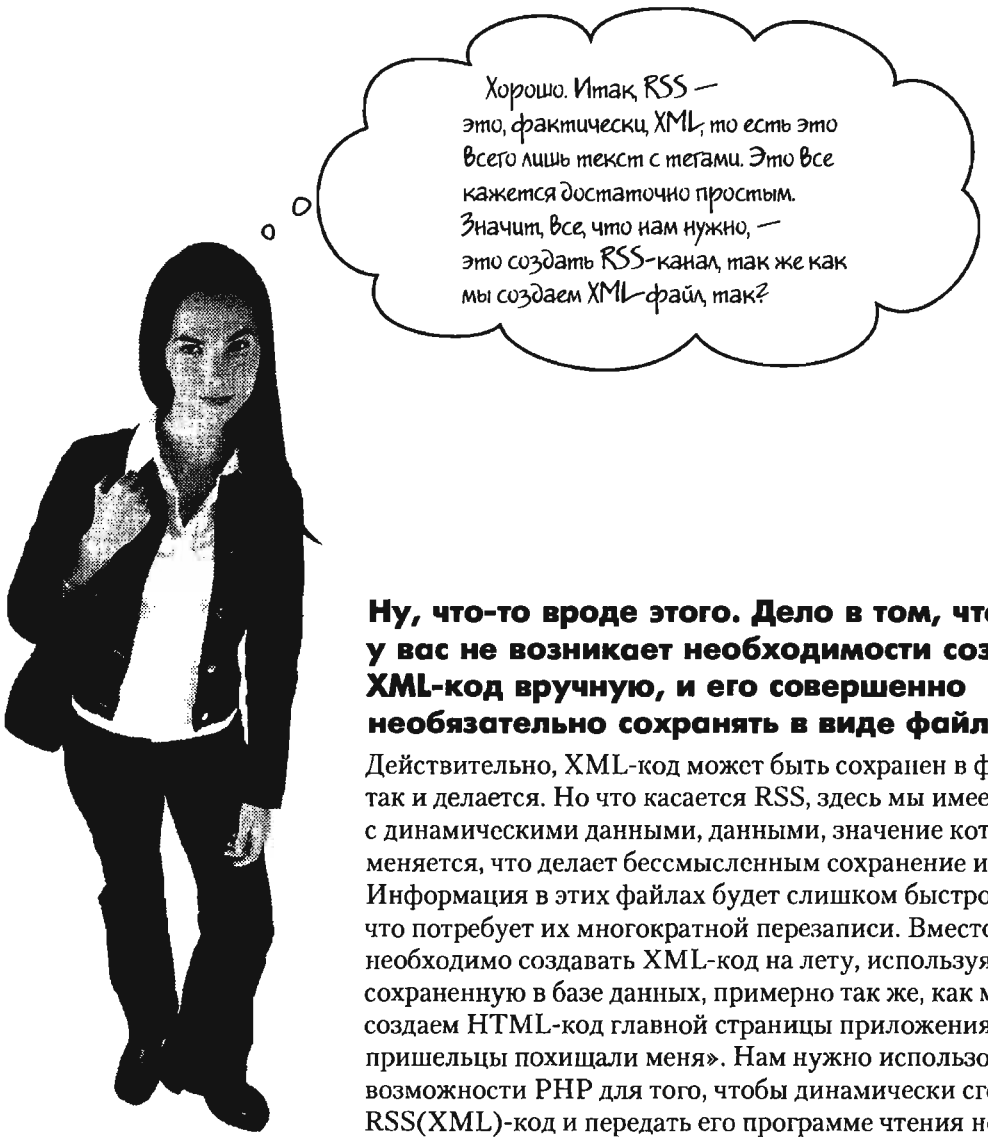
Имя похищенного и сообщение о его похищении космическими пришельцами объединены в одну строку и используются в качестве заголовка для каждой новости.

Этот RSS-документ содержит только один канал, чего вполне достаточно, если вы не хотите распределить все новости по категориям.

Каждый открывающий XML-тег имеет соответствующий ему закрывающий тег. Этот тег закрывает RSS-документ.

Дата, указанная в теге <pubDate>, соответствует формату даты, определенному RFC 822, который является стандартом для детального представления даты и времени в виде текста.

Ссылка на конкретную новость обычно указывает на URL полного содержания новости на сайте.



Ну, что-то вроде этого. Дело в том, что обычно у вас не возникает необходимости создавать XML-код вручную, и его совершенно необязательно сохранять в виде файла.

Действительно, XML-код может быть сохранен в файле, и часто так и делается. Но что касается RSS, здесь мы имеем дело с динамическими данными, данными, значение которых постоянно меняется, что делает бессмысленным сохранение их в файлах. Информация в этих файлах будет слишком быстро устаревать, что потребует их многократной перезаписи. Вместо этого нам необходимо создавать XML-код на лету, используя информацию, сохраненную в базе данных, примерно так же, как мы динамически создаем HTML-код главной страницы приложения «Космические пришельцы похищали меня». Нам нужно использовать возможности PHP для того, чтобы динамически сгенерировать RSS(XML)-код и передать его программе чтения новостей по ее запросу.

От базы данных к программе чтения новостей

Для того чтобы начать распространение информации о похищениях космическими пришельцами, Оуэну необходимо обеспечить динамическую генерацию RSS-кода, используя данные базы MySQL. Этот код сформирует законченный RSS-документ, готовый для просмотра в программе чтения новостей. Таким образом, PHP, взяв неподготовленные данные из базы, должен преобразовать их в RSS-формат, чтобы, используя программу для чтения новостей, пользователи могли ознакомиться с предоставленной информацией. Самая значительная часть этого процесса заключается в том, что как только информация становится доступной в RSS-формате, все остальное, включая демонстрацию информации пользователю, — это дело программы чтения новостей.

Программа чтения новостей RSS служит для того, чтобы получать данные, доступные через RSS-каналы.

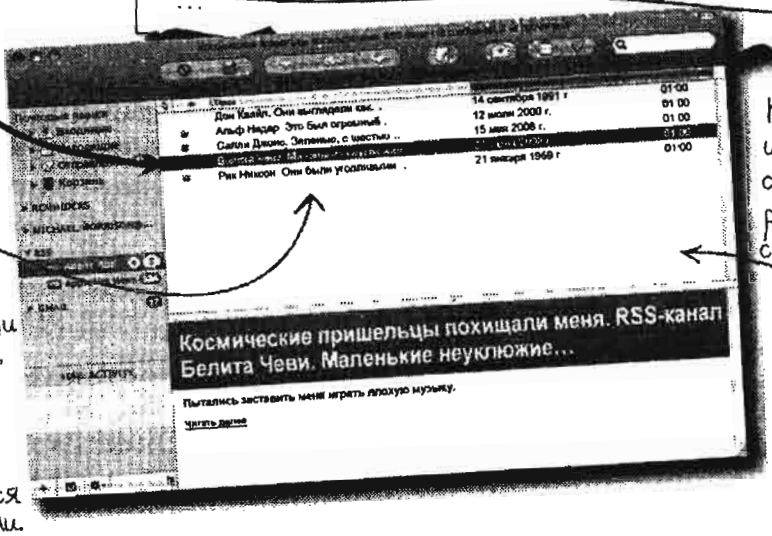
aliens_abduction

abduction_id	first_name	last_name	when_happened	how_long	how_many	alien_description	what_they_did	...
1	Альф	Надер	2007-07-12	одна неделя	по крайней мере, 12	Это был огромный невращающийся сияющий диск, полный...	Спикировали с неба и...	...
2	Дон	Квайл	1991-09-14	37 секунд	не знаю	Они выглядели как обезьяны, сделанные из металла..	Я сидел там и ел жареные...	
3	Рик	Никсон	1969-01-21	примерно 4 года	только один	Они были угодливыми и какими-то скользкими, но совсем не...	Устроили мне, конечно, напичкавит, а потом попытались ..	
4	Белита	Чеви	2008-06-21	почти	27	Маленькие неуклюжие	Пытались заставить меня исполнять плохую музыку.	
5	Салли	Джонс	2008-05-					

PHP, используя информацию, сохраненную в базе данных MySQL, создает документы, доступные через RSS-каналы.

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
 <channel>
  <title>Космические пришельцы похищали меня. Канал RSS</title>
  <link>http://aliensabductedme.com/</link>
  <description>Сообщения о похищении космическими пришельцами, собранные со всего мира
благодаря Оуэну и его похищенной собаке Фэнгу.</description>
  <language>ru-ru</language>

  <item>
   <title>Белита Чеви. Маленькие неуклюжие подонки без какого...</title>
   <link>http://www.aliensabductedme.com/index.php?abduction_id=7</link>
   <pubDate>Суббота, 21 января 00:00:00</pubDate>
   <description>Пытались заставить меня исполнять плохую музыку.</description>
  </item>
  ...
```

Программа для чтения новостей интерпретирует отдельные новости, составленные на XML, и выводит их на экран монитора для просмотра.



Каждое индивидуальное сообщение расположено в своей секции RSS-документа.

Каждая программа чтения новостей представляет новости вполне определенным, своим способом. Здесь показан способ представления, аналогичный тому, который используется почтовыми клиентами.

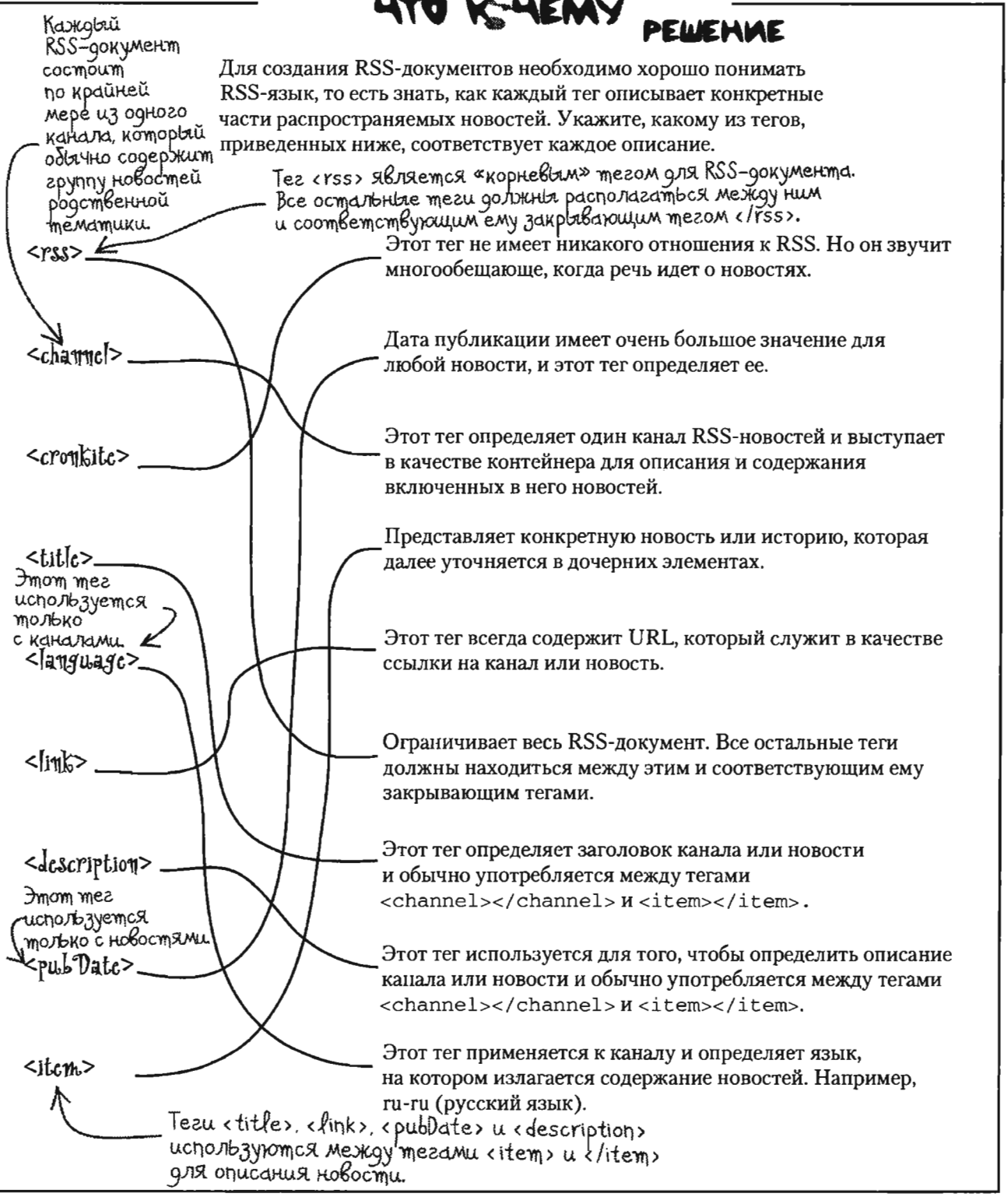
Эта программа чтения новостей встроена в стандартную программу чтения электронной почты в Mac OS X. Другие почтовые клиенты также имеют встроенные программы чтения новостей.

Для создания RSS-документов необходимо хорошо понимать RSS-язык, то есть знать, как каждый тег описывает конкретные части распространяемых новостей. Укажите, какому из тегов, приведенных ниже, соответствует каждое описание.

<rss>	Этот тег не имеет никакого отношения к RSS. Но он звучит многообещающе, когда речь идет о новостях.
<channel>	Дата публикации имеет очень большое значение для любой новости, и этот тег определяет ее.
<cronkite>	Этот тег определяет один канал RSS-новостей и выступает в качестве контейнера для описания и содержания включенных в него новостей.
<title>	Представляет конкретную новость или историю, которая далее уточняется в дочерних элементах.
<language>	Этот тег всегда содержит URL, который служит в качестве ссылки на канал или новость.
<link>	Ограничивает весь RSS-документ. Все остальные теги должны находиться между этим и соответствующим ему закрывающим тегами.
<description>	Этот тег определяет заголовок канала или новости и обычно употребляется между тегами `<channel></channel>` и `<item></item>`.
<pubDate>	Этот тег используется для того, чтобы определить описание канала или новости и обычно употребляется между тегами `<channel></channel>` и `<item></item>`.
<item>	Этот тег применяется к каналу и определяет язык, на котором излагается содержание новостей. Например, ru-ru (русский язык).

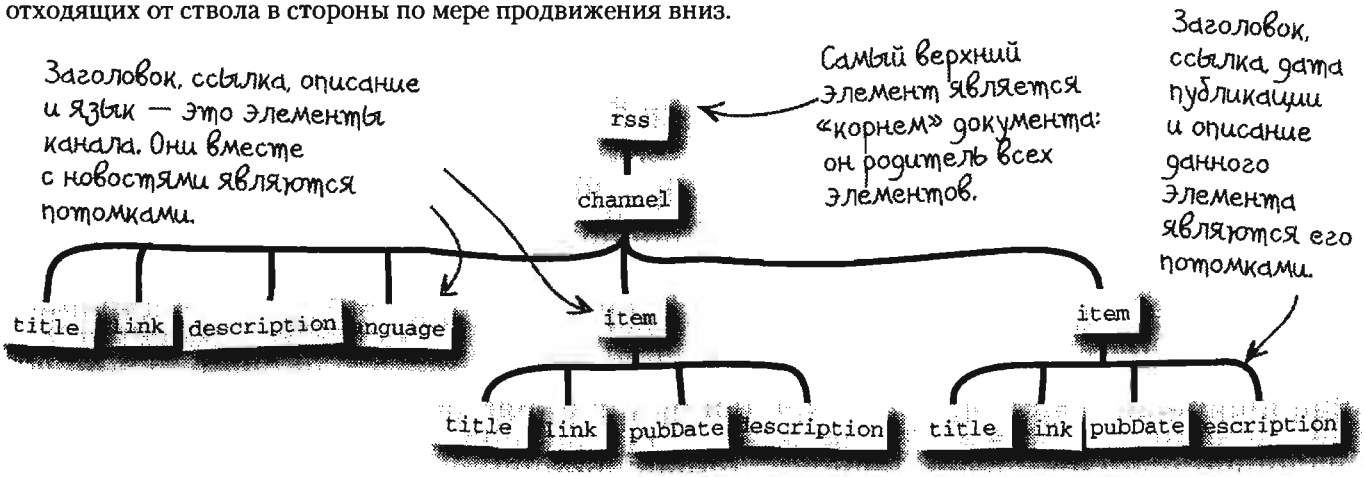
Что к чему — решение

Для создания RSS-документов необходимо хорошо понимать RSS-язык, то есть знать, как каждый тег описывает конкретные части распространяемых новостей. Укажите, какому из тегов, приведенных ниже, соответствует каждое описание.



Визуализация ~~XML~~ RSS

Вы уже знаете, что XML-код состоит из тегов, которые иногда называются **элементами**, образующими отношения типа **«родитель — потомок»** внутри XML-документа. Очень полезно иметь возможность визуализации этих отношений, когда вы работаете с XML-кодом. Например, RSS-документ может быть представлен визуально в виде иерархии элементов данных новостей, напоминающей генеалогическое древо с родительским элементом, расположенным в самой верхней точке, и потомками в виде ветвей (листьев), отходящих от ствола в стороны по мере продвижения вниз.



УПРАЖНЕНИЕ

Ниже приведено новейшее сообщение о похищении космическими пришельцами, добавленное в таблицу `aliens_abduction`. Напишите XML-код для RSS-тега `<item>` этого сообщения, соблюдая все требования формата RSS для каналов новостей.

aliens_abduction

abduction_id	first_name	last_name	when_it_happened	how_long	how_many	alien_description	what_they_did	...
...								
14	Шил	Ватнер	2008-07-05	два часа	не знаю	В небе вспыхнул яркий свет...	Они спустили меня в луче света на бензозаправочную станцию...	...

упражнение решение

Ниже приведено новейшее сообщение о похищении космическими пришельцами, добавленное в таблицу `aliens_abduction`. Напишите XML-код для RSS-тега `<item>` этого сообщения, соблюдая все требования формата RSS для каналов новостей.

aliens_abduction

abduction_id	first_name	last_name	when_it_happened	how_long	how_many	alien_description	what_they_did	...
...								
14	Шил	Ватнер	2008-07-05	два часа	не знаю	В небе вспыхнул яркий свет...	Они спустили меня в луче света на бензозаправочную станцию...	...

Все данные по конкретной новости находятся между открывающим <item> и закрывающим </item> тегами.

Между открывающим и закрывающим тегами <title></title>, <link></link>, <pubDate></pubDate> заключены детали каждой конкретной новости.

```
<item>
  <title>Шил Ватнер. В небе вспыхнул яркий свет...</title>
  <link>http://www.aliensabductedme.com/index.php?abduction_id=14</link>
  <pubDate>Суббота, 05 июля 2008 00:00:00</pubDate>
  <description>Они спустили меня в луче света на бензозаправочную станцию...</description>
</item>
```

Тег <pubDate> должен записываться буквами в разных регистрах, то есть прописной буквой D и остальными строчными буквами. Не допускается написание этого тега в формах <pubdate> или <PUBDATE>.

Не бывает глупых вопросов

В: XML чувствителен к регистру букв, используемых для написания тегов?

О: Да. Поэтому необходимо быть внимательным при выборе строчных и прописных букв при написании тегов и атрибутов. Хороший пример — тег RSS `<pubDate>`, который должен быть написан всеми строчными буквами кроме буквы D — она должна быть прописной. Большинство тегов XML пишутся либо строчными буквами, либо строчными и прописными.

В: А что по поводу пробельных символов? Как они используются в XML?

О: Прежде всего отметим, что пробельными символами в XML являются: символ возврата каретки (`\r`), символ новой строки (`\n`), символ табуляции (`\t`) и символ пробела (`' '`). В большинстве случаев в документах XML пробельные символы используются в служебных и эстетических целях, например для выделения фрагмента кода путем отступа его на несколько символов пробела. Эти «несущественные» пробельные символы обычно игнорируются интерпретаторами XML, используемыми приложениями, которые работают с XML-документами, такими, например, как программы чтения RSS-новостей. Но пробельные символы, встречающиеся внутри тегов, рассматриваются как «существенные» и обычно обрабатываются в соответствии с требованиями синтаксиса языка. Все это позволяет успешно представить средствами XML такие перегруженные пробельными символами тексты, как стихи.

В: Могут новости RSS содержать изображения?

О: Да. Только имейте в виду, что не каждая программа чтения новостей может с ними работать. В RSS 2.0 вы не можете непосредственно добавить изображение в отдельные новости, а только в канал в целом. Вы можете добавить изображение в канал, используя тег `<image>`, который должен быть между открывающим канал тегом `<channel>` и закрывающим `</channel>`. Ниже приведен пример:

```
<image>
  <url>http://www.aliensabductedme.com/fang.jpg</url>
  <title>My dog Fang</title>
  <link>http://www.aliensabductedme.com</link>
</image>
```

Технически в RSS 2.0 можно добавить изображение в конкретную новость. Трюк заключается в использовании HTML-тега `<img>` в описании новости. Хотя такая возможность и существует, она требует от вас включения HTML-кода в XML-код, и это во многих аспектах входит в противоречие с базовым принципом рассмотрения RSS-документа как чистого текста.

RSS: близкий взгляд

Интервью этой недели: Что придает репортеру силы

Корреспондент редакции Head First: Итак, я слышал, что, когда люди ищут новости в Интернете, они обращаются к вашей помощи. Это правда?

RSS: Я думаю, это зависит от того, какой смысл вы вкладываете в слово «новости». В большинстве случаев я занимаюсь тем, что упаковываю информацию в форму, которая легкодоступна для читателей новостей. А является эта информация новостью или нет... это, вообще говоря, вне моей компетенции. Это решать людям, которые имеют дело с данной информацией.

Корреспондент редакции Head First: Значит, под выражением «читатели новостей» вы подразумеваете людей, читающих новости?

RSS: Нет, я имею в виду программы, которые понимают, что я такое и как я представляю информацию. Например, большое количество программ обработки электронной почты работают со мной, а это означает, что вы можете подписаться на какой-либо канал и получать все доступные обновления точно так же, как вы получаете электронную почту.

Корреспондент редакции Head First: Интересно. Тогда чем же вы отличаетесь от обычной электронной почты?

RSS: О, это отличие очень значительно. Электронное письмо посылается от одного человек к другому и обычно является частью двухстороннего диалога. То есть вы можете ответить на электронное письмо, получить, в свою очередь, на него ответ и т. д. Я же работаю исключительно в одном направлении: от сайта к подписчику.

Корреспондент редакции Head First: И как же организуется это одностороннее общение?

RSS: Когда человек решает получать новости, подписавшись на них через свою программу чтения новостей, он обычно хочет получать все обновления, появляющиеся на данном сайте. Когда на сайте появляется новая информация, я оформляю ее в таком виде, чтобы программа чтения новостей могла узнать об этом событии, и передаю все это ей для того, чтобы подписчик мог ознакомиться с данной информацией. Но у подписчика отсутствует какая-либо возможность ответа, что и объясняет односторонний характер общения сайта с подписчиком.

Корреспондент редакции Head First: Я понял. Тогда кто же вы на самом деле?

RSS: Фактически я просто формат данных, согласованный способ сохранения информации в виде, в котором эта информация может быть получена и обработана программой чтения новостей. Используйте меня для сохранения информации, и программы чтения информации смогут получить доступ к ней.

Корреспондент редакции Head First: Хорошо. Тогда в чем заключается ваше отличие от HTML?

RSS: Мы оба являемся текстовыми форматами данных, основанными на XML, то есть мы оба используем теги и атрибуты для описания этих данных. Но если HTML был разработан специально для обработки и вывода информации пользователю с помощью браузеров, я предназначен для обработки и вывода информации пользователю с помощью программ чтения новостей. Можно сказать, что мы рассматриваем одни и те же данные с разных точек зрения.

Корреспондент редакции Head First: Но мне приходилось видеть браузеры, которые предоставляют возможность читать RSS-новости. Как это происходит?

RSS: Хороший вопрос. Дело в том, что некоторые браузеры имеют встроенные программные модули, предназначенные для получения и чтения RSS-новостей. И должны рассматриваться как приложение, выполняющее две (в данном случае) разные функции. И когда вы читаете RSS-новости в таком браузере, это не имеет ничего общего с просмотром HTML-страниц.

Корреспондент редакции Head First: Но ведь большинство RSS-новостей ссылаются на веб-страницы HTML, не так ли?

RSS: Правильно. Поэтому я и работаю рука об руку с HTML, чтобы предоставить наиболее удобный доступ к информации, опубликованной на сайтах. Основная идея заключается в том, что вы используете меня для того, чтобы узнать о новой информации, появившейся на сайте, без необходимости непосредственно посещать сайт. В программе чтения новостей вы имеете дело только с новой информацией, опубликованной на сайте, и если вы увидите что-то, что вас заинтересует, и захотите ознакомиться с этим подробнее, вы простым щелчком кнопкой мыши можете перейти по ссылке к интересующему вас материалу. Именно поэтому в каждой новости содержится ссылка на оригинальную веб-страницу.

Корреспондент редакции Head First: Можно сказать, вы предоставляете возможность предварительного просмотра обновленных веб-страниц.

RSS: Да, что-то вроде этого. И не забывайте: я прихожу к вам, но вам нет необходимости приходить ко мне. Это одна из моих особенностей, которая нравится людям. Я избавляю их от необходимости многократного посещения сайта в поисках новой информации.

Корреспондент редакции Head First: Я понял. Это действительно очень удобно. Благодарю вас за то, что вы прояснили свою роль в Интернете.

RSS: Я очень рад. Всего хорошего.

Динамическое создание RSS-документа

Понимание формата RSS — это, конечно, очень хорошо, но Оуэну необходим RSS-канал для того, чтобы распространять среди пользователей Интернета сообщения о похищениях космическими пришельцами. Пришло время привлечь PHP к динамическому созданию RSS-документов с извлеченной из базы данных Оуэна информацией о похищениях космическими пришельцами. К счастью, все это может быть успешно выполнено с разбивкой на следующие этапы:

Результат должен быть сохранен в файле как XML-документ.

1 Определение типа документа как XML.

В заголовке RSS-документа мы должны указать его тип как XML.

```
<?php header('Content-Type: text/xml'); ?>
```

2 Создание директивы XML, указывающей на то, что этот документ является XML-документом.

```
<?php echo '<?xml version="1.0" encoding="utf-8"?>'; ?>
```

3 Создание статического RSS-кода, для которого не нужны данные, сохраненные в базе. Это тег <rss> и информация о канале.

```
<rss version="2.0">
  <channel>
    <title>...
    <link>...
    <description>...
    <language>...
```

Этот код не зависит от данных, сохраненных в базе. Он всегда один и тот же для данного документа.

4 Выполнение запроса к базе данных aliens_abduction для извлечения информации о похищении космическими пришельцами.

Прежде чем генерировать RSS-код для новости, нам необходимо запросить в базе данных MySQL информацию о похищении космическими пришельцами.

5 Прохождение в цикле через данные результата запроса и создание RSS-кода для каждой новости.

```
<item>
  <title>...
  <link>...
  <pubDate>...
  <description>...
</item>
```

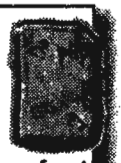
Этот код содержит данные, извлеченные из базы для каждой новости.

6 Создание статического RSS-кода, необходимого для окончания документа, в том числе закрывающих тегов </channel> и </rss>.

```
  </channel>
</rss>
```

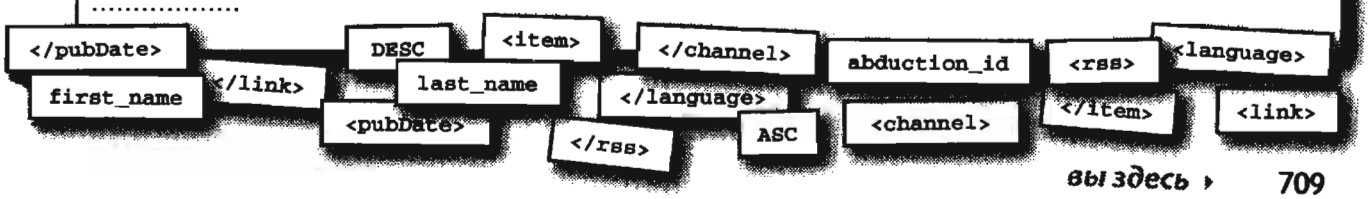
Магниты PHP и MySQL и XML!

В сценарии создания RSS-канала в приложении Оуэна «Космические пришельцы похищали меня» (newsfeed.php) пропущена важная часть кода. Внимательно выберите магниты для того, чтобы вставить их в нужные места сценария и создать RSS-канал в динамическом режиме.



newsfeed.php

```
<?php header('Content-Type: text/xml'); ?>
<?php echo '<?xml version="1.0" encoding="utf-8"?>'; ?>
<rss version="2.0">

  <title>Космические пришельцы похищали меня. Канал RSS</title>

  .....http://aliensabductedme.com/.....

3

  <description>Сообщения о похищении космическими пришельцами, собранные со всего мира
благодаря Оуэну и его похищенной собаке Фэнгу.</description>

  .....ru-ru.....

<?php
  require_once('connectvars.php');

  // Подключение к базе данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  // Извлечение свидетельств о космических пришельцах из базы данных MySQL
  $query = "SELECT abduction_id, first_name, last_name, " .
    "DATE_FORMAT(when_it_happened,'%a, %d %b %Y %T') AS when_it_happened_rfc, " .
    "alien_description, what_they_did " .
4   "FROM aliens_abduction " .

    "ORDER BY when_it_happened .....";

  $data = mysqli_query($dbc, $query);

  // Прохождение в цикле свидетельств о космических пришельцах и преобразование этих данных в RSS-формат
  while ($row = mysqli_fetch_array($data)) {
    // Вывод каждой записи результата запроса в виде RSS-новости

    echo '.....';

    echo '  <title>' . $row['first_name'] . ' ' . $row['last_name'] . ' - ' .
5     substr($row['alien_description'], 0, 32) . '...</title>';
    echo '  <link>http://www.aliensabductedme.com/index.php?abduction_id=' .

      $row['.....'] . '</link>';

    echo '.....' . $row['when_it_happened_rfc'] . ' ' . date('T') . '.....';

    echo '  <description>' . $row['what_they_did'] . '</description>';
    echo '</item>';
  }
?>

  </channel>

6

.....
```



Магниты PHP и MySQL и XML!

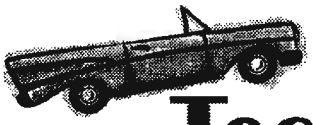
В сценарии создания RSS-канала в приложении Оуэна «Космические пришельцы похищали меня» (newsfeed.php) пропущена важная часть кода. Внимательно выберите магниты для того, чтобы вставить их в нужные места сценария и создать RSS-канал в динамическом режиме.

Аналогично тому, как в случае с CAPTCHA мы использовали заголовок для вывода изображения PNG, этот заголовок определяет тип выводимого документа как XML.

newsfeed.php

```
1 <?php header('Content-Type: text/xml'); ?>
2 <?php echo '<?xml version="1.0" encoding="utf-8"?>'; ?>
<rss version="2.0">

  <channel>

    <title>Космические пришельцы похищали меня. Канал RSS</title>

    <link>http://aliensabductedme.com/</link>
3
    <description>Сообщения о похищении космическими пришельцами, собранные со всего мира
благодаря Оуэну и его похищенной собаке Фэнгу.</description>

    <language>ru-ru</language>

<?php
  require_once('connectvars.php');

  // Подключение к базе данных
  $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

  // Извлечение свидетельств о космических пришельцах из базы данных MySQL
  $query = "SELECT abduction_id, first_name, last_name, " .
    "DATE_FORMAT(when_it_happened,'%a, %d %b %Y %T') AS when_it_happened_rfc, " .
    "alien_description, what_they_did " .
4   "FROM aliens_abduction " .

    "ORDER BY when_it_happened DESC";

  $data = mysqli_query($dbc, $query);

  // Прохождение в цикле свидетельств о космических пришельцах и преобразование этих данных в RSS-формат
  while ($row = mysqli_fetch_array($data)) {
    // Вывод каждой записи результата запроса в виде RSS-новости

    echo '<item>';

    echo '  <title>' . $row['first_name'] . ' ' . $row['last_name'] . ' - ' .
5     substr($row['alien_description'], 0, 32) . '...</title>';
    echo '  <link>http://www.aliensabductedme.com/index.php?abduction_id=' .

      $row['abduction_id'] . '</link>';

    echo '  <pubDate>' . $row['when_it_happened_rfc'] . ' ' . date('T') . '</pubDate>';

    echo '  <description>' . $row['what_they_did'] . '</description>';
    echo '</item>';
  }
?>

  </channel>
6
</rss>
```

first_name last_name </channel> ASC <rss> </item>



Тест-драйв

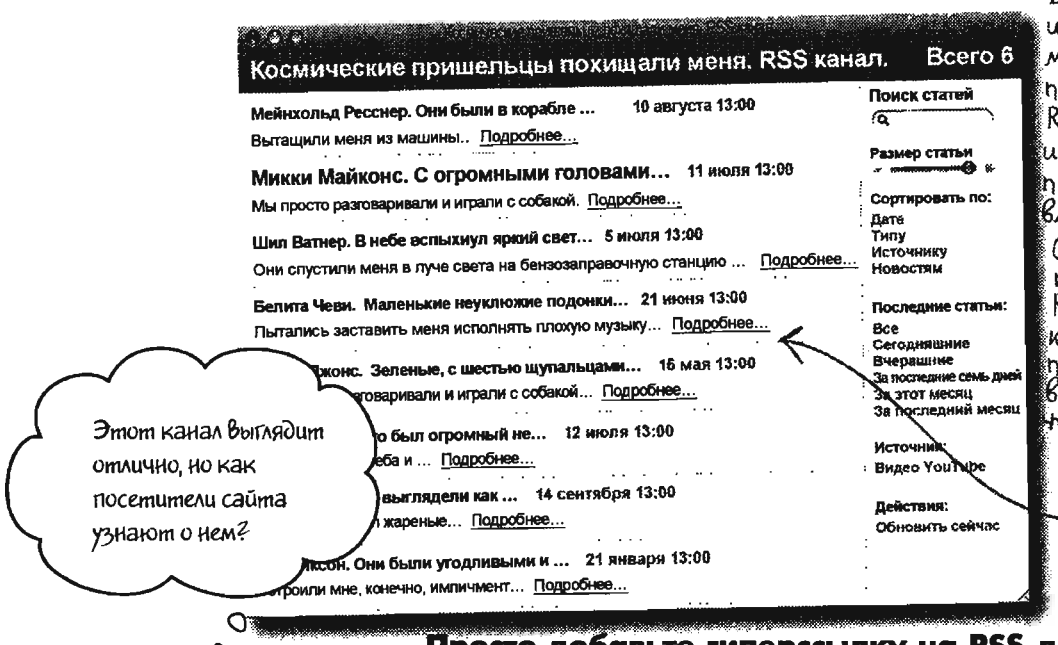
Добавьте сценарий «Космические пришельцы похищали меня. Канал RSS» в приложение «Космические пришельцы похищали меня».

Создайте новый текстовый файл с именем `newsfeed.php` и поместите в него код сценария «Космические пришельцы похищали меня. RSS канал» из упражнения с магнитами на предыдущей странице (или загрузите сценарий с сайта по адресу `www.headfirstlabs.com/books/hfphp`).

Загрузите сценарий на ваш веб-сервер и откройте его в программе чтения новостей. Многие браузеры и программы по работе с электронной почтой имеют встроенные модули, которые позволяют просматривать RSS-каналы, поэтому, если у вас нет отдельной программы для просмотра новостей, попробуйте вначале эти модули. В результате выполнения сценария вы должны увидеть последние сообщения о похищениях, извлеченные непосредственно из базы данных приложения «Космические пришельцы похищали меня».

Если ваш браузер имеет встроенный модуль, позволяющий просматривать RSS-каналы, попробуйте использовать в качестве префикса URL `feed://` вместо `http://`.

Сценарий `newsfeed.php` создает RSS-документ, который можно просматривать в любой программе чтения новостей.



Просто добавьте гиперссылку на RSS-документ на вашей домашней странице.

Не забывайте, что `newsfeed.php` — не более чем PHP-сценарий. Единственное отличие его от большинства сценариев, которые вы видели в этой книге до сих пор, заключается в том, что в результате его выполнения создается RSS-документ вместо HTML-документа. Но вы можете получить к нему доступ так же просто, как и к любому другому PHP-сценарию. Достаточно указать его URL. Что Оуэн пока упустил — так это способ доступа к сценарию для тех людей, которые посещают его сайт. Это осуществляется очень просто: добавлением на сайт гиперссылки на RSS-канал, то есть на сценарий `newsfeed.php`.

Ссылка на RSS-канал

Очень важно предусмотреть на своем сайте хорошо заметную гиперссылку на RSS-канал, что по достоинству оценят многие посетители. Для того чтобы помочь пользователям Интернета быстро найти ссылку на RSS-канал для сайта, который они сейчас просматривают, установлен стандартный ярлык. Вы можете использовать его для того, чтобы визуально объявить о предоставлении вашим сайтом такого сервиса. Мы сможем использовать этот ярлык при создании гиперссылки на RSS-канал в нижней части страницы сайта Оуэна (`index.php`).

Для RSS установлен стандартный ярлык, использование которого поможет пользователям понять, что вы предоставляете доступ к RSS-каналу.

Изображение этого ярлыка доступно во многих форматах и размерах, но общий вид всегда один и тот же.

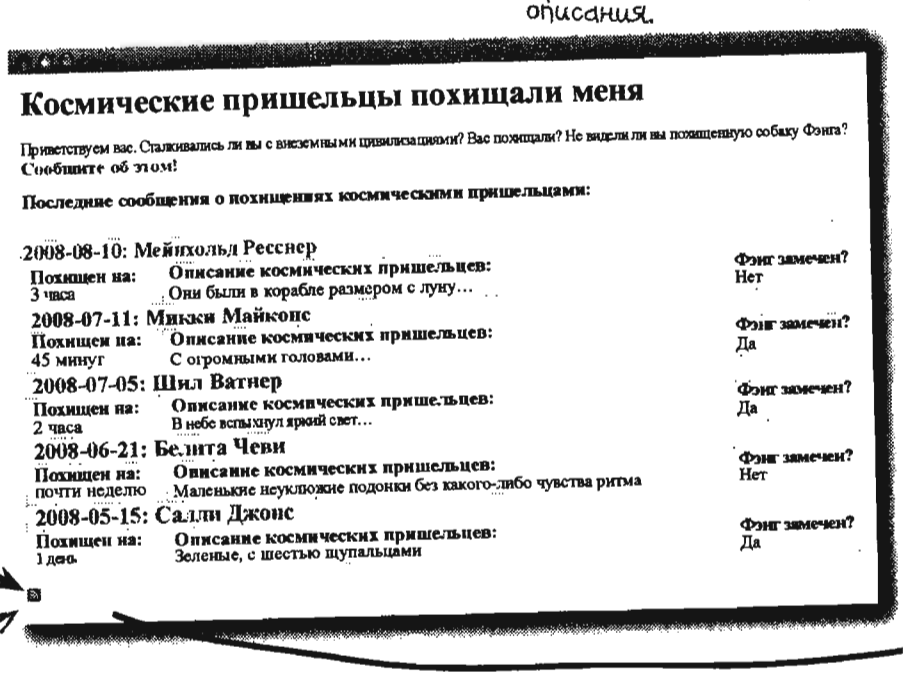
Для того чтобы загрузить коллекцию изображений ярлыков RSS других цветов и форматов, обратитесь на сайт www.feedicons.com.

В качестве URL RSS-канала выступает имя сценария newsfeed.php, что будет работать в том случае, если файл сценария сохранен в том же самом каталоге, что и файл главной страницы сайта.

```
<p>
  <a href="newsfeed.php">
    <img style="vertical-align:top; border:none"
      src="rssicon.png" alt="Получение доступа к RSS-каналу" />
    Щелкните кнопкой мыши, чтобы получить доступ к RSS-каналу.
  </a>
</p>
```

Гиперссылка на RSS-канал содержит как изображение ярлыка, так и текст описания.

Хорошо заметная гиперссылка на главной странице приложения «Космические пришельцы похищали меня» предоставляет посетителям быстрый способ доступа к RSS-каналу.

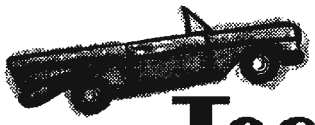


Космические пришельцы похищали меня

Приветствуем вас. Сталкивались ли вы с внеземными цивилизациями? Вас похищали? Не видели ли вы похищенную собаку Фэнга? Сообщите об этом!

Последние сообщения о похищениях космическими пришельцами:

2008-08-10: Мейнхольд Реснер		
Похищен на: 3 часа	**Описание космических пришельцев:** Они были в корабле размером с луну…	**Фэнг замечен?** Нет
2008-07-11: Микки Майконс		
Похищен на: 45 минут	**Описание космических пришельцев:** С огромными головами…	**Фэнг замечен?** Да
2008-07-05: Шил Ватнер		
Похищен на: 2 часа	**Описание космических пришельцев:** В небе вспыхнул яркий свет…	**Фэнг замечен?** Да
2008-06-21: Белита Чеви		
Похищен на: почти неделю	**Описание космических пришельцев:** Маленькие неуклюжие подонки без какого-либо чувства ритма	**Фэнг замечен?** Нет
2008-05-15: Салли Джонс		
Похищен на: 1 день	**Описание космических пришельцев:** Зеленые, с шестью щупальцами	**Фэнг замечен?** Да

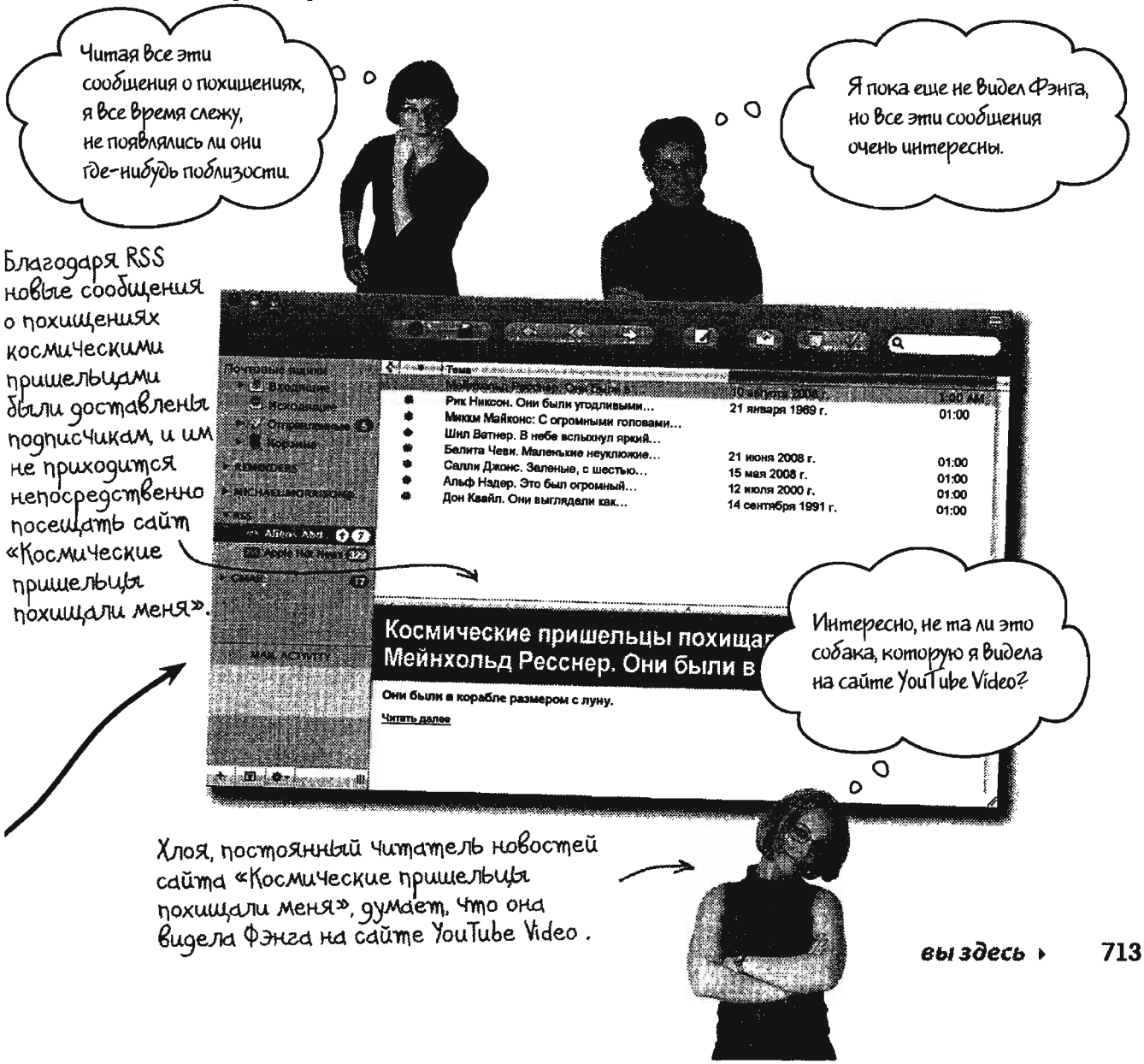


Тест-драйв

Добавьте гиперссылку на сценарий «Космические пришельцы похищали меня. Канал RSS» в приложение «Космические пришельцы похищали меня».

Внесите в сценарий `index.php` приложения «Космические пришельцы похищали меня» изменения, устанавливающие в нижней части главной страницы гиперссылку для доступа к RSS-каналу. Загрузите также изображение ярлыка RSS `rssicon.png` с сайта по адресу `www.headfirstlabs.com/books/hfphp`.

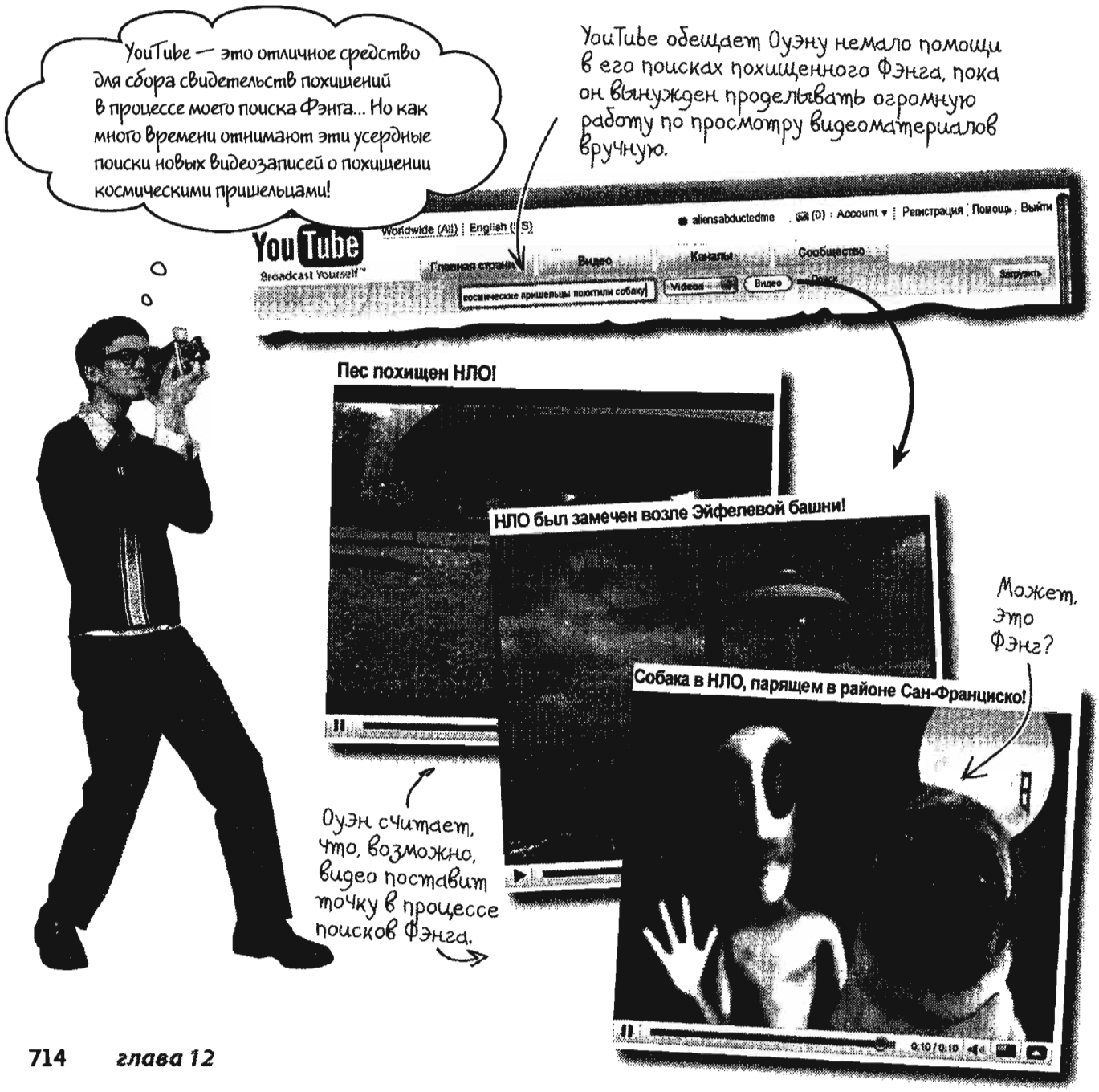
Загрузите сценарий `index.php` и файл изображения `rssicon.png` на ваш веб-сервер и откройте приложение в браузере. Щелкните кнопкой мыши на новой гиперссылке, чтобы просмотреть RSS-канал.



добавление содержания YouTube к сайту Оуэна

Видео ~~Изображение~~ красноречивее миллиона ~~тысячи~~ слов

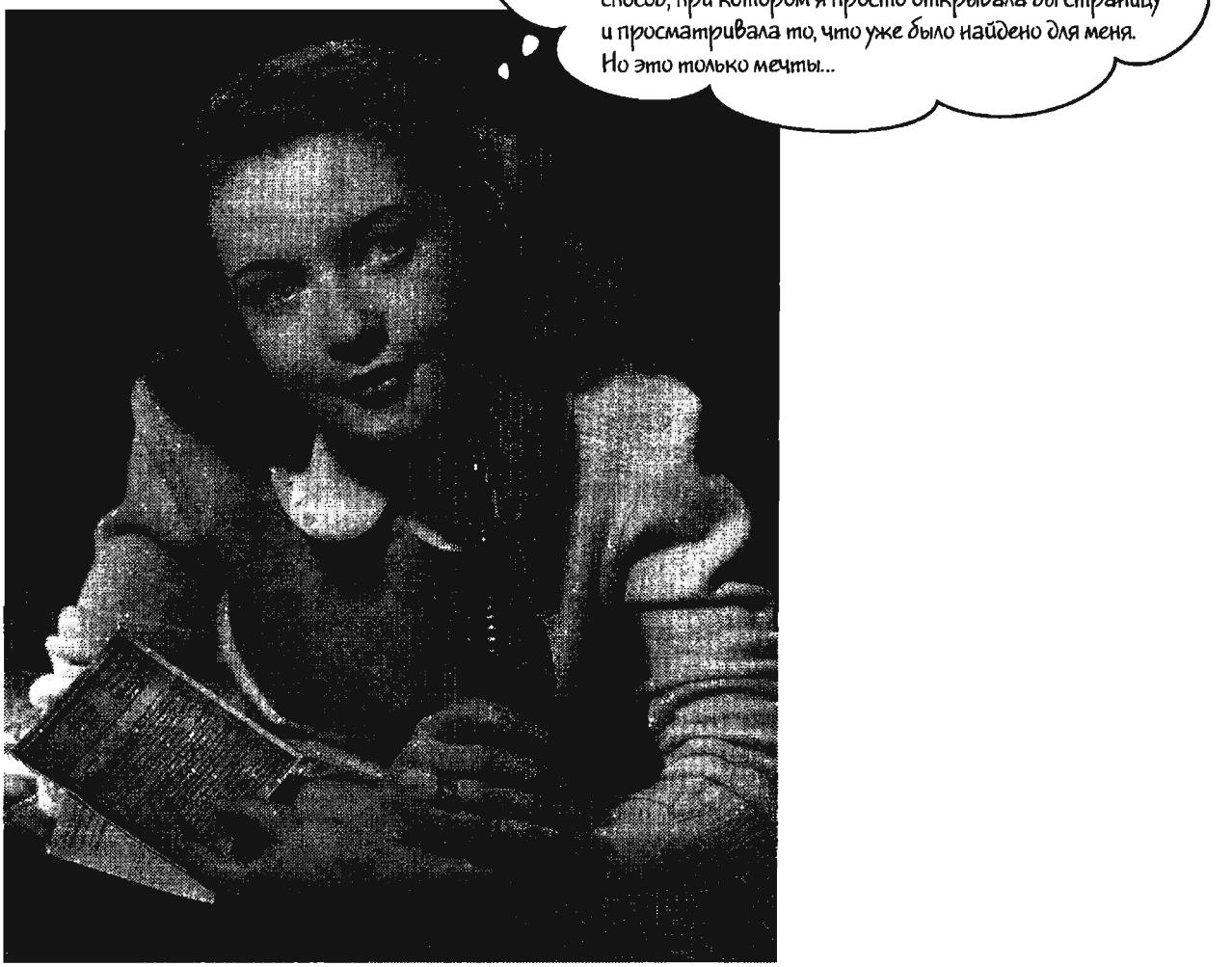
После того как один из подписчиков на RSS-канал предупредил Оуэна о видео на YouTube с собакой, похожей на Фэнга, Оуэн решил, что ему необходимо использовать еще одну технологию для расширения поисков Фэнга. Но как? Если бы он смог включить видеозаписи с сайта YouTube в приложение «Космические пришельцы похищали меня», его пользователи приняли бы участие в поисках Фэнга. И не только. Ему действительно необходимо найти способ, позволяющий не заниматься постоянно поиском на YouTube сообщений о похищенном Фэнге, а получать эти сообщения автоматически.



Попробуйте это!

1. **Посмотрите видео Оуэна по адресу www.youtube.com/user/aliensabductedme.**

2. **Посмотрите несколько видеозаписей о похищении космическими пришельцами, которые нашел Оуэн. Как вы думаете, собака, показанная на этих видео, — это Фэнг?**

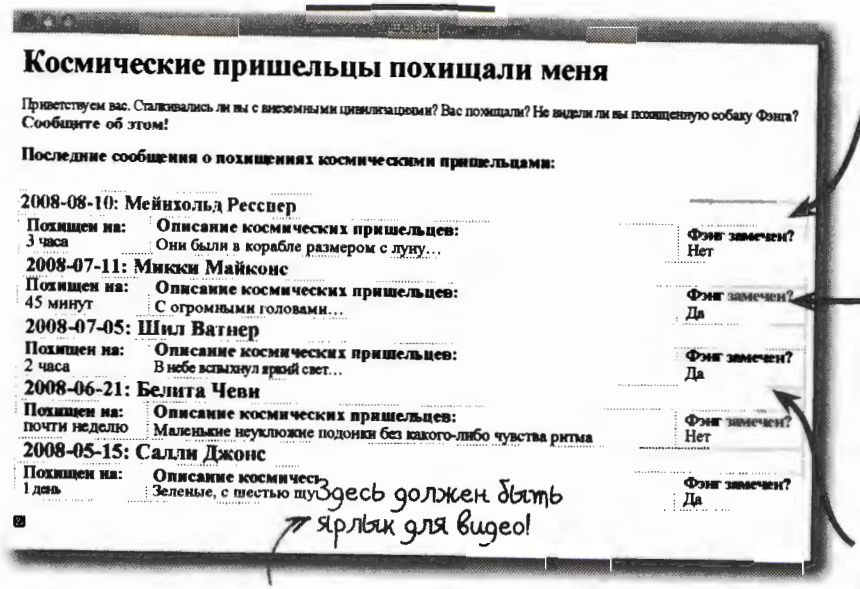


Извлечение содержания из других веб-ресурсов

Основная идея RSS заключается в том, что вы распространяете содержание своего сайта среди пользователей, так что у них не возникает необходимости постоянно посещать ваш сайт в расчете на то, что там появилось новое содержание. Как убедился Оуэн, это значительно более удобный способ следить за содержанием его сайта. Но у медали «распространение информации» есть и другая сторона, и она включает **извлечение содержания другого сайта** и размещение его на своем. Вы становитесь потребителем, в то время как этот другой — провайдером. В случае показа видеозаписей сайта YouTube на сайте Оуэна YouTube становится провайдером.

Сайт YouTube — это провайдер видеозаписей.

Приложение «Космические пришельцы похищали меня» — это потребитель видеозаписей.



Дизайн главной страницы приложения «Космические пришельцы похищали меня» должен быть слегка изменен для того, чтобы поместить на нее результаты поиска видеозаписей.

Очень важно понимать, что Оуэн хочет не просто вставить какую-то видеозапись с YouTube или гиперссылку на нее. Это сделать достаточно просто копированием HTML-кода с YouTube. Оуэн хочет **производить поиск** видеозаписей на YouTube и показывать на своей странице результаты этого поиска. Поэтому приложению «Космические пришельцы похищали меня» необходимо делать запросы на получение данных с сайта YouTube и выводить результаты этих запросов в динамическом режиме. Это позволит Оуэну и его легиону искателей Фэнга поминутно следить за присылаемыми на YouTube видеозаписями о похищениях космическими пришельцами.

Видеозаписи, полученные в результате поиска похищений космическими пришельцами на сайте YouTube, передаются на сайт Оуэна и включаются в содержание главной страницы.

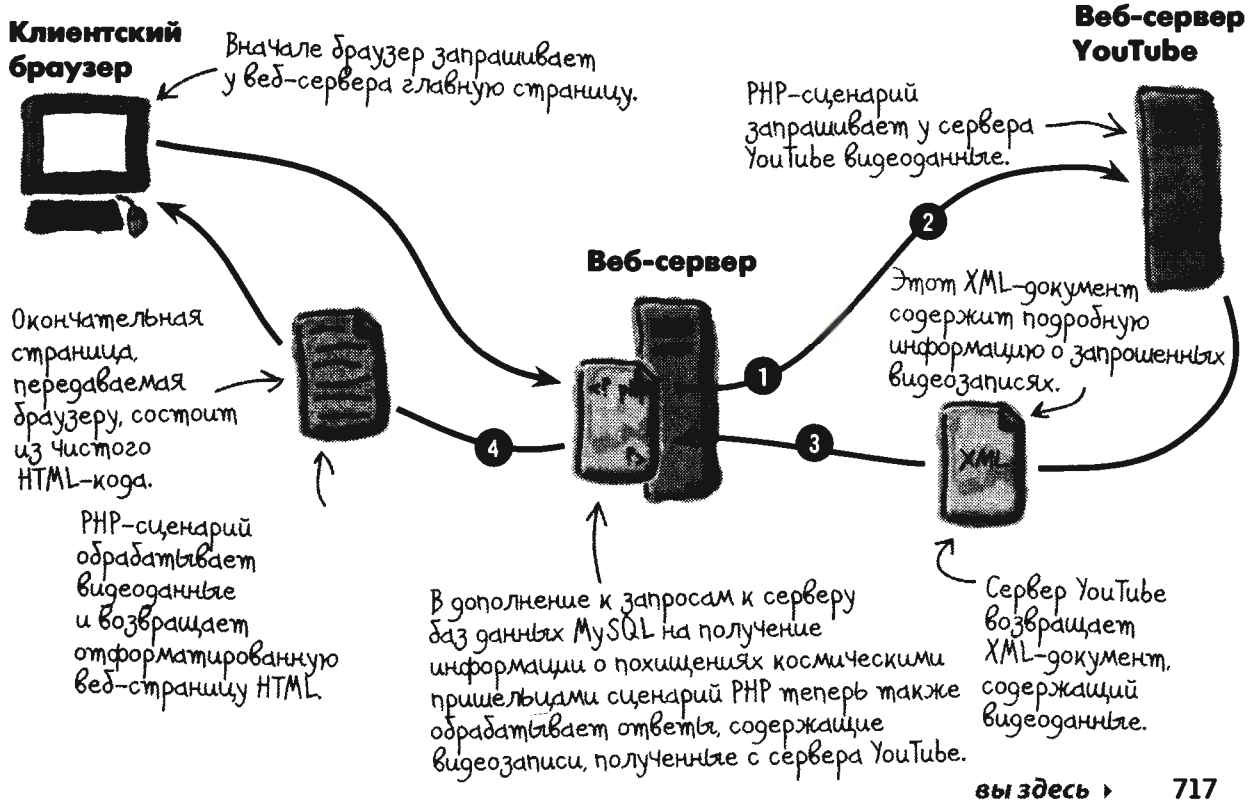
Распространение видеозаписей сайта YouTube

Для того чтобы получать видеозаписи с сайта YouTube, мы должны точно знать, как YouTube предоставляет свои видеозаписи для распространения. Это осуществляется через процедуру обмена «запрос/ответ», в которой вы делаете запрос на определенные видеозаписи и затем получаете информацию об этих видеозаписях в виде ответа сервера YouTube. Ваше приложение ответственно как за составление запроса в формате, понятном серверу YouTube, так и за обработку результата, которая включает выборку в ответе данных для получения определенных необходимых вам видео (заголовок, ярлык видео, гиперссылка и т. д.).

Далее следуют этапы, которые необходимо пройти при извлечении видеозаписей с сервера YouTube и выводе их на веб-страницу.

Распространение видеозаписей с сервера YouTube включает составление запросов и обработку ответов.

1. **Составление запроса на видеозаписи YouTube.** ← Запрос часто в форме URL.
2. **Отправление запроса на YouTube.**
3. **Получение от YouTube ответа, содержащего информацию о видеозаписях.** ← Сервер YouTube использует XML в ответах на запросы видеозаписей.
4. **Обработка данных ответа и форматирование их в виде HTML-кода.**



Создание видеозапроса к серверу YouTube

Получение видеозаписей с сервера YouTube и добавление их к вашей веб-странице начинается с запроса. Сервер YouTube ожидает, что запрос на видеозаписи будет сделан как ***запрос REST***, являющийся специальным URL, который указывает на определенный ресурс, например, содержащий видеоданные. Вы составляете URL с указанием нужной вам видеозаписи, и сервер YouTube возвращает вам информацию об этой видеозаписи в виде XML-документа.

Детали URL для запроса к серверу YouTube определяются конкретными видеозаписями, к которым вы хотите получить доступ. Например, вы можете запросить любимые видеозаписи определенного пользователя. В случае Оуэна наилучшим способом было бы, скорее всего, осуществить поиск по ключевым словам среди всех видеозаписей, доступных на сервере YouTube. URL, необходимые для всех этих типов запросов REST, будут немного отличаться друг от друга, но базовый URL всегда будет начинаться так:

```
http://gdata.youtube.com/feeds/api/
```

Этот базовый URL используется для составления всех запросов REST к серверу YouTube.

Не бывает глупых вопросов

В: От каких слов происходит аббревиатура REST?

О: REpresentational State Transfer (передача состояния представления). Это, определенно, одна из тех аббревиатур, которые звучат малопонятно и очень технически. Важная для рассматриваемого случая особенность архитектуры REST заключается в том, что веб-ресурс доступен через уникальный URL. Это означает, что вы можете получить доступ к данным, поддерживающим эту архитектуру, просто через этот URL. В случае получения видеозаписей с сервера YouTube это означает, что вы можете делать все ваши запросы через URL, который содержит поисковые критерии.

Запрос видеозаписей по имени пользователя

Запрос любимых видеозаписей определенного пользователя включает добавление к базовому URL имени этого пользователя.

Имя пользователя, зарегистрированного на сервере YouTube предоставляет доступ к его любимым видеозаписям.

```
http://gdata.youtube.com/feeds/api/users/username/favorites
```

Для того чтобы запросить любимые видео пользователя `elmerpriestley`, используйте следующий URL:

```
http://gdata.youtube.com/feeds/api/users/elmerpriestley/favorites
```

Результатом этого запроса REST будут любимые видеозаписи пользователя elmerpriestley, зарегистрированного на сервере YouTube.

Запрос видеозаписей с ключевыми словами для поиска

Значительно более эффективным и часто более полезным запросом к серверу YouTube является запрос на проведение поиска видеозаписей по ключевым словам независимо от их принадлежности пользователям. Вы можете использовать несколько ключевых слов, отделяя их друг от друга косой чертой. Ключевые слова добавляются в конец URL.

```
http://gdata.youtube.com/feeds/api/videos/-/keyword1/keyword2/...
```

Множество ключевых слов, отделенных друг от друга косой чертой, может быть использовано в запросе.

URL начинается так же, как и в случае запроса с использованием имени пользователя, только теперь вместо слова users используется слово videos.

Не забудьте про косые черты и дефис!

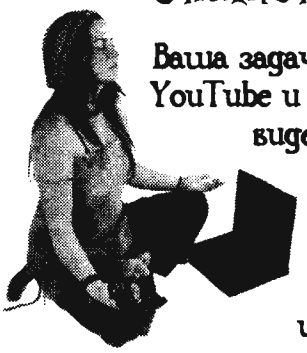
Для того чтобы запросить видеозаписи по ключевым словам `elvis` и `impersonator`, используйте следующий URL:

```
http://gdata.youtube.com/feeds/api/videos/-/elvis/impersonator
```

Здесь используются ключевые слова elvis и impersonator для поиска видеозаписей.

Ключевые слова не чувствительны к регистру клавиатуры, поэтому elvis, Elvis и eLvis дадут один и тот же результат.

Станьте REST-запросом к серверу YouTube



Ваша задача — проникнуть в мысли сервера YouTube и сыграть роль REST-запроса на видеоинформацию. Используя магниты, составьте REST-запрос на следующую видеоинформацию, имеющуюся на сервере YouTube. Проверьте работу этих запросов из своего браузера.

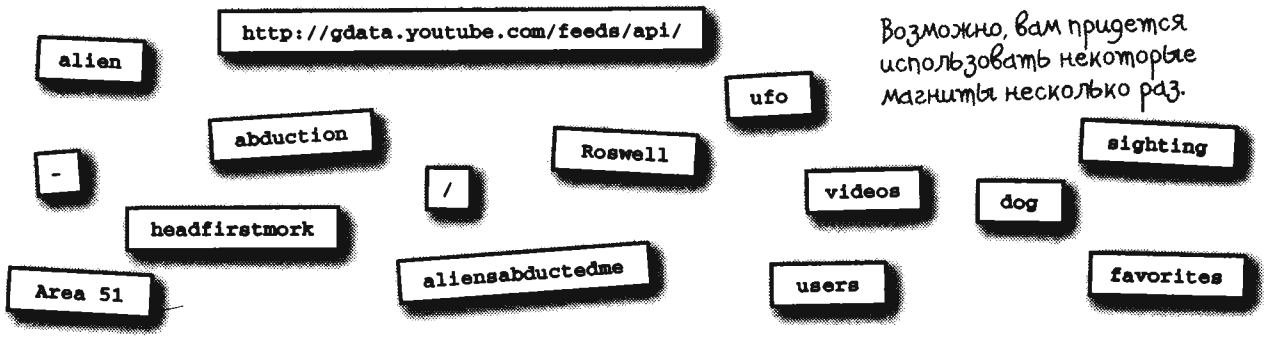
Все видеозаписи, соответствующие ключевому слову Rosewell:

Все видеозаписи, соответствующие ключевым словам alien и abduction:

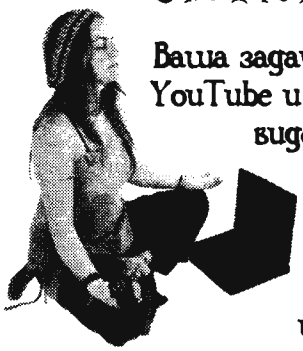
Все видеозаписи пользователя headfirstmork, отмеченные как favorites:

Все видеозаписи, соответствующие ключевым словам ufo, sighting и dog:

Все видеозаписи пользователя aliensabductedme, отмеченные как favorites:



Станьте REST-запросом к серверу YouTube



Ваша задача — проникнуть в мысли сервера YouTube и сыграть роль REST-запроса на видеоинформацию. Используя магниты, составьте REST-запрос на следующую видеоинформацию, имеющуюся на сервере YouTube. Проверьте работу этих запросов из своего браузера.

Возможно, вам придется использовать некоторые магниты несколько раз.

Один и тот же базовый URL используется для всех REST-запросов.

В конце запроса добавляется ключевое слово.

Все видеозаписи, соответствующие ключевому слову Rosewell:

```
http://gdata.youtube.com/feeds/api/ videos / - / Roswell
```

В конце запроса добавляются ключевые слова, отделенные друг от друга косой чертой.

Все видеозаписи, соответствующие ключевым словам alien и abduction:

```
http://gdata.youtube.com/feeds/api/ videos / - / alien / abduction
```

Все видеозаписи пользователя headfirstmork, отмеченные как favorites:

```
http://gdata.youtube.com/feeds/api/ users / headfirstmork / favorites
```

URL для запроса любимых видеозаписей пользователя требует слова users (пользователи) вместо слова videos (видеозаписи).

Все видеозаписи, соответствующие ключевым словам ufo, sighting и dog:

```
http://gdata.youtube.com/feeds/api/ videos / - / ufo / sighting / dog
```

В конце запроса добавляется слово favorites (любимые).

Все видеозаписи пользователя aliensabductedme, отмеченные как favorites:

```
http://gdata.youtube.com/feeds/api/ users / aliensabductedme / favorites
```

Area 51

Этот магнит не используется, он добавлен для конспирации!

Это имя пользователя, к чьим любимым видеозаписям мы хотим получить доступ.

не бывает глупых вопросов

В: Чем отличается REST-запрос от, скажем, GET-запроса?

О: Ничем. Всякий раз, когда вы делаете GET-запрос, например при запросе веб-страницы, вы, по сути, делаете REST-запрос. В рамках архитектуры REST вы можете рассматривать любую веб-страницу как REST-ресурс в том смысле, что доступ к ней может быть осуществлен через ее URL, и GET — это всего лишь REST-метод, используемый для получения доступа к этому ресурсу. Где REST становится еще интереснее, так это при составлении запросов, подобных тем запросам на видеозаписи, которые мы делали к серверу YouTube. В этих случаях, хотя мы и имеем дело с REST-запросами, они не являются запросами статических веб-страниц, а в конечном счете осуществляются к базе данных YouTube.

В: Имеет ли значение порядок следования ключевых слов в REST-запросе к серверу YouTube?

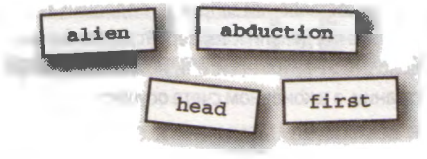
О: Да. Первым ключевым словам назначается более высокий приоритет по сравнению с последующими словами, поэтому не забывайте располагать их в порядке уменьшения важности.

В: В случае если критериям поиска соответствует множество видеозаписей, как сервер YouTube определяет, какую из них необходимо включить в результат поиска?

О: Сервер YouTube при поиске по ключевым словам возвращает видеозаписи, основываясь на их релевантности. Это означает, что вы получите те видеозаписи, которые наиболее соответствуют критериям поиска, независимо от того, когда они поступили на сервер.

Оуэн готов создать REST-запрос

Так как задача Оуэна заключается в поиске на YouTube видеозаписей, которые могут содержать информацию о Фэнге, поиск по ключевым словам имеет больший смысл при выборе типа REST-запроса к серверу YouTube. Можно составить множество различных комбинаций ключевых слов для поиска видеозаписей с Фэнгом, но одна из них поможет нацелиться на те, которые имеют отношение непосредственно к Фэнгу:



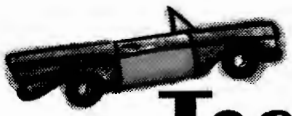

```
http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first
```

Последние два ключевых слова обеспечивают нахождение видеозаписей, имеющих отношение к Оуэну и Фэнгу!

Вряд ли вы будете указывать наименования серий книг в качестве ключевых слов в случае обычного запроса на YouTube. Но в данном случае это должно помочь. Давайте просто назовем совпадением то, что большинство видеозаписей о похищении космическими пришельцами сделаны поклонниками серии книг Head First. С REST-запросом на руках Оуэн может вычеркнуть первую строку перечня этапов, которые необходимо пройти для извлечения видеозаписей с сервера YouTube и вывода их на веб-страницу.

Первый этап выполнен благодаря составлению REST-запроса к серверу YouTube.

1. ~~Составление запроса на видеозаписи YouTube.~~
2. **Отправление запроса на YouTube.**
3. **Получение от YouTube ответа, содержащего информацию о видеозаписях.**
4. **Обработка данных ответа и форматирование их в виде HTML-кода.**



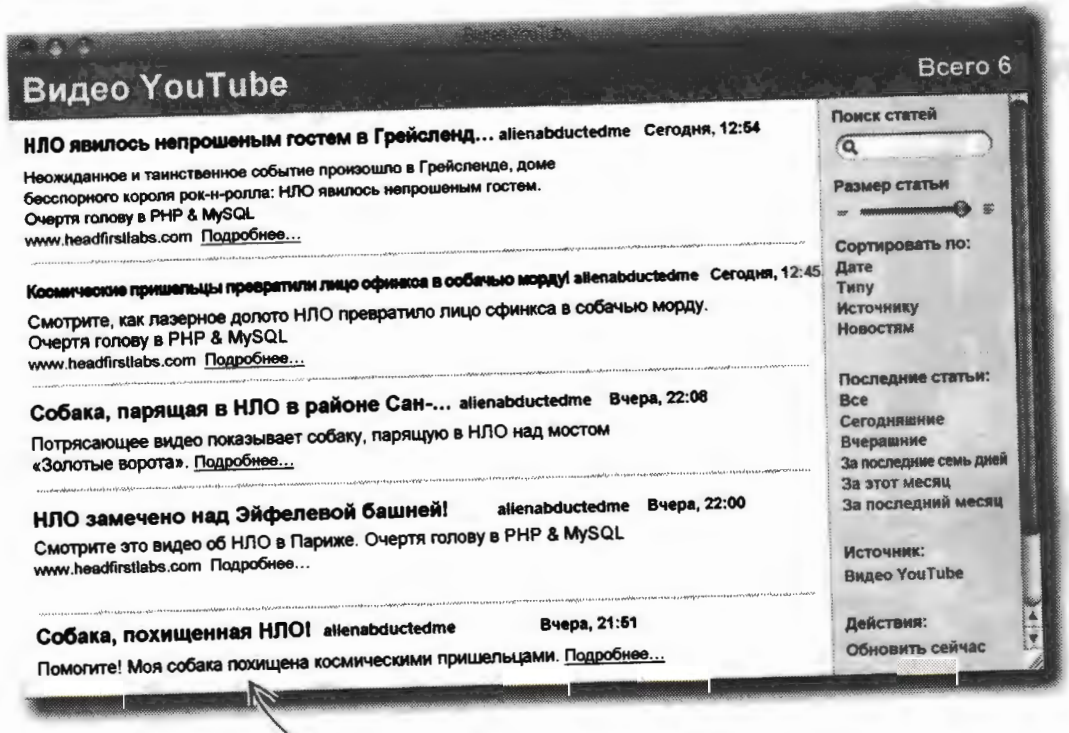
Тест-драйв

Проверьте REST-запрос Оуэна к серверу YouTube.

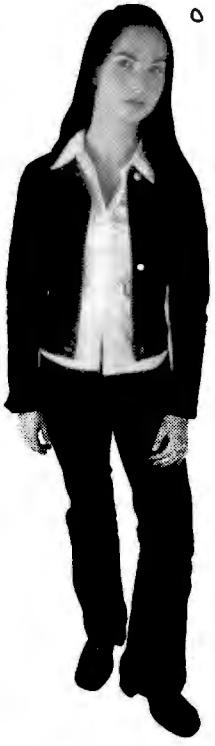
Введите URL, содержащий REST-запрос Оуэна к серверу YouTube, в адресную строку вашего браузера:

```
http://gdata.youtube.com/feds/api/videos/-/alien/abduction/head/first
```

Что вы видите? Попробуйте посмотреть исходный текст страницы для того, чтобы увидеть код, который передал браузеру сервер YouTube.



Браузер выводит как новости данные, переданные ему сервером YouTube в формате XML, которые в рассматриваемом случае являются видеозаписями.



Запрашивание видеозаписей с сервера YouTube путем ввода URL в адресную строку браузера работает просто безупречно, но какое отношение это имеет к PHP? Разве мы не можем получить доступ к этим видеозаписям из сценария?

SimpleXML, расширение PHP, предлагает функцию simplexml_load_file(). Это расширение добавлено к PHP начиная с версии 5.0, поэтому более ранние версии PHP не имеют встроенной поддержки обработки XML-документов.

Нам необходима функция PHP, которая позволит передать REST-запрос и получить ответ.

Встроенная в PHP функция `simplexml_load_file()` дает нам возможность передать на сервер REST-запрос и получить ответ в виде XML-документа, как мы делали это с YouTube. Функция фактически загружает XML-документ как PHP-объект, из которого затем извлекается вся необходимая информация. Так как же это влияет на запрос Оуэна к серверу YouTube? Посмотрите на этот код, благодаря которому создается константа, содержащая URL-запроса к серверу YouTube, и затем делается REST-запрос с использованием функции `simplexml_load_file()`:

```
define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first
$xml = [redacted](YOUTUBE_URL);
```

Хотя и не совсем обязательно, но очень полезно сохранять статические URL в константах, чтобы вы знали, куда вносить изменения при необходимости.

1. ~~Составление запроса на видеозаписи YouTube.~~
2. ~~Отправление запроса на YouTube.~~
3. ~~Получение от YouTube ответа, содержащего информацию о видеозаписях.~~
4. **Обработка данных ответа и форматирование их в виде HTML-кода.**

И эти два этапа сделаны!

Не переживайте, если вы не понимаете, что такое объект, особенно в контексте PHP.

Объект PHP — это особый тип данных. В объекте данные и функции по их обработке объединены в одну общую структуру. Все, что вам пока необходимо знать, — это то, что обработку XML-документов в PHP значительно удобнее делать с использованием объектов. В ближайшее время вы узнаете, как это делается на практике.

А вот и есть! XML-код, возвращенный сервером YouTube, никакой свалкой данных не является, как это может показаться неискушенному взгляду... Просто нужно знать, куда смотреть.

YouTube разговаривает на XML

Ответ, который YouTube отправляет по вашему запросу, — это совсем не DVD в яркой коробке, приносимый почтальоном ко входной двери вашей квартиры. Это не сама видеозапись, а XML-документ, содержащий детальную информацию о ней.

Сервер YouTube отвечает на запрос видео XML-документом, который описывает это видео.

Хотя в этом XML-документе достаточно много информации, одна вещь достойна дополнительного внимания. Она заключается в том, что все данные о конкретной видеозаписи собраны между тегами <entry> и </entry>.

```
<?xml version='1.0' encoding='UTF-8'?>
<feed xmlns='http://www.w3.org/2005/Atom'
 xmlns:openSearch='http://a9.com/-/spec/opensearchrss/1.0/'
 xmlns:gml='http://www.opengis.net/gml'
 xmlns:georss='http://www.georss.org/georss'
 xmlns:media='http://search.yahoo.com/mrss/'
 xmlns:batch='http://schemas.google.com/gdata/batch'
 xmlns:yt='http://gdata.youtube.com/schemas/2007'
 xmlns:gd='http://schemas.google.com/g/2005'>
 <id>http://gdata.youtube.com/feeds/api/users/aliensabductedme/favorites</id>
 <updated>2008-07-25T03:22:37.001Z</updated>
 <category scheme='http://schemas.google.com/g/2005#kind'
  term='http://gdata.youtube.com/schemas/2007#video'/>
 <title type='text'>Любимые видеозаписи пользователя alienabductedme</title>
 ...
 <entry>
  <id>http://gdata.youtube.com/feeds/api/videos/_6UibqfOvtA</id>
  <published>2006-06-20T07:49:05.000-07:00</published>
  ...
  <media:group>
   <media:title type='plain'>НЛО замечено в Йосмитском национальном парке, в районе зоны 51</media:title>
   <media:description type='plain'>Я путешествовал по Йосмитскому национальному парку возле зоны 51 в 2005 г.
Йосмитский национальный парк расположен очень близко к границе между Калифорнией и Невадой и очень близко к зоне
51...</media:description>
   <media:keywords>51,космический, пришелец, космические, пришельцы, зона, Калифорния, Невада, наблюдение,
наблюдения, НЛО</media:keywords>
   <yt:duration seconds='50'/>
   <media:category label='Travel &amp;amp; Events'
    scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
   <media:content url='http://www.youtube.com/v/_6UibqfOvtA' type='application/x-shockwave-flash'
    medium='video' isDefault='true' expression='full' duration='50' yt:format='5'/>
   <media:content url='rtsp://rtsp2.youtube.com/CheLEMy7JwIaEQnQvvSnbiKl_xMYDSANFEgGDA==/0/0/0/video.3gp'
    type='video/3gpp' medium='video' expression='full' duration='50' yt:format='1'/>
   <media:content url='rtsp://rtsp2.youtube.com/CheLEMy7JwIaEQnQvvSnbiKl_xMYESARFEgGDA==/0/0/0/video.3gp'
    type='video/3gpp' medium='video' expression='full' duration='50' yt:format='6'/>
   <media:player url='http://www.youtube.com/watch?v=_6UibqfOvtA'/>
   <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/2.jpg' height='97' width='130'
    time='00:00:25'/>
   <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/1.jpg' height='97' width='130'
    time='00:00:12.500'/>
   <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/3.jpg' height='97' width='130'
    time='00:00:37.500'/>
   <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/0.jpg' height='240' width='320'
    time='00:00:25'/>
  </media:group>
  <yt:statistics viewCount='2478159' favoriteCount='1897'/>
  <gd:rating min='1' max='5' numRaters='1602' average='4.11'/>
  <gd:comments>
   <gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6UibqfOvtA/comments'
    countHint='4426'/>
  </gd:comments>
 </entry>
 <entry>
  <id>http://gdata.youtube.com/feeds/api/videos/XpNd-Dg6_zQ</id>
  <published>2006-11-19T16:44:43.000-08:00</published>
  ...
 </entry>
</feed>
```

С тега <entry> начинается описание другой видеозаписи внутри XML-документа.

Время поработать

Просмотрите выделенный код ответа сервера YouTube на предыдущей странице и ответьте на следующие вопросы. Возможно, вы знаете больше, чем это может показаться на первый взгляд, об XML-формате, в котором сервер YouTube передает информацию о видеозаписях.

1. Какой заголовок у видеозаписи? .....

2. Какие три ключевых слова связаны с этой видеозаписью? .....

3. Какова длительность воспроизведения этой видеозаписи в секундах? .....

4. К какой видеокатегории YouTube принадлежит эта видеозапись? .....

5. Сколько раз просматривалась эта видеозапись? .....

6. Каково среднее значение рейтинга, данное пользователями этой видеозаписи? .....

Решение задачи

Просмотрите выделенный код ответа сервера YouTube на предыдущей странице и ответьте на следующие вопросы. Возможно, вы знаете больше, чем это может показаться на первый взгляд, об XML-формате, в котором сервер YouTube передает информацию о видеозаписях.

```
<media:title type='plain'>НЛО замечено в Йосмитском национальном парке, в районе зоны 51</media:title>
```

1. Какой заголовок у видеозаписи? НЛО замечено в Йосмитском национальном парке, в районе зоны 51

```
<media:keywords>51, космический, пришелец, космические, пришельцы, зона, Калифорния,
Невада наблюдение, наблюдения. НЛО</media:keywords>
```

2. Какие три ключевых слова связаны с этой видеозаписью? 51, пришельцы, Невада

3. Какова длительность воспроизведения этой видеозаписи в секундах? 50

```
<yt:duration seconds='50'/>
```

Некоторые символы в XML записываются в виде специальных кодов. Например, код & представляет символ амперсанда (&).

```
<media:category label='Travel &amp;amp; Events'
  scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
```

4. К какой видеокатегории YouTube принадлежит эта видеозапись? Travel & Events

```
<yt:statistics viewCount='2478159' favoriteCount='1897'/>
```

Ого, сколько просмотров!.. Почти два с половиной миллиона!

5. Сколько раз просматривалась эта видеозапись? 2478159

```
<gd:rating min='1' max='5' numRaters='1602' average='4.17'/>
```

6. Каково среднее значение рейтинга, данное пользователями этой видеозаписи? 4.17

Гм, мне не совсем понятны эти XML-теги с именами, состоящими из двух слов, разделенных двоеточием. Это какой-то способ упорядочивания тегов? И что это за странный код & в категории видео?

Необычные XML-коды — это пространства имен и символьные подстановки. Они используются для организации тегов и кодирования специальных символов.

Когда вы сталкиваетесь с XML-тегами, имена которых состоят из двух слов, разделенных двоеточием, вы имеете дело с ***пространствами имен***, которые используются для того, чтобы объединить несколько тегов в логическую группу. Цель пространств имен заключается в том, чтобы предотвратить конфликты имен в случаях использования слишком большого количества различных тегов в одном XML-документе. В качестве примера давайте рассмотрим следующие два XML-тега:

```
<title type='text'>Любимые видеозаписи пользователя alienabductedme</title>
```

```
<media:title type='plain'>НЛО замечено в Йосмитском национальном парке, в районе зоны 51</media:title>
```

Пространство имен — это именованная группа XML-тегов, в то время как символьная подстановка используется для кодирования специальных (служебных) символов в XML-документе.

Без использования пространства имен `media` во втором теге `<title>` было бы невозможно отделить эти два тега друг от друга при их использовании в одном XML-коде. Таким образом, вы можете рассматривать пространства имен как фамилии тегов. Они помогают предотвращать конфликты одноименных тегов путем присваивания им «фамилий». В XML-документе, полученном с сервера YouTube, используется несколько пространств имен, что можно рассматривать так, будто в нем используется несколько XML-«языков» одновременно. Использование пространств имен позволяет отделить их друг от друга.

Для обеспечения уникальности пространства имен XML всегда ассоциируются с URL. Например, пространство имен `media`, используемое сервером YouTube, определено в теге `<feed>` следующим образом:

```
xmlns:media='http://search.yahoo.com/mrss/'
```

Этот URL не является веб-страницей. Это уникальный идентификатор для данного пространства имен.

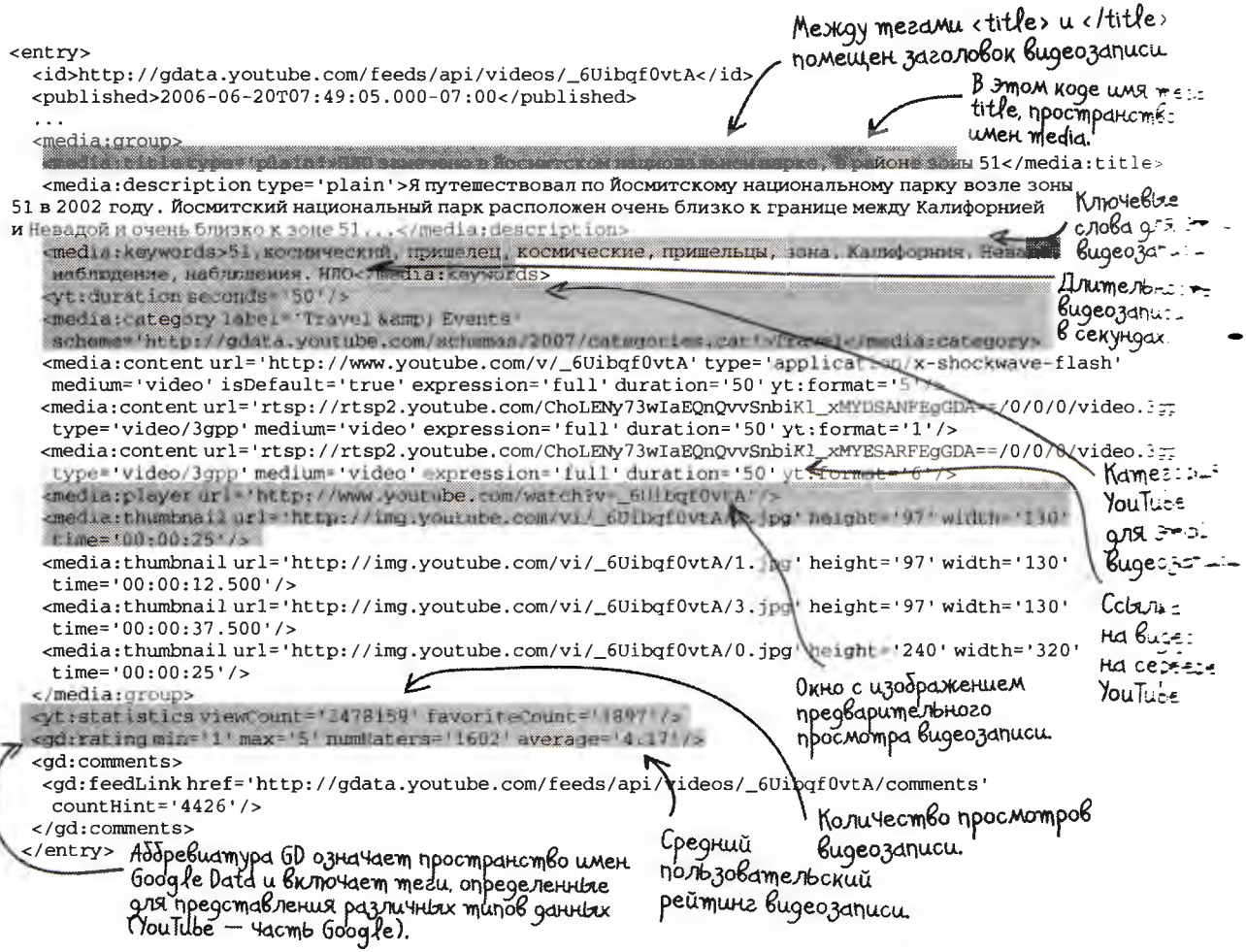
Может показаться странным тот факт, что YouTube использует пространство имен Yahoo!. Это означает, что YouTube частично полагается на формат данных XML, созданный Yahoo!.

Другим непонятным кодом в XML-документе, полученном с сервера YouTube, является код &, который используется для представления символа амперсанда (&). Это используемая в XML **символьная подстановка** — способ представления в XML-документе символов, имеющих специальное (служебное) значение в XML-коде, таких как &, <, > и других. Ниже приведены пять предопределенных символьных подстановок XML и символов, которые они представляют; вы будете сталкиваться с ними по мере дальнейшего углубления в XML-код:

`&` = `&` `<` = `<` `>` = `>` `&quot;` = `"` `&apos;` = `'`

Разбор XML-ответа сервера YouTube

После того как вы поняли структуру ответа, переданного сервером YouTube, извлечение необходимых вам видеоданных становится достаточно несложным процессом. Дополнительно к пониманию того, как теги и атрибуты описывают видеоданные, очень важно также знать, как эти теги соотносятся друг с другом. Если вы помните из более ранних разделов этой главы, где анализировались RSS-каналы, XML-документ можно рассматривать как иерархию элементов. То же самое справедливо для XML-данных, переданных сервером YouTube.




```
<entry>
  <id>http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA</id>
  <published>2006-06-20T07:49:05.000-07:00</published>
  ...
  <media:group>
    <media:title type='plain'>Йосмитский национальный парк, район зоны 51</media:title>
    <media:description type='plain'>Я путешествовал по Йосмитскому национальному парку возле зоны
51 в 2002 году. Йосмитский национальный парк расположен очень близко к границе между Калифорнией
и Невадой и очень близко к зоне 51...</media:description>
    <media:keywords>51, космический, пришелец, космические, пришельцы, зона, Калифорния, Невада,
     наблюдение, наблюдения, НЛО</media:keywords>
    <yt:duration seconds='50'/>
    <media:category label='Travel &amp;amp; Events'
     scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
    <media:content url='http://www.youtube.com/v/_6Uibqf0vtA' type='application/x-shockwave-flash'
     medium='video' isDefault='true' expression='full' duration='50' yt:format='5'/>
    <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbiKl_xMYDSANFEgGDA==/0/0/0/video.3gp'
     type='video/3gpp' medium='video' expression='full' duration='50' yt:format='1'/>
    <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbiKl_xMYESARFEgGDA==/0/0/0/video.3gp'
     type='video/3gpp' medium='video' expression='full' duration='50' yt:format='6'/>
    <media:player url='http://www.youtube.com/watch?v=_6Uibqf0vtA'/>
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/2.jpg' height='97' width='130'
     time='00:00:25'/>
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/1.jpg' height='97' width='130'
     time='00:00:12.500'/>
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/3.jpg' height='97' width='130'
     time='00:00:37.500'/>
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/0.jpg' height='240' width='320'
     time='00:00:25'/>
  </media:group>
  <yt:statistics viewCount='2478159' favoriteCount='1897'/>
  <gd:rating min='1' max='5' numRaters='1602' average='4.47'/>
  <gd:comments>
   <gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA/comments'
    countHint='4426'/>
  </gd:comments>
</entry>
```

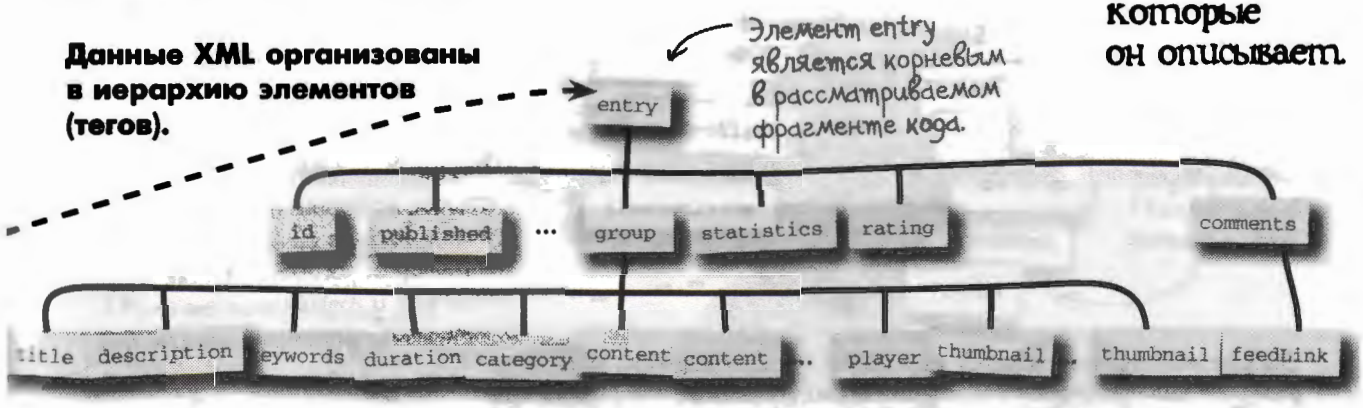
Осмысление принципа использования пространства имен имеет ключевое значение для понимания того, как видеоданные описываются XML-кодом. Пространство имен `media` используется при описании большинства видеоданных, в то время как пространство имен `yt` применяется только с тегом `<statistic>`. Наконец, комментарии записываются между тегами `<comments>` и `</comments>`, которые принадлежат пространству имен `gd`. Все эти пространства имен будут иметь большое значение когда вы начнете писать PHP-код по поиску определенных тегов и данных, которые они описывают.

Визуализация видеоданных XML

Ранее в этой главе, во время обсуждения RSS, было сказано, что XML-документ может быть представлен визуально в виде иерархического дерева элементов (тегов), которые связаны отношениями родители — потомки. Значение этих отношений значительно возрастает, как только вы начинаете обрабатывать XML-код с целью получения доступа к данным, которые он описывает. Очень важно учиться быстро представлять себе все эти связи между элементами, глядя на XML-документ. Не забывайте о том, что каждый элемент, находящийся внутри другого элемента, — это потомок, а элемент, который включает в себя другой элемент, является по отношению к нему родителем. Представляя XML-код, приведенный на предыдущей странице, в виде иерархического дерева, получаем следующую картину:

Элементом мы называем здесь абстрактное представление XML-тега и данных, которые он описывает.

Данные XML организованы в иерархию элементов (тегов).



Значение этой иерархии заключается в том, что вы можете проследить весь путь от каждого элемента до любого другого в этом дереве, начиная от его корня (наверху). Так, например, если вы хотите получить заголовок видео, то можете проследить путь к нему следующим образом:

Не бывает глупых вопросов

В: Почему я вообще должен беспокоиться о пространствах имен?

О: Потому что XML-код, составленный другими, часто включает пространства имен, которые влияют на то, как вы будете получать доступ к XML-элементам из программы. Как вы, наверное, уже догадались, пространство имен, связанное с определенным элементом, непосредственно влияет на то, как вы получите доступ к этому элементу из вашего кода по обработке XML-документа. Поэтому при извлечении данных из элемента пространство имен, с которым он связан, обязательно должно быть учтено в коде.

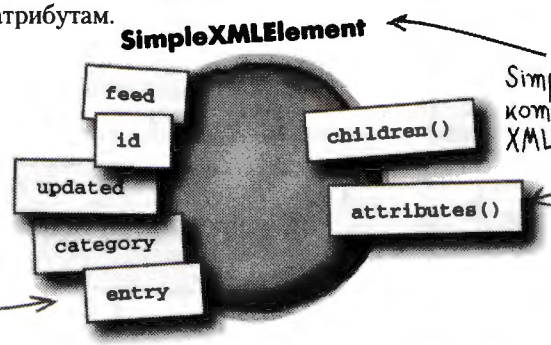
В: Как я узнаю, что тег связан с пространством имен?

О: Хотя можно использовать пространство имен по умолчанию, которое не указывается в коде тега в явном виде, в большинстве случаев вы можете увидеть имя пространства имен, включенное непосредственно в имя тега. Последнее в этом случае кодируется как `<media:title>`, а не просто `<title>`. Слово, расположенное слева от двоеточия, — это всегда имя пространства имен.

Доступ к данным XML с помощью объектов

Объект PHP — это особый тип данных, в котором и данные, и функции по обработке этих данных объединены в одну общую структуру.

Существует множество различных способов работы с данными XML в PHP. И один из наиболее удобных — это метод с использованием объектов. **Объект** — это особый тип данных PHP, в котором и данные, и функции по обработке этих данных объединены в одну общую структуру. Но какое отношение это имеет к XML? Вся иерархия элементов XML-документа содержится в одной переменной — объекте. Вы можете использовать этот объект для получения доступа к конкретным элементам. В объекте также имеются так называемые **методы**, которые по своей сути являются функциями, включенными в состав объекта и позволяющими манипулировать данными этого объекта. Что касается методов объекта, содержащего XML-документ, они позволяют вам получать доступ ко всем элементам этого документа и их атрибутам.



Каждый элемент может быть доступен как атрибут (переменная) XML-объекта.

SimpleXMLElement — тип объекта PHP, который используется для сохранения XML-данных и манипулирования ими.

Объект SimpleXMLElement имеет методы, которые позволяют вам узнать больше об элементах, например об их элементах-потомках и атрибутах этих элементов-потомков.

Вы уже видели, как создается этот XML-объект для поиска на сервере YouTube видеозаписей о похищении космическими пришельцами для Оуэна.

Не забывайте, эта функция доступна только в версиях PHP 5.0 и выше.

```
define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first'

$xml = simplexml_load_file(YOUTUBE_URL);
```

Эта функция создает PHP-объект типа SimpleXMLElement, содержащий все XML-данные ответа сервера YouTube.

В результате выполнения этого кода будет создана переменная с именем `$xml`, содержащая все видеоданные, полученные с сервера YouTube в формате XML и упакованные в объект PHP. Доступ к этим данным осуществляется через **атрибуты** объекта, которые являются конкретными переменными, сохраненными в этом объекте. Каждый атрибут объекта соответствует элементу XML-документа. Взгляните на следующий пример, который демонстрирует доступ ко всем элементам записи документа:

```
$entries = $xml->entry;
```

Указывая имя элемента (entry), вы можете извлечь все имеющиеся элементы.

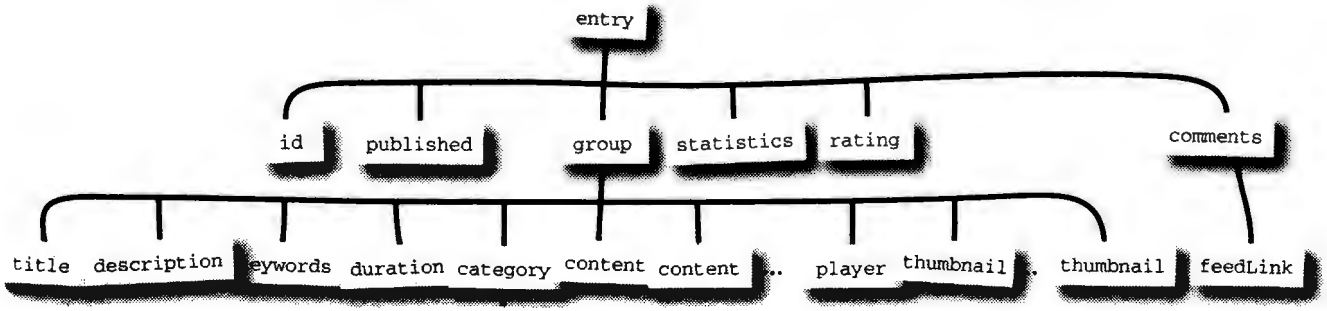
Оператор -> дает вам возможность получить доступ к атрибуту объекта.

Все видеозаписи сохранены как элементы массива $entries.

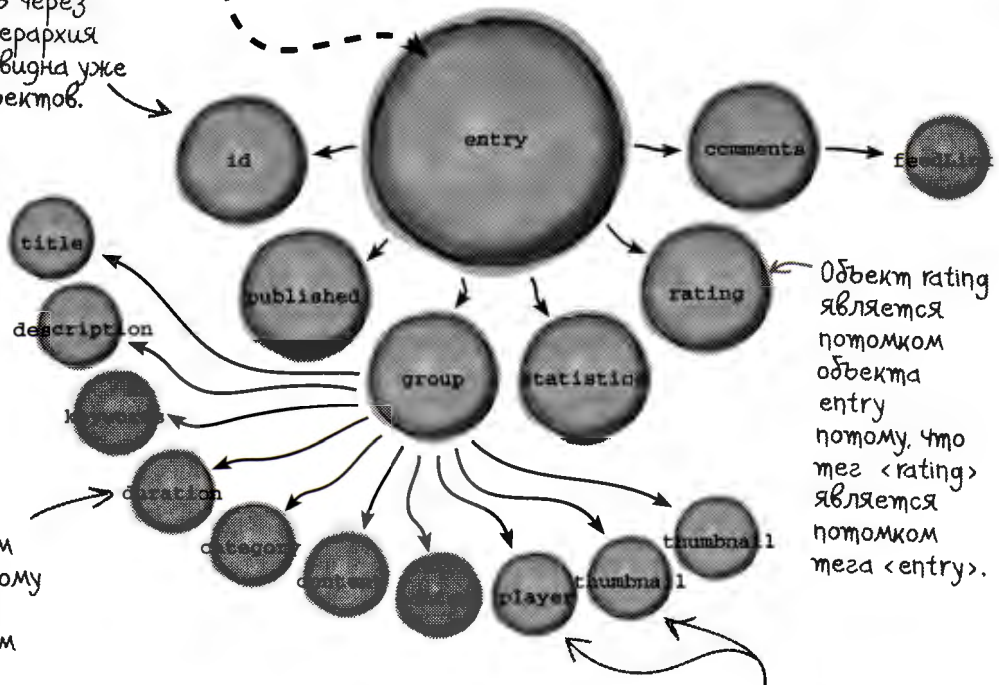
В результате выполнения этого кода вы получаете доступ ко всем элементам `entry` в XML-документе, используя атрибут объекта. Так как в документе имеется множество элементов `entry`, в переменной `$entries` будет сохранен массив объектов, которые вы можете использовать для того, чтобы получить доступ к индивидуальным данным видеозаписей. А раз мы имеем дело с массивом, доступ к каждому из тегов `<entry>` может быть получен через его индекс. Например, первый тег `<entry>` соответствует первому элементу массива, второй тег — второму элементу и т. д.

От элементов XML — к объектам PHP

Когда мы говорим о данных XML и объектах PHP, в действительности мы подразумеваем иерархии объектов. Помните ту картинку, изображающую иерархическое дерево элементов XML-документа? Так вот, в виде точно такого же дерева может быть представлена иерархия объектов PHP. Посмотрите:



Если посмотреть через XML-объектив, иерархия XML-элементов видна уже как иерархия объектов.



Объект rating является потомком объекта entry потому, что тег <rating> является потомком тега <entry>.

Объект duration является потомком объекта group, потому что тег <duration> является потомком тега <group>.

Эта иерархия элементов/объектов создает основу для понимания того, как получать доступ к данным XML-документа в PHP. Если иметь в виду связи между индивидуальными данными XML-документа, становится понятным, как следует писать код для получения к ним доступа. Зная структуру иерархии элементов/объектов, можно получить доступ к данным на любом ее уровне.

Большинство интересных данных о видеозаписях YouTube содержится в объектах-потомках объекта group.

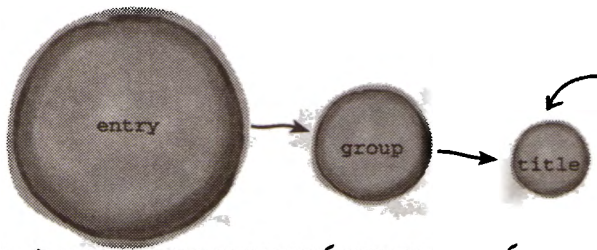
Извлечение данных XML из объекта

Вернемся к проблеме Оуэна. Нашей целью является извлечение видеоинформации из XML-документа, возвращенного сервером YouTube. Мы уже знаем, как преобразовать эти данные в иерархию объектов PHP, используя функцию `simplexml_load_file()`, но большинство интересующих нас данных находится на более глубоких уровнях этой иерархии. Как нам перемещаться по этой иерархии объектов? Ответ заключается в использовании оператора `->`, который позволяет получить доступ к членам объекта: атрибуту (переменной, причем ее значением может быть в том числе и объект-потомок) или методу (функции). В нашем случае оператор `->` позволяет получить доступ к каждому из объектов-потомков. Таким образом, показанный ниже код позволяет получить доступ к заголовку видеозаписи, сохраненному в переменной `$entry`:

```
echo $entry->group->title;
```

Операторы -> используются здесь для того, чтобы, спустившись по иерархическому дереву от объектов-родителей к их объектам-потомкам, получить доступ к объекту title.

Этот код полностью полагается на связи между объектами `title`, `group` и `entry`, которые формируют отношения типа родитель — потомок в указанном порядке.



Объект title является потомком объекта group, который, в свою очередь, является потомком объекта entry.

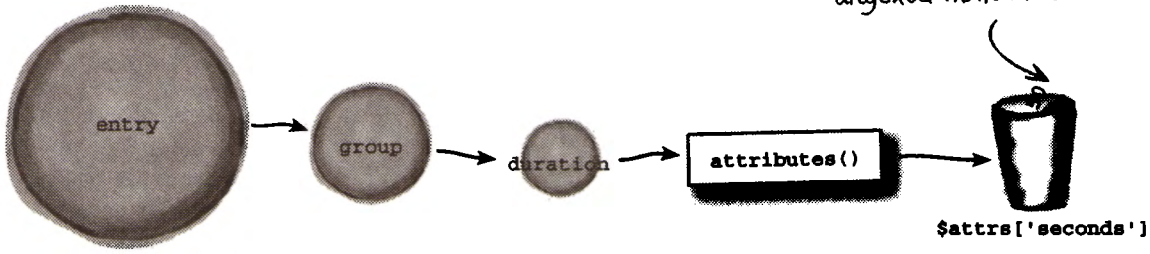
Оператор `->` возвращает ссылку на объект-потомок объекта, расположенного слева от этого оператора. А так как объект `title` является потомком объекта `group`, который, в свою очередь, является потомком объекта `entry`, в результате выполнения вышеприведенного кода для команды `echo` возвращается ссылка на объект `title`. Не забывайте, что оператор `->` может быть также использован для получения доступа к атрибутам (переменным) и методам (функциям) объекта. Одним из очень полезных методов является метод `attributes()`, который возвращает значение атрибутов объекта (XML-элементов).

```
$attrs = $entry->group->duration->attributes();

echo $attrs['seconds'];
```

Метод attributes() возвращает массив атрибутов объекта (XML-элемента) duration.

В результате выполнения этого кода мы спускаемся по иерархическому дереву до объекта `duration` и извлекаем все его атрибуты в виде массива, который сохраняем в переменной `$attrs`. Затем мы извлекаем значение атрибута `seconds`.

Значение определенного атрибута может быть получено путем указания имени атрибута как индекса (ключа) в массиве.



Только с учетом пространства имен!

Пространства имен несколько усложняют доступ к элементам внутри XML-документа.

Имеется небольшая проблема с кодом, приведенным на предыдущей странице, который получает доступ к XML-данным, используя объекты, в то время как эти данные зависят от пространств имен. Если вы помните, пространства имен выступают в роли «фамилий» для тегов, организуя последние в логически значимые группы. В XML-документе, полученном с сервера YouTube, тег `<duration>` фактически записан как `<yt:duration>`, а тег заголовка видеозаписи — как `<media:title>`, а не просто `<title>`. Когда XML-элемент ассоциирован с пространством имен, в PHP-коде вы не можете просто ссылаться на имя тега. Вначале вы должны определить пространство имен вызовом метода родительского объекта `children()`.

```
$media = $entry->[redacted]('http://search.yahoo.com/mrss/');
```

Метод children() возвращает массив, содержащий все элементы-потомки, ассоциированные с пространством имен, переданным этому методу в качестве аргумента при вызове.

В результате вызова этого метода возвращаются все объекты-потомки элемента `entry`, ассоциированные с пространством имен определенным аргументом, переданным этому методу при вызове (`http://search.yahoo.com/mrss/`). Но это не пространство имен, это URL, который определяет это пространство имен. Он расположен между тегами `<feed>` и `</feed>` в начале XML-документа. Здесь вы найдете все пространства имен, используемые в этом документе.

```
<feed xmlns='http://www.w3.org/2005/Atom'
  xmlns:openSearch='http://a9.com/-/spec/opensearchrss/1.0/'
  xmlns:gml='http://www.opengis.net/gml'
  xmlns:georss='http://www.georss.org/georss'
  [redacted]
  xmlns:batch='http://schemas.google.com/gdata/batch'
  [redacted]
  xmlns:gd='http://schemas.google.com/g/2005'>
```

Все теги с префиксом `<media:` принадлежат этому пространству имен.

Это пространство имен ассоциируется с тегами, начинающимися с `<yt:`.

Этот код показывает, как каждое пространство имен ассоциируется с URL. Точнее, он показывает, как определены интересующие нас в данном случае пространства имен `media` и `yt` для использования в документе. Это все, что вам необходимо знать для того, чтобы определить теги, ассоциированные с этими двумя пространствами имен.

Используйте метод children() для того, чтобы выделить элементы, ассоциированные с определенным пространством имен.

Как только вы выделили элементы-потомки, ассоциированные с конкретным пространством имен вызовом метода родительского элемента `children()`, вы можете продолжить доступ к членам объекта-потомка, используя оператор `->`. Например, в результате выполнения этого кода будет получен заголовок видеозаписи их тега `<media:group>`:

```
$title = $media->group->title;
```

Тег `<title>` является потомком тега `<media:group>`.

Время поработать

Используя информацию о пространствах имен и PHP-код, приведенный выше, завершите код, в результате выполнения которого будет определена длительность видеозаписи (в секундах).

```
$yt = $media->children('.....');

$attrs = .....;

echo $attrs['.....'];
```

Решение задачи

Используя информацию о пространствах имен и PHP-код, приведенный выше, завершите код, в результате выполнения которого будет определена длительность видеозаписи (в секундах).

```
$yt = $media->children('http://gdata.youtube.com/schemas/2007');
$attrs = $yt->duration->attributes();
echo $attrs['seconds'];
```

Это URL для одного из пространств имен, перечисленных между тегами <feed> и </feed> в начале документа.

Извлечение всех атрибутов для элемента <yt:duration>.

Имя атрибута используется как ключ для доступа к значению элемента массива.

не бывает глупых вопросов

В: Чем объект отличается от массива? Разве в массиве не сохранено множество различных данных?

О: Да. Объект и массив имеют много общего. Но одно принципиальное отличие между ними заключается в том, что объект в качестве своей составной части может иметь исполняемый код, называемый методом. Методы, по существу, являются функциями, членами объекта и обычно создаются для того, чтобы обрабатывать данные объекта, членами которого они являются. Массив используется исключительно для сохранения группы родственных данных и не имеет ни малейшего представления о таких понятиях, как методы. Кроме того, доступ к элементам массива осуществляется путем указания индекса (ключа) в квадратных скобках ([]), в то время как доступ к членам массива (атрибутам, методам, объектам-потомкам) осуществляется по их имени с использованием оператора ->.

В: Что, в сущности, представляет собой объект? Можем ли мы рассматривать его как обычную переменную?

О: Да. Объект можно рассматривать как переменную PHP, но только такую, которая может сохранять более сложные данные, чем обычная переменная. Вместо того чтобы сохранять строку текста или число, объект может сохранять комбинацию строк и чисел. Основная идея заключается в том, что в результате объединения связанных данных (атрибутов) и функций по их обработке (методов) алгоритм приложения и его реализация в коде становятся более логичными и завершенными.

В: Тогда как использование объектов помогает в обработке XML-данных?

О: Использование объектов очень удобно при обработке XML-данных потому, что имеется возможность смоделировать иерархию элементов XML-документа в виде иерархии объектов. Преимущество такого способа представления данных заключается в том, что вы можете перемещаться по иерархическому дереву от объектов-родителей к объектам-потомкам, используя оператор ->, и получать таким образом доступ к данным на любом уровне этой иерархии.

В: Я считал, что оператор -> используется для доступа к атрибутам объекта. Как он может предоставить мне доступ к объекту-потомку?

О: Дело в том, что, когда мы имеем дело с XML-объектами в PHP, объекты-потомки сохраняются в объекте-родителе фактически как его атрибуты. Поэтому, когда вы используете оператор -> для доступа к объекту-потомку, вы, по сути, используете его для доступа к атрибуту объекта-родителя. Объект SimpleXMLElement делает это возможным.

В: Стоп, что это за объект SimpleXMLElement?

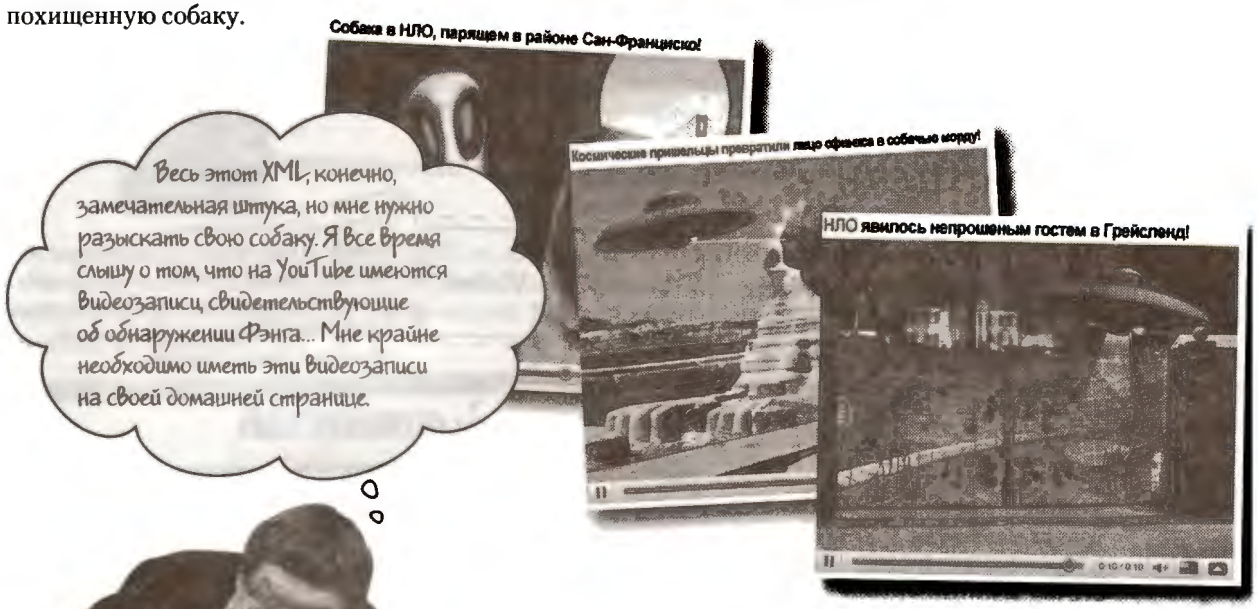
О: Каждый объект в PHP относится к какому-либо типу с учетом того, что слово «объект» является общим термином. Поэтому, когда вы создаете объект, вы создаете объект определенного типа, который призван обеспечить выполнение вполне конкретной задачи. В случае обработки XML-документа в результате вызова функции `simplexml_load_file()` создается объект типа SimpleXMLElement, то есть такого типа, который необходим нам для решения задачи по обработке XML-документа.

В: Что я должен знать об объекте SimpleXMLElement?

О: На удивление, совсем немного. Основное заключается в том, что он представляет элементы XML-документа как атрибуты-объекты типа SimpleXMLElement, у которых могут быть объекты-потомки, а те, в свою очередь, являются объектами типа SimpleXMLElement и т. д. Объекты типа SimpleXMLElement имеют методы, которые позволяют вам получить доступ к данным элементов. К этим методам относятся, в частности, функции-члены: `children()` и `attributes()`.

Работа с видеозаписями — свидетельствами наблюдения Фэнга — на подъеме

В то время как Оуэн совершенствовался в XML и решал, как лучше соединиться с YouTube, Фэнг тоже не сидел без дела. Многочисленные видеозаписи позволяли сделать вывод, что наш пес, похоже, выступал в роли гида для своих похитителей. Оуэн готов закончить свой сценарий, получить видеозаписи для демонстрации на веб-странице и найти свою похищенную собаку.



Хорошо то, что Оуэн уже почти закончил свой сценарий по получению видеозаписей с сервера YouTube. Фактически все, что осталось в нем доделать, — это закончить обработку XML-данных и отформатировать их в виде HTML-кода.

Выполните этот этап, и сценарий по получению видеозаписей с сервера YouTube будет готов!

1. ~~Составление запроса на видеозаписи YouTube.~~
2. ~~Отправление запроса на YouTube.~~
3. ~~Получение от YouTube ответа, содержащего информацию о видеозаписях.~~
4. **Обработка данных ответа и форматирование их в виде HTML-кода.**

Создайте схему, как бы вы представили данные, полученные с сервера YouTube, в форме видеоклипов в нижней части страницы «Космические пришельцы похищали меня».

Размещение видеозаписей для просмотра

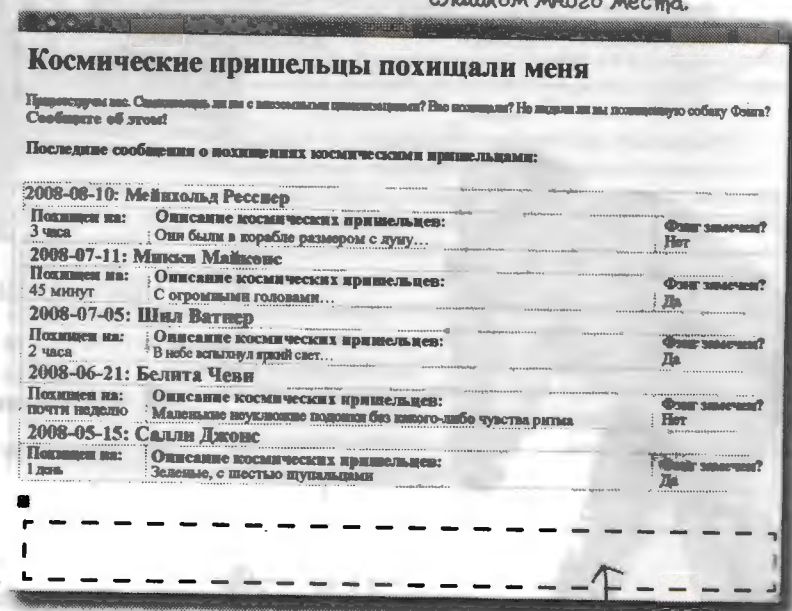
Это видеозаписи, полученные в динамическом режиме с сервера YouTube в виде XML-документа.

Идея, лежащая в основе сценария `youtube.php`, заключается в том, что он будет включен в главный сценарий `index.php` приложения «Космические пришельцы похищали меня». Это означает, что сценарий `youtube.php` должен предусматривать создание и отправку запроса на видеозаписи, обработку полученного XML-документа и форматирование отдельных записей так, чтобы они могли демонстрироваться согласованно с сообщениями о похищении космическими пришельцами, которые уже имеются на главной странице. Неплохой способ достичь этого заключается в том, чтобы расположить видеозаписи горизонтально вдоль нижней границы страницы.

Это ряд изображений предварительного просмотра, что должен создать сценарий youtube.php.

Пять изображений предварительного просмотра — вполне достаточное количество для расположения их горизонтально в ряд, чтобы они не занимали слишком много места.

Сценарий youtube.php будет включен в приложение так, чтобы изображения предварительного просмотра видеозаписей появились сразу под списком сообщений о похищении космическими пришельцами.



Размещение изображения предварительного просмотра видеозаписей горизонтально на главной странице не позволяет им занимать слишком много места у списка сообщений о похищении космическими пришельцами. Кроме того, мы говорим о размещении изображений предварительного просмотра, а не самих видеозаписей, поэтому пользователи должны еще щелкнуть кнопкой мыши на одном из этих изображений, чтобы соединиться с сервером YouTube и начать просмотр самой видеозаписи. Видеоролики слишком велики по объему и потребовали бы слишком много ресурсов, если бы мы включили их, а не изображения предварительного просмотра в приложения «Космические пришельцы похищали меня».

Вот отличное место для размещения изображений предварительного просмотра видеозаписей, чтобы посетители могли легко получить к ним доступ.

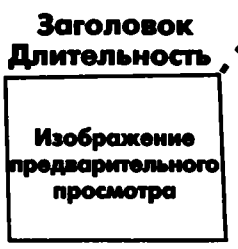
Форматирование видеоданных для вывода на дисплей

Хотя изображение предварительного просмотра видеозаписи играет очень важную роль в принятии решения, есть ли смысл просматривать саму видеозапись или нет, оно не является единственной информацией, полезной для сценария Оуэна по получению видеозаписей с сервера YouTube. Например, заголовок видеозаписи может содержать весьма важные данные, вроде упоминаний о собаке. Длительность видеозаписи также может иметь значение. И, конечно, нам необходима ссылка на URL, указывающий на место размещения видеозаписи, чтобы пользователь мог приступить к ее просмотру простым щелчком кнопкой мыши по изображению предварительного просмотра. Итак, вот та информация, которую нам необходимо извлечь из XML-документа, полученного с сервера YouTube:

Заголовок **Длительность** **Изображение предварительного просмотра** **Ссылка**

Для размещения видеозаписей YouTube на веб-странице требуется несколько различных видов данных.

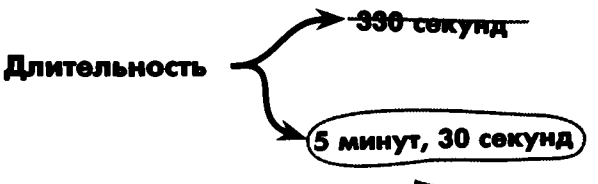
Эти данные формируют основу HTML-кода, в результате выполнения которого в нижней части главной страницы будет создана горизонтальная строка изображений предварительного просмотра видеозаписей. Каждый элемент этой строки будет выглядеть примерно так:



Эта гиперссылка содержит URL видеозаписи, размещенной на сервере YouTube, и переход по ней осуществляется при щелчке кнопкой мыши на заголовке видеозаписи, ее длительности или изображении предварительного просмотра.

В XML-документе, полученном с сервера YouTube, длительность видеозаписи указана в теге `<yt:duration>` в секундах. Однако большинство из нас не определяют время в секундах, нам более привычно выражение времени в минутах и секундах. Например, далеко не сразу очевидно, что 330-секундная видеозапись будет длиться 5 минут и 30 секунд. Представление временных интервалов в таком виде имеет для нас больший смысл, но для того чтобы этого добиться, нам нужно произвести в уме ряд математических операций. Учитывая сказанное, будет правильным произвести все эти математические преобразования непосредственно в сценарии, перед тем как выводить значение длительности видеозаписи на экран монитора.

Это означает, что если вы не являетесь членом руководства YouTube, то вы не можете загрузить на сервер видеозапись длительностью более 10 минут.



Более привычная и понятная большинству форма выражения временных интервалов.

Чокнутые биты

Нет необходимости учитывать в выражении длительности видеозаписи также и часы, так как YouTube в настоящее время ограничивает длительность загружаемых на сервер видеозаписей временем в 10 минут.

УПРАЖНЕНИЕ

В результате выполнения кода сценария youtube.php извлекается информация о пяти самых первых видеозаписях из результата поиска на сервере YouTube. Затем изображения предварительного просмотра этих видеозаписей располагаются в один ряд в нижней части главной страницы. Эти изображения вместе с заголовками и длительностями кодируются в виде гиперссылок на сами видеозаписи, размещенные на сервере YouTube. Добавьте в сценарий пропущенный код, используя как руководство пример XML-документа, полученного с сервера YouTube и рассмотренного на предыдущей странице.

```
<?php
 define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first');
 define('NUM_VIDEOS', 5);

 // Создание иерархии объектов PHP из иерархии элементов XML
 $xml = ..... (YOUTUBE_URL);

 $num_videos_found = count( ..... );
 if ($num_videos_found > 0) {
  echo '<table><tr>';
  for ($i = 0; $i < min($num_videos_found, NUM_VIDEOS); $i++) {
   // Извлечение заголовка
   $entry = $xml->entry[$i];
   $media = $entry->children('http://search.yahoo.com/mrss/');
   $title = $media->group-> ..... ;

   // Извлечение длительности в минутах и секундах и форматирование результата
   $yt = $media->children('http://gdata.youtube.com/schemas/2007');
   $attrs = $yt->duration->attributes();
   $length_min = floor($attrs[' ..... '] / 60);
   $length_sec = $attrs[' ..... '] % 60;
   $length_formatted = $length_min . (($length_min != 1) ? ' minutes, ':' minute, ') .
    $length_sec . (($length_sec != 1) ? ' seconds':' second');

   // Извлечение URL видеозаписи
   $attrs = $media->group->player-> ..... ();
   $video_url = $attrs['url'];
```

```
    // Извлечение URL изображения предварительного просмотра видеозаписи
    $attrs = $media->.....->thumbnail[0]->attributes();
    $thumbnail_url = $attrs['url'];
    // Вывод результатов для этой видеозаписи
    echo '<td style="vertical-align:bottom; text-align:center" width="' . (100 / NUM_VIDEOS) .
     '%"><a href="' . $video_url . '">' . ..... '<br /><span style="font-size:smaller">' .
     $length_formatted . '</span><br /><img src="' ..... '" /></a></td>';
   }
   echo '</tr></table>';
  }
  else {
   echo '<p>Извините, видеозапись не найдена.</p>';
  }
 ?>
```

Используйте этот пример XML-кода при разработке PHP-кода.

```
...
<entry>
 <id>http://gdata.youtube.com/feeds/api/videos/_6UibqfOvtA</id>
 <published>2006-06-20T07:49:05.000-07:00</published>
 ...
 <media:group>
  <media:title type='plain'>НЛО замечено в Йосмитском национальном парке, в районе зоны 51</media:title>
  <media:description type='plain'>Я путешествовал по Йосмитскому национальному парку возле зоны 51 в 2002 году.
Йосмитский национальный парк расположен очень близко к границе между Калифорнией и Невадой и очень близко к зоне
51...</media:description>
  <media:keywords>51, космический, пришелец, космические, пришельцы, зона, Калифорния, Невада, наблюдение,
наблюдения. НЛО</media:keywords>
  <yt:duration seconds='50'/>
  <media:category label='Travel &amp;amp; Events'
   scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
  <media:content url='http://www.youtube.com/v/_6UibqfOvtA' type='application/x-shockwave-flash'
   medium='video' isDefault='true' expression='full' duration='50' yt:format='5'/>
  <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbiKl_xMYDSANFEgGDA==/0/0/0/video.3gp'
   type='video/3gpp' medium='video' expression='full' duration='50' yt:format='1'/>
  <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbiKl_xMYESARFEgGDA==/0/0/0/video.3gp'
   type='video/3gpp' medium='video' expression='full' duration='50' yt:format='6'/>
  <media:player url='http://www.youtube.com/watch?v=_6UibqfOvtA'/>
  <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/2.jpg' height='97' width='130'
   time='00:00:25'/>
  <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/1.jpg' height='97' width='130'
   time='00:00:12.500'/>
  <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/3.jpg' height='97' width='130'
   time='00:00:37.500'/>
  <media:thumbnail url='http://img.youtube.com/vi/_6UibqfOvtA/0.jpg' height='240' width='320'
   time='00:00:25'/>
 </media:group>
 <yt:statistics viewCount='2478159' favoriteCount='1897'/>
 <gd:rating min='1' max='5' numRaters='1602' average='4.17'/>
 <gd:comments>
  <gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6UibqfOvtA/comments'
   countHint='4426'/>
 </gd:comments>
</entry>
<entry>
 ...
</entry>
...
```

Заголовок видеозаписи.

Длительность видеозаписи в секундах.

Ссылка на видео на сервере YouTube.

URL изображения предварительного просмотра видеозаписи.

РЕШЕНИЕ К УПРАЖНЕНИЮ

В результате выполнения кода сценария youtube.php извлекается информация о пяти самых первых видеозаписях из результата поиска на сервере YouTube. Затем изображения предварительного просмотра этих видеозаписей располагаются в один ряд в нижней части главной страницы. Эти изображения вместе с заголовками и длительностями кодируются в виде гиперссылок на сами видеозаписи, размещенные на сервере YouTube. Добавьте в сценарий пропущенный код, используя как руководство пример XML-документа, полученного с сервера YouTube и рассмотренного на предыдущей странице.

```
<?php
 define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first');
 define('NUM_VIDEOS', 5);

 // Создание иерархии объектов PHP из иерархии элементов XML
 $xml = simplexml_load_file(YOUTUBE_URL);

 $num_videos_found = count($xml->entry);
 if ($num_videos_found > 0) {
  echo '<table><tr>';
  for ($i = 0; $i < min($num_videos_found, NUM_VIDEOS); $i++) {
   // Извлечение заголовка
   $entry = $xml->entry[$i];
   $media = $entry->children('http://search.yahoo.com/mrss/');
   $title = $media->group->title;

   // Извлечение длительности в минутах и секундах и форматирование результата
   $yt = $media->children('http://gdata.youtube.com/schemas/2007');
   $attrs = $yt->duration->attributes();
   $length_min = floor($attrs['seconds'] / 60);
   $length_sec = $attrs['seconds'] % 60;
   $length_formatted = $length_min . (($length_min != 1) ? ' minutes, ':' minute, ')
    $length_sec . (($length_sec != 1) ? ' seconds':' second');

   // Извлечение URL видеозаписи
   $attrs = $media->group->player->attributes();
   $video_url = $attrs['url'];
```

Количество видеозаписей для вывода на главной странице сохраняется в константе.

URL поиска по ключевым словам на сервере YouTube.

Для извлечения данных из XML-документа, полученного с сервера YouTube, используется функция simplexml_load_file().

Проверка количества видеозаписей, полученных с сервера YouTube.

Прохождение каждой видеозаписи в цикле.

Извлечение всех объектов-потомков этого объекта-родителя, относящихся к пространству имен Yahoo! Media.

Извлечение заголовка видеозаписи, который находится в теге <media:title>.

Извлечение всех объектов-потомков этого объекта-родителя, относящихся к пространству имен YouTube yt.

Извлечение значения длительности видеозаписи в секундах и преобразование ее в минуты и секунды.

Извлечение ссылки из атрибута url тега <media:player>.

```
// Извлечение URL изображения предварительного просмотра видеозаписи
$attrs = $media->group->thumbnail[0]->attributes();
$thumbnail_url = $attrs['url'];

// Вывод результатов для этой видеозаписи
echo '<td style="vertical-align:bottom; text-align:center" width="' . (100 / NUM_VIDEOS) .
 '%"><a href="' . $video_url . '">' . $title . '<br /><span style="font-size:smaller">' .
 $length_formatted . '</span><br /><img src="' . $thumbnail_url . '" /></a></td>';
 }
 echo '</tr></table>';
}
else {
 echo '<p>Извините, видеозапись не найдена.</p>';
 }
?>
```

Извлечение URL первого изображения предварительного просмотра из атрибута url тега <media:thumbnail>.

Форматирование результата в виде ячеек таблицы с заголовком, длительностью и изображением предварительного просмотра.

1. ~~Составление запроса на видеозаписи YouTube.~~
2. ~~Отправление запроса на YouTube.~~
3. ~~Получение от YouTube ответа, содержащего информацию о видеозаписях.~~
4. ~~Обработка данных ответа и форматирование их в виде HTML-кода.~~

Готово!

```
<entry>
 <id>http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA</id>
 <published>2006-06-20T07:49:05.000-07:00</published>
 ...
 <media:group>
  <media:title type='plain'>НЛО замечено в Йосмитском национальном парке, в районе зоны 51</media:title>
  <media:description type='plain'>Я путешествовал по Йосмитскому национальному парку возле зоны 51 в 2002 году.
Йосмитский национальный парк расположен очень близко к границе между Калифорнией и Невадой и очень близко к зоне
51…</media:description>
  <media:keywords>51,космический, пришелец, космические, пришельцы, зона, Калифорния, Невада, наблюдение,
наблюдения. НЛО</media:keywords>
  <yt:duration seconds='50'/>
  <media:category label='Travel &amp;amp; Events'
   scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
  <media:content url='http://www.youtube.com/v/_6Uibqf0vtA' type='application/x-shockwave-flash'
   medium='video' isDefault='true' expression='full' duration='50' yt:format='5'/>
  <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbiKl_xMYDSANFEgGDA==/0/0/0/video.3gp'
   type='video/3gpp' medium='video' expression='full' duration='50' yt:format='1'/>
  <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbiKl_xMYESARFEgGDA==/0/0/0/video.3gp'
   type='video/3gpp' medium='video' expression='full' duration='50' yt:format='6'/>
  <media:player url='http://www.youtube.com/watch?v=_6Uibqf0vtA' />
  <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/2.jpg' height='97' width='130'
   time='00:00:25'/>
  <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/1.jpg' height='97' width='130'
   time='00:00:12.500'/>
  <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/3.jpg' height='97' width='130'
   time='00:00:37.500'/>
  <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/0.jpg' height='240' width='320'
   time='00:00:25'/>
 </media:group>
 <yt:statistics viewCount='2478159' favoriteCount='1897'/>
 <gd:rating min='1' max='5' numRaters='1602' average='4.17'/>
 <gd:comments>
  <gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA/comments'
   countHint='4426'/>
 </gd:comments>
</entry>
<entry>
 ...
</entry>
...
```

Заголовок видеозаписи.

Длительность видеозаписи в секундах.

Ссылка на видео на сервере YouTube.

URL изображения предварительного просмотра видеозаписи.

Тест-драйв

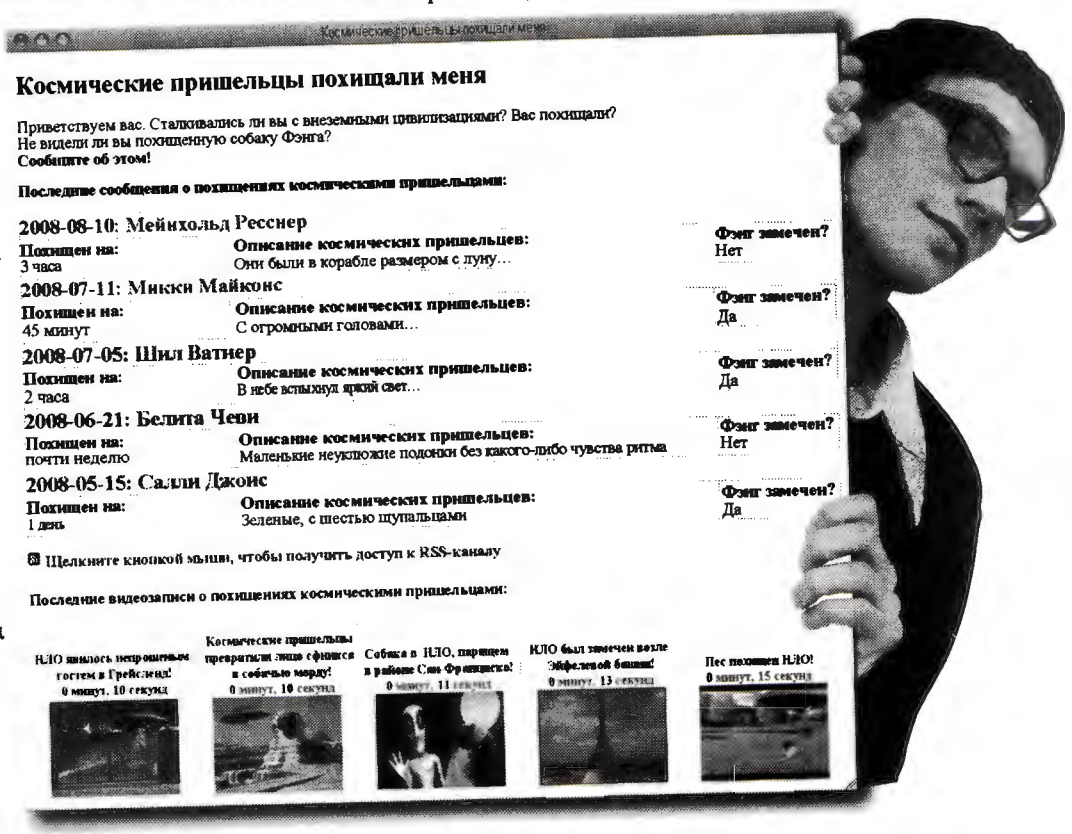
Добавьте сценарий `youtube.php` в приложение «Космические пришельцы похищали меня».

Создайте новый текстовый файл с именем `youtube.php` и поместите в него код сценария, разработанный нами на двух предыдущих страницах (или загрузите сценарий с сайта по адресу `www.headfirstlabs.com/books/hfphp`). Вы также должны включить код в сценарий `index.php`, чтобы в нижней части главной страницы приложения «Космические пришельцы похищали меня» появился ряд из пяти гиперссылок на видеозаписи YouTube в виде изображений их предварительного просмотра. Вот две строки PHP-кода, в результате выполнения которого этот сценарий `youtube.php` будет включен в сценарий `index.php`:

```
echo '<h4>Последние видеозаписи о похищениях космическими пришельцами:</h4>';
require_once('youtube.php');
```

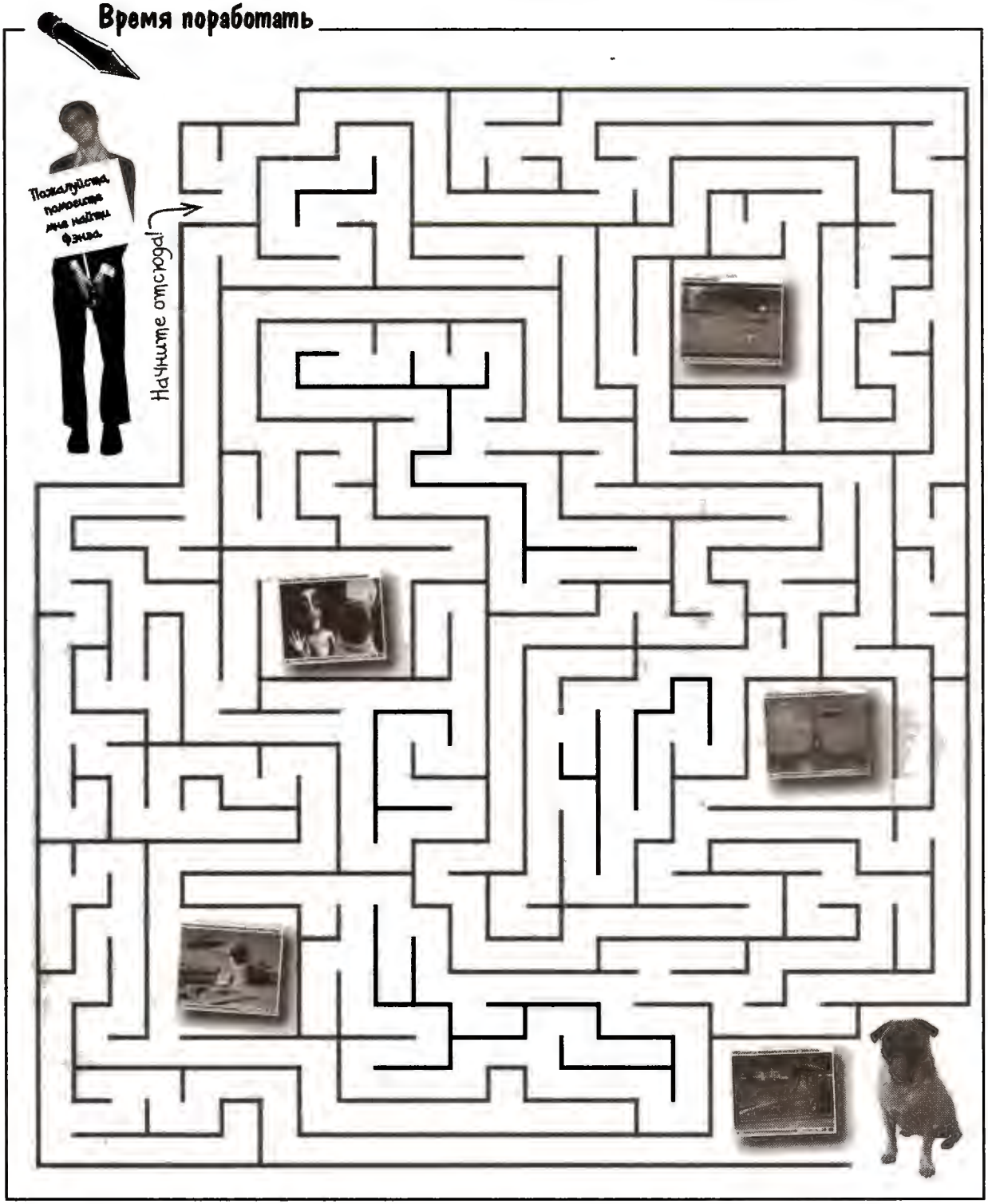
Включение сценария youtube.php в сценарий index.php — все, что необходимо, чтобы в нижней части главной страницы приложения «Космические пришельцы похищали меня» появился ряд из пяти гиперссылок на видеозаписи YouTube о похищениях.

Загрузите сценарии на ваш веб-сервер и откройте главную страницу приложения «Космические пришельцы похищали меня» в браузере. В нижней части главной страницы должен появиться ряд из пяти гиперссылок на видеозаписи YouTube, имеющие отношение к похищению космическими пришельцами.

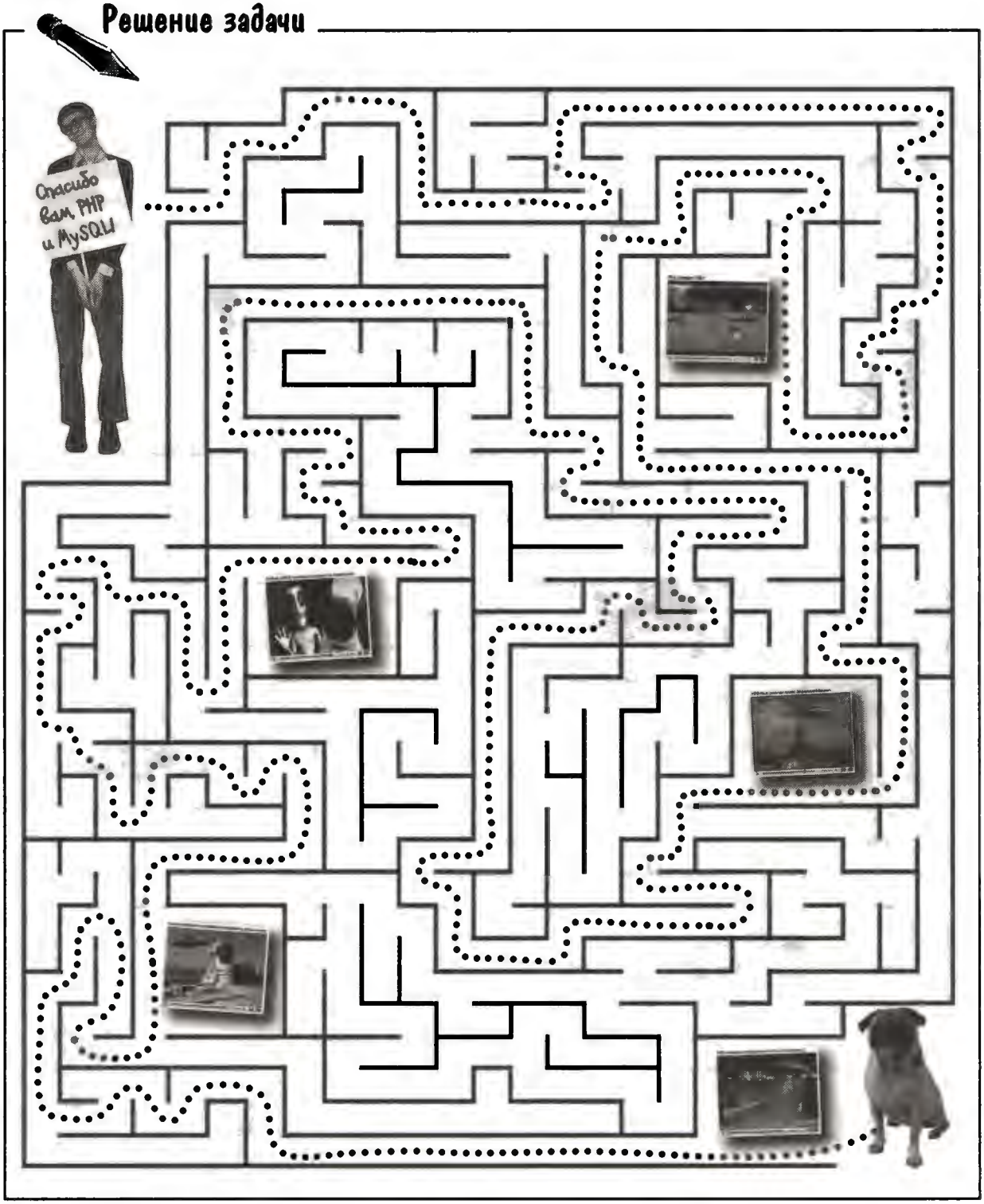


Видеозаписи с сервера YouTube помогли Оуэну сузить круг поиска Фэнга.

Время поработать



Решение задачи



Ваш инструментарий PHP и MySQL

Принимая во внимание всю эту историю с Фэнгом, полезно немного поразмышлять, от чего зависел успех его обнаружения. Как оказалось, PHP и MySQL потребовалась помощь нескольких других технологий.

REST

Способ получения доступа к информации на сайте исключительно через URL. REST дает вам возможность делать достаточно сложные запросы всего лишь путем составления соответствующих URL.

XML

Расширяемый язык разметки, используемый для представления структуры данных. Существует множество различных языков разметки, основанных на XML, такие, например, как XHTML и RSS. Основная идея заключается в том, что вы создаете набор тегов, описывающих представление данных в документе.

RSS

Язык разметки, основанный на XML и предназначенный для распространения информации (например, новостей) в Интернете. RSS дает возможность сайтам предоставлять свои данные другим сайтам и приложениям.

`simplexml_load_file()`

Встроенная функция PHP, загружающая XML-документ по его URL и возвращающая иерархию объектов, соответствующую иерархии элементов XML-документа.

`SimpleXMLElement`

Встроенный объект PHP, который используется для доступа к данным XML-документа. Этот объект возвращается функцией `simplexml_load_file()` и содержит иерархию объектов, соответствующую иерархии элементов XML-документа.

Пространство имен

Способ объединения набора XML-тегов в логические группы примерно так же, как фамилия объединяет имена членов вашей семьи в одну логическую группу. Пространство имен всегда ассоциируется с URL, что обеспечивает уникальность между различными пространствами имен.

Конец.

Приложение I дополнения

Десять основных тем (которые мы не затронули)

Даже после всего изложенного есть еще некоторые вещи, которые, как мы думаем, вам нужно знать. Мы были бы не правы, игнорируя их, хотя они и требуют только краткого упоминания. Поэтому перед тем, как отложить книгу в сторону, обратите внимание на эти краткие, но важные детали, касающиеся PHP и MySQL. Кроме того, раз уже вы дочитали до этого места, все, что вам осталось, — это несколько коротких дополнений… и алфавитный указатель… и, возможно, немного рекламы… и вот тогда книга действительно закончится.

№ 1. Модификация кода этой книги для поддержки функций PHP 4 и MySQL

За исключением функций XML, приведенных в главе 12, большая часть кода в этой книге будет выполняться на серверах PHP 4 с небольшими изменениями. В этой книге мы использовали те функции семейства `mysqli`, которые доступны только в PHP 4.1 и более поздних его версиях, а так как эта библиотека должна устанавливаться вручную, некоторые серверы не будут их поддерживать.

Функции `mysqli` выполняются в целом быстрее, но это начинает иметь значение только тогда, когда ваша база данных становится очень объемной. Доступ к базам данных небольших или средних размеров не будет заметно тормозиться при использовании более старых функций `mysql`. Этот раздел написан для того, чтобы рассказать вам, как модифицировать ваши `mysqli`-функции, чтобы они выполнялись как `mysql`-функции в старых версиях PHP.

Если вы видите:

```
$dbc = mysqli_connect(localhost, 'mork', 'fromork');
mysqli_select_db($dbc, 'alien_database');
```

Переменная соединения с базой данных (`$dbc`) не является здесь первым по порядку аргументом, как в случае с `mysqli_select_db()`.

используйте:

```
$dbc = mysql_connect(localhost, 'mork', 'fromork');
mysql_select_db('alien_database', $dbc);
```

По сути, вы просто удаляете символ `i` из слова `mysqli`, преобразуя его в `mysql`, и затем меняете порядок аргументов таким образом, чтобы переменная соединения с базой данных (`$dbc` в данном примере) стала последней в списке аргументов.

Но это становится немного сложнее в случае, если функция `mysqli_connect()` вызывается с именем базы данных, что делает вызов функции `mysqli_select_db()` излишним. В семействе функций `mysql` ничего подобного нет. Для повторения того же, что делает единственная функция `mysqli_connect()`, вызываемая с именем базы данных, вам понадобится две функции `MySQL`.

Если вы видите:

```
$dbc = mysqli_connect(localhost, 'mork', 'fromork', 'alien_database');
```

Здесь база данных `alien_database` выбирается как составная часть соединения с ней, что невозможно сделать в один этап, используя функции MySQL.

вам необходимо использовать две команды:

```
$dbc = mysql_connect(localhost, 'mork', 'fromork');
mysql_select_db('alien_database', $dbc);
```

При использовании функций MySQL всегда необходимо сделать два вызова функций, чтобы установить соединение с определенной базой данных.

Эта переменная также известна под названием «ссылка» на соединение.

Здесь показано, как соотносятся между собой функции `mysql` и `mysqli`.

Закрыть MySQL-соединение	`mysqli_close(conn)`	`mysqli_close(conn)`
Открыть соединение с MySQL-сервером	`mysql_connect(host, username, password)` Вы должны использовать функцию `mysql_select_db()`, чтобы выбрать базу данных.	`mysqli_connect(host, username, password, database)` Вам необязательно использовать функцию `mysqli_select_db()`, чтобы выбрать базу данных.
Вернуть текст сообщения об ошибке предыдущей MySQL-операции	`mysql_error(conn)`	`mysqli_error(conn)`
Экранировать специальные символы в строке	`mysql_escape_string(string, conn)` Обратный порядок следования аргументов: вначале строка, затем ссылка на соединение.	`mysqli_escape_string(conn, string)` Вначале ссылка на соединение, затем строка.
Обработать запись и вернуть ассоциативный массив, индексный массив или и тот, и другой	`mysql_fetch_row(result)`	`mysqli_fetch_row(result)`
Определить количество записей в результате запроса	`mysql_num_rows(result)`	`mysqli_num_rows(result)`
Выполнить MySQL-запрос	`mysql_query(conn, query)`	`mysqli_query(conn, query)`
Экранировать специальные символы в строке с учетом кодировки	`mysql_real_escape_string(string, conn)` Обратный порядок следования аргументов: вначале строка, затем ссылка на соединение.	`mysqli_real_escape_string(conn, string)` Вначале ссылка на соединение, затем строка.
Выбрать базу данных MySQL	`mysql_select_db(dbname, conn)` Обратный порядок следования аргументов: вначале база данных, затем ссылка на соединение.	`mysqli_select_db(conn, dbname)` Вначале ссылка на соединение, затем база данных.

№ 2. Права пользователей MySQL

Предположим, вы создали веб-приложение, которое позволяет пользователям только просматривать данные таблицы вашей базы данных. Вы делаете запрос к определенной базе на необходимые вам данные, и MySQL предоставляет вам эту информацию.

Но представьте себе такую ситуацию: имя пользователя и его пароль, передаваемые с функцией `mysqli_connect()`, в случае непосредственного соединения через терминал MySQL или графическую оболочку могли бы позволить пользователю добавлять, изменять или удалять данные с помощью запросов INSERT, UPDATE и DELETE соответственно.

Если вашему приложению нет необходимости выполнять подобные операции, нет никакого резона, чтобы имя пользователя и его пароль, применяемые для подключения, могли быть использованы для выполнения таких же операций непосредственно, минуя приложение. MySQL позволяет вам ограничить доступ к вашей базе данных. Вы можете настроить MySQL таким образом, чтобы разрешить пользователю только просматривать (`SELECT`) данные. Или просматривать (`SELECT`) и добавлять (`INSERT`). Или любую другую комбинацию запросов, которая вам необходима.

И что впечатляет еще больше, вы можете управлять доступом к отдельным таблицам. Например, если ваше приложение работает только с таблицей `alien_info` и не нуждается в данных таблицы `cyborg_info`, вы можете ограничить доступ к последней.

Первое, что вы вправе сделать, — это создать совершенно нового пользователя и его пароль для этого приложения. Вы можете сделать это через MySQL-терминал:

```
Файл Правка Окно Помощь Пришельцы!
mysql> CREATE USER alienguy IDENTIFIED BY 'aliensRsOdry';
Запрос выполнен успешно. 0 строки в списке (0.07 сек)
```

Затем с помощью запроса MySQL `GRANT` вы можете определить, что пользователь `alienguy` может делать с вашей базой данных. Если ему необходимо только просматривать (`SELECT`) и добавлять (`INSERT`) данные, это будет выглядеть так:

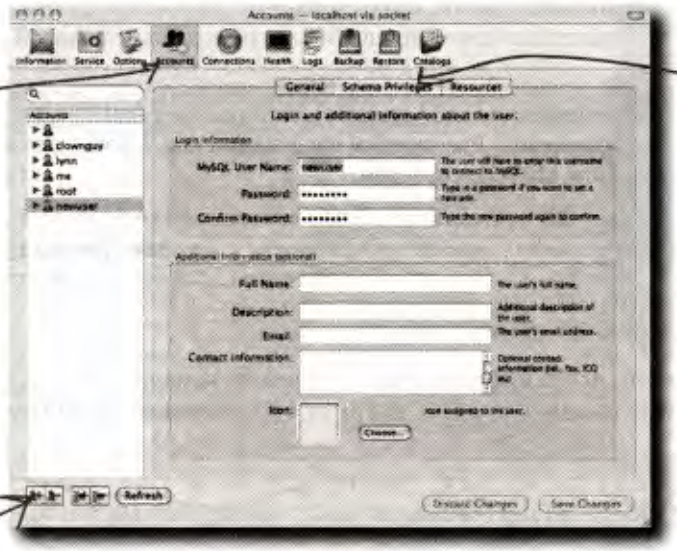
```
Файл Правка Окно Помощь Пришельцы!
mysql> USE alien_database';
База данных изменена
mysql> GRANT SELECT, INSERT ON alien_info TO alienguy';
Запрос выполнен успешно. 0 строки в списке (0.03 сек)
```

Если вы не хотите использовать MySQL-терминал для создания пользователей и установки их прав доступа, вы можете загрузить и установить удобную программу MySQLAdministrator. Она доступна по адресу:

`http://dev.mysql.com/downloads/gui-tools/5.0.html`.

Вы можете исключительно выборочно устанавливать права доступа для каждого пользователя и даже определять, что ваш пользователь может делать с конкретной колонкой конкретной таблицы. Более подробно это описано в книге Head First SQL.

MySQLAdministrator позволяет вам управлять учетными записями пользователей и устанавливать, к каким ресурсам вашей базы данных каждая такая учетная запись имеет доступ. Программа даже разрешает вам определять, какие виды запросов конкретный пользователь может делать к каждой таблице вашей базы.

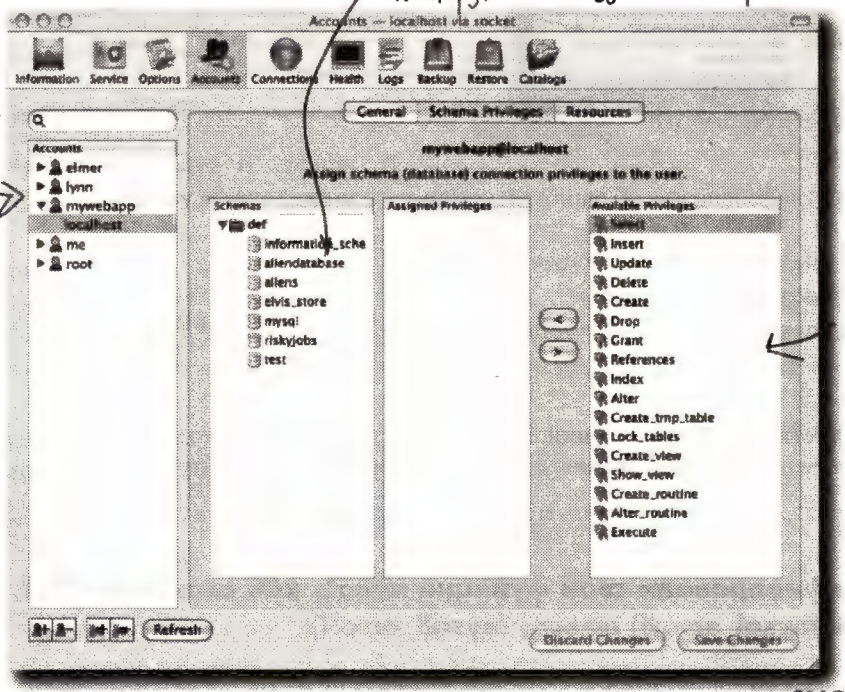


Вначале нажмите кнопку Accounts.

После того как вы назначили имя пользователя и его пароль, щелкните здесь, чтобы назначить ему права доступа.

Затем используйте эту кнопку, чтобы добавить новую учетную запись.

Здесь приведен список таблиц конкретной базы данных. Выберите ту, которую использует ваше приложение.



Здесь приведен список пользователей. Вы можете создавать новых пользователей, чтобы определить конкретный набор прав доступа для каждого приложения. Выберите того пользователя, чью конфигурацию вы хотите изменить.

Если вы просмотрите этот список, то увидите основные запросы MySQL, с которыми уже сталкивались, читая эту книгу. Выберите только те, которые требуются вашему приложению для работы.

№ 3. Сообщения об ошибках MySQL

Во многих примерах нашего кода вы часто видите подобные строки:

```
mysqli_connect(localhost, 'mork', 'fromork') or die ('Соединение не удалось.');
```

Когда такую команду выполнить не удается, на веб-странице появляется строка «Соединение не удалось». Это говорит нам о том, что произошла какая-то ошибка, но не содержит никакой дополнительной информации.

К счастью, PHP предлагает функцию `mysql_error()`, которая дает нам ключ к объяснению того, что же в действительности произошло. Рассмотрим следующий код, с помощью которого мы пытаемся соединиться с несуществующим MySQL-сервером.

```
<?php
  mysqli_connect('badhostname', 'mork', 'fromork') or die (mysqli_error());
?>
```

Вы увидите вот такое сообщение об ошибке.

```
Неизвестный сервер MySQL 'badhostname' (1)
```

В результате будет получена ясная информация о том, почему команду `mysqli_connect()` выполнить не удалось. Вы можете использовать `mysqli_error()` с другими функциями `mysqli`:

```
<?php
  $dbc = mysqli_connect('localhost', 'mork', 'fromork');
  mysqli_select_db($dbc, 'alien_database');
  echo mysqli_error($dbc) . '<br />';
  mysqli_select_db($dbc, 'alien_database');
  mysqli_query($dbc, "SELECT * FROM alien_info");
  echo mysqli_er
?>
```

Мы пытаемся соединиться с несуществующей базой данных.

Мы пытаемся извлечь данные из несуществующей таблицы.

Вывод на экран:

Вот некоторые другие сообщения об ошибках, которые вы также можете увидеть:

```
Таблица 'test.no_such_table' не существует
Невозможно создать таблицу
Невозможно создать базу данных 'yourdatabase'; база данных существует
Невозможно удалить базу данных 'yourdatabase'; база данных не существует
```

Существует множество других сообщений, и мы бы просто попусту расходовали бумагу, перечисляя их здесь. Для получения более подробной информации обратитесь к сайту:

```
http://dev.mysql.com/doc/refman/5.0/en/error-messages-server.html
```

Если вы модифицировали свои функции `mysql`, как сказано в № 1, то можете использовать mysql_error() вместо mysqli_error().

№ 4. Обработка исключений (Exceptions) в PHP

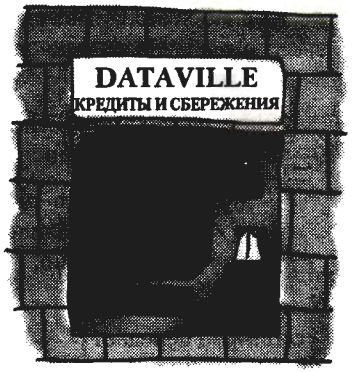
Обработка исключений (Exceptions) позволяет вам изменить нормальный ход процесса исполнения программы и выполнить специальный блок кода в случае возникновения какой-либо исключительной ситуации. PHP 5 и 6 предлагают такую возможность. Далее следует краткое описание.

Предположим, вы хотите получить 200 баксов в банкомате.

Но предположим также, что минимальный баланс на вашем счете не может быть меньше 1000 долларов, а эта операция по снятию денег приведет к тому, что ваш баланс станет меньше этой суммы. Это запрещено.

Транзакция будет отвергнута!

Фрагмент кода, приведенный ниже, показывает, как обработка исключений в PHP позволяет перехватить и обработать эту неудавшуюся транзакцию.




```
<?php
  function checkBalance($balance) {
    if($balance < 1000) {
      throw new Exception(Баланс менее 1000 долларов);
    }
    return true;
  }
  try {
    checkBalance(999);
    echo 'Баланс более 1000 долларов.';
  }
  catch(Exception $e) {
    echo 'Error: ' . $e->getMessage();
  }
?>
```

Это сообщение, которое мы отправим вам, если ваш баланс станет менее 1000 долларов.

Блок try используется для проверки значений без прерывания нормального хода процесса.

Здесь мы проверяем наш баланс.

Если произошло исключение, мы исполняем этот блок кода. В данном случае на экран выводится наше сообщение.

В процессе исполнения этого кода вы увидите следующее:

```
Ошибка: Баланс менее 1000 долларов.
```

№ 4. Обработка исключений (Exceptions) в PHP (продолжение)

Обработчик исключений состоит из трех блоков кода.

1) try — в этом блоке вы проверяете, действительно ли ваши значения соответствуют заданным.

Если это так, тогда все отлично, и ваш код продолжает обрабатываться дальше в обычном порядке. Если нет — генерируется исключение. На программистском жаргоне — исключение «вбрасывается».

А если что-то вброшено, должно быть что-то, что бы перехватило его. Если исключение сгенерировано, исполняется блок catch, если нет — исполнение кода происходит как обычно.

2) throw — передает управление блоку catch и посылает ему сообщение об ошибке. Каждому throw соответствует по крайней мере один блок catch.

3) catch — создается **объект** с информацией об исключении. Более подробно об объектах далее.

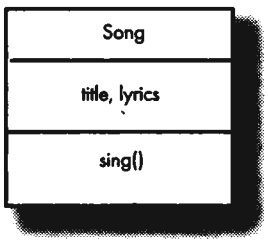
```
<?php
  function checkBalance($balance) {
    if($balance < 1000) {
      throw new Exception(Баланс менее 1000 долларов.);
    }
    return true;
  }
  try {
    checkBalance(999);
    echo 'Баланс более 1000 долларов.';
  }
  catch(Exception $e) {
    echo 'Error: ' . $e->getMessage();
  }
?>
```

№ 5. Объектно-ориентированный PHP

Объектно-ориентированные языки используют модели программирования, отличающиеся в значительной степени от моделей, используемых процедурными аналогами. До сих пор вы использовали PHP процедурно, но он имеет также и объектно-ориентированную сторону. Вместо того чтобы представлять программу в виде набора последовательных, шаг за шагом выполняемых инструкций, она составляется из определенных структурных элементов, называемых объектами. **Объекты включают не только описание данных, но также и все операции, которые могут быть использованы для их обработки.** Когда вы используете объектно-ориентированный PHP, вы создаете **объекты** и манипулируете ими.

Прежде чем мы приступим к обсуждению, почему у вас могло бы появиться желание использовать объектно-ориентированный PHP, давайте напишем кое-что.

Это класс Song, определяющий наш объект.



❶ Напишите ваш класс.

```
class Song
{
  var $title;
  var $lyrics;

  function Song($title, $length) {
    $this->title = $title;
    $this->lyrics = $lyrics;
  }

  function sing() {
   echo 'Это называется ' . $this->title . '.<br />';
   echo 'Раз, два, три… ' . $this->lyrics;
  }
}
```

Это переменные объекта.

Здесь определяются название и слова песни в момент ее создания.

Это метод, который использует переменные объекта.

❷ Создайте новый объект.

```
$shoes_song = new Song('Голубые замшевые туфли',
'Раз, два, три… Так, раз — это деньги…');

$shoes_song->sing();
```

Название нашей новой песни «Голубые замшевые туфли».

Здесь мы вызываем метод нашего объекта sing().

❸ Ваша песня исполняется.

Когда вы исполните этот код, вы получите следующее:

Но зачем все эти объекты, если мы можем просто вывести текст с помощью команды echo? Так ли нам нужен этот объектно-ориентированный PHP?

На это есть веские причины...

№ 5. Объектно-ориентированный PHP (продолжение)

Вместо того чтобы представлять программу в виде набора последовательных, шаг за шагом выполняемых инструкций, она составляется из определенных структурных элементов, называемых объектами. **Объекты включают не только описание данных, но также и все операции, которые могут быть использованы для их обработки**. В нашем примере `Song` мы определяем название песни и ее слова внутри класса и создаем метод `sing()` также внутри класса. Если нам необходимо добавить функциональности к нашему объекту `Song`, мы можем добавить новые переменные и методы к нашему классу `Song`. Например, если мы захотим добавить автора песни, связанного с каждым объектом `Song`, мы можем добавить соответствующую переменную к нашему классу.

Возможности объектно-ориентированных методов программирования проявляются наиболее ярко по мере увеличения объема приложения. Предположим, вы решили использовать класс `Song` как часть приложения караоке с сотнями или даже тысячами индивидуальных объектов — песен — со своими словами, названием и автором. А теперь, например, кто-нибудь захотел выбрать только песни, написанные Элвисом. Все, что мы должны сделать, — это проверить переменную `songwriter` каждого объекта.

А передать слова песни приложению караоке? Мы могли бы просто вызвать метод `sing()` для каждого объекта, обрабатываемого в текущий момент. И хотя мы вызываем один и тот же метод каждого объекта, он использует данные того объекта, от которого вызван.

Таким образом, два больших достоинства использования объектно-ориентированного PHP — это:

- объекты легко могут быть использованы повторно; они создаются независимо от кода, который их использует, и могут быть использованы повторно при необходимости;
- код становится легче для понимания и поддержки; если необходимо изменить тип данных, такие изменения производятся только в самом объекте и больше ни в каком другом месте.

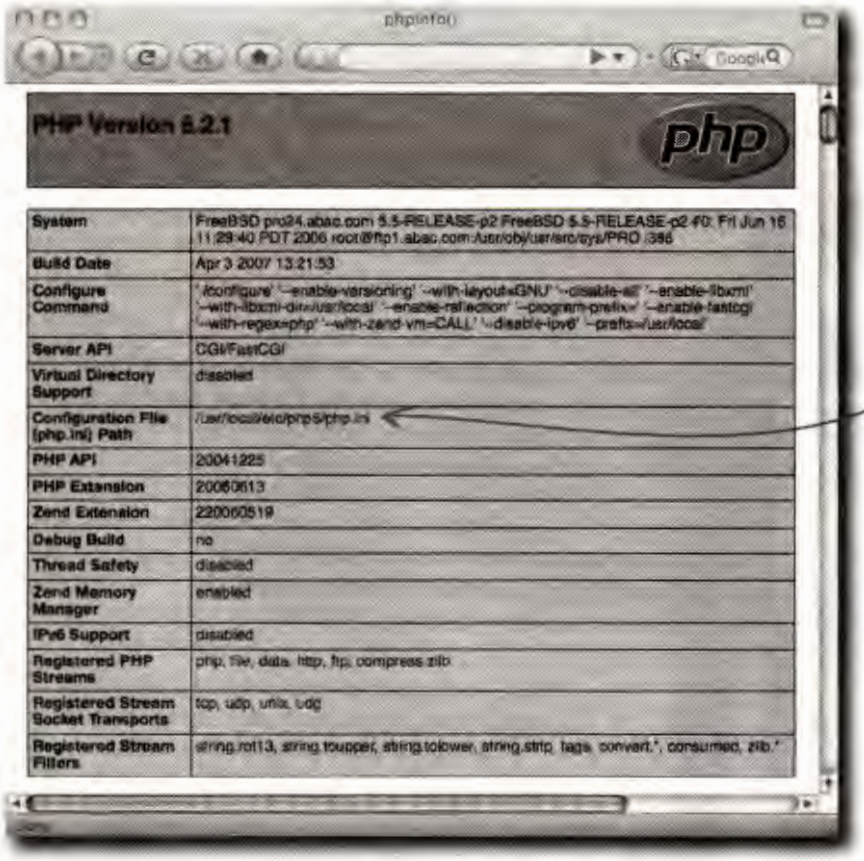
Большим недостатком в целом является то, что объектно-ориентированный код объемнее и требует больше времени для разработки. Если вам просто нужно вывести на экран слова одной песни, тогда разработка небольшой процедурной программы, возможно, будет наилучшим выбором. Но если вы считаете, что вам нужно разработать серьезное онлайновое караоке-приложение, подумайте о том, чтобы углубиться в объектно-ориентированный PHP.

№ 6. Защита вашего PHP-приложения

Существуют некоторые простые правила, следуя которым вы можете защитить свои PHP-сценарии от хакеров, склонившихся над своей клавиатурой в ожидании вашей оплошности.

Удалите ссылки на `phpinfo()`. Когда вы в первый раз начнете разработку PHP-приложения на новом веб-сервере, вы, скорее всего, создадите сценарий, содержащий функцию `phpinfo()`, чтобы определить версию используемого PHP, поддерживает ли он MySQL, а также список других установленных библиотек. Это очень хорошо проверяется с помощью `phpinfo()`, но вы должны удалить эту функцию, после того как все это просмотрели. Если вы не сделаете этого, любой хакер, узнавший о новой уязвимости PHP, сможет увидеть, чувствителен ли ваш сайт к ней.

Если вы не пользуетесь веб-хостингом и у вас есть доступ к файлу `php.ini`, существует несколько параметров, которые вы можете изменить в этом файле, чтобы увеличить защиту вашего PHP-приложения. Ирония в том, что расположение вашего файла `php.ini` может быть определено с помощью `phpinfo()`:



Здесь указан путь вашего файла php.ini. После того как вы его записали, не забудьте удалить функцию phpinfo().

Далее на этой странице есть более чувствительная информация.

№ 6. Защита вашего PHP-приложения (продолжение)

Далее следуют некоторые параметры настройки, на которые вы должны обратить внимание, внося изменения в файл `php.ini`. Откройте файл в текстовом редакторе, внесите изменения, сохраните их и перезапустите ваш веб-сервер.

```
safe_mode = On
```

Если вы включите `safe_mode`, никакой PHP-сценарий не может быть вызван никаким другим сценарием иного владельца на этом веб-сервере. Очевидно, что если вам необходимо разрешить сценариям других владельцев вызывать ваш сценарий, то вы не можете использовать такую настройку этого параметра.

```
open_basedir = directory[:...]
```

Такое значение параметра ограничивает указанным каталогом и его подкаталогами перечень сценариев и файлов, которые PHP может исполнять и к которым ему разрешен доступ.

```
expose_php = Off
```

Если этому параметру присвоено значение `On`, каждому браузеру, посещающему ваш сайт, будет послан заголовок, раскрывающий информацию о вашем PHP-сервере. Выключение этого параметра скроет эту информацию и сделает ваш сервер менее открытым, а следовательно, более защищенным.

```
display_errors = Off
```

После того как вы разработали ваше приложение и оно работает на вашем веб-сервере, больше нет необходимости видеть сообщения об ошибках. Вы, конечно, уже исправили все ошибки, но иногда они проскакивают из-за сбоев. Для того чтобы скрыть сообщения об ошибках от посетителей сайта, присвойте этому параметру значение `Off`.

```
log_errors = On
```

Это отправляет все ваши сообщения об ошибках в журнал ошибок. Это хорошее решение на тот случай, если вы захотите проверить ваше приложение на ошибки. При присвоении параметру `display_errors` значения `Off`, а `log_errors` — `On` вы будете иметь возможность видеть причины проблем, а посетители сайта — нет.

```
error_log = filename
```

Вы должны будете проверить с помощью программы, доступной на вашем веб-сервере, расположение этого файла. В него будут записываться сообщения об ошибках при значении параметра `log_errors`, равном `On`.

№ 7. Защита вашего приложения от межсайтового скриптинга

Возможно, вы слышали о межсайтовом скриптинге, который иногда называют XSS-атакой. Межсайтовый скриптинг — это атака на веб-приложение, при которой атакующий вносит в HTML-форму специальный код (URI) и перехватывает результат его обработки. Это является серьезной проблемой в PHP-программировании. Давайте посмотрим внимательно, как это происходит и как защититься от такой атаки.

Межсайтовый скриптинг обычно направлен на сайты, которые предоставляют посетителям возможность передавать на сервер какие-либо данные. Любые данные, которые вы получаете от своих посетителей, потенциально могут быть искажены, что может привести к уязвимости вашего сайта.

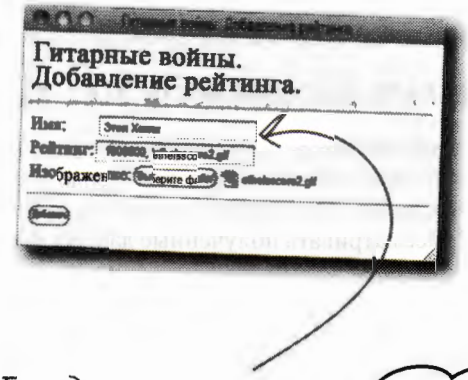
Используя XSS-атаку, хакер может доставить немало неприятностей. Одна из худших заключается в перенаправлении адреса страницы ответа на другую страницу, расположенную на сайте, контролируемом хакером, что может позволить ему запрашивать у посетителя дополнительную информацию. Посетитель может не заметить, что он уже находится не на вашем сайте, а так как он доверяет вашему порталу, то вполне способен передать конфиденциальную информацию непосредственно на атакующий сервер.

Далее показано, как это могло бы произойти на сайте «Гитарные войны».

Этел, вместо того чтобы внести свое имя в поле формы Имя, вводит код на языке JavaScript. В этом примере она использует функцию `window.location` для перенаправления браузера на свой собственный сайт. А так как она управляет своим сайтом, она может показать посетителю все что захочет, включая сайт, который выглядит как «Гитарные войны». Она может сделать что-нибудь более опасное, предлагая посетителям передать данные поважнее, чем рейтинг, например финансовую информацию.

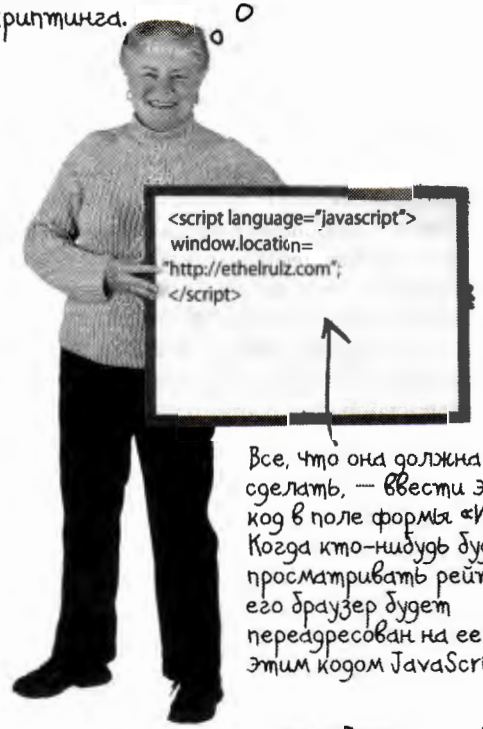
Существуют другие, еще более коварные вещи, которые она может сделать, включая кражу куки или представление посетителю окна диалога для ввода имени и пароля. Как только пользователь введет эти данные, Этел может опять вернуть посетителя на первоначальный сайт.

Так как же предотвратить XSS-атаку на ваше веб-приложение?



Если Этел не сможет мошенничать, она перенаправит страницу рейтингов на свой собственный сайт с помощью межсайтового скриптинга.

Вы думаете, вы расстроили меня? Я атакую ваш сайт, и ваш рейтинг упадет!



Все, что она должна сделать, — ввести этот код в поле формы «Имя». Когда кто-нибудь будет просматривать рейтинги, его браузер будет переадресован на ее сайт этим кодом JavaScript.

№ 7. Защита вашего приложения от межсайтового скриптинга (продолжение)

К счастью, если вы проверяете достоверность ваших данных, то уже находитесь на пути к защите вашего приложения. Вы уже узнали, как осуществить такую атаку на примере «Гитарных войн». Вот краткое руководство по защите вашего приложения.

Проверяйте достоверность всех данных

Любые данные, которые вы получаете, такие как результаты заполнения форм, должны быть проверены на достоверность с целью обнаружения хакерского кода до того, как он сможет повредить вашему приложению. Вы будете лучше защищены, если всегда станете рассматривать полученные данные как опасные до тех пор, пока не убедитесь в обратном, проверив их на достоверность.

Встроенные функции PHP могут помочь

Используйте встроенные функции PHP, такие как `strip_tags()`, чтобы «дезинфицировать» внешние данные. `Strip_tags()` — отличная функция, она удаляет из строки теги HTML. Поэтому, если вы примените `strip_tags()` к переменной Этел `$_POST['name']`, то получите следующее:

```
window.location='http://ethelrulz.com'
```

Хотя это все еще не имя, оно не сможет переадресовать браузер, потому что важные теги JavaScript удалены.

Данные опасны, пока не доказана их безопасность

Начните с наиболее жесткого испытания на достоверность, какое вы можете себе позволить, и смягчайте его только в случаях, когда это необходимо. Например, если вы начнете с разрешения ввода только цифр в поле телефонного номера и затем разрешите также дефисы и скобки, вы будете лучше защищены, чем если бы вы разрешили ввод любых алфавитно-цифровых символов с самого начала. Или в случае «Гитарных войн»: если бы вы запретили все, кроме букв в поле Имя, вы бы никогда не получили символ «менее чем» (<), который открывает злонамеренный код JavaScript, составленный Этел. Регулярные выражения (глава 10) играют огромную роль в процессе проверки данных на соответствие разрешенным критериям.

№ 8. Приоритеты операторов

Представьте строку кода:

```
$marbles = 4 / 2 - 1;
```

Это будет 1.

Значение переменной `$marbles` могло бы быть либо 4, либо 1. Мы не можем определить это непосредственно из кода, но можем принять определенные правила **приоритета**. Под приоритетом мы понимаем **порядок**, в котором выполняются операции. Операторы в PHP выполняются в определенном порядке. В примере вверху деление будет осуществляться перед вычитанием, поэтому `$marbles` будет равно 1.

В зависимости от того, в каком порядке мы хотим осуществить выполнение нашего кода, мы могли бы записать его двумя различными способами:

```
$marbles = (4 / 2) - 1;
$marbles = 4 / (2 - 1);
```

В первом выражении мы делим 4 на 2 и затем вычитаем 1. Во втором случае мы вычитаем 1 из 2 и затем делим 4 на результирующую единицу. Использование скобок позволяет нам точно управлять порядком операций. Но знание приоритета операций в PHP может помочь вам понять, что будет происходить в сложных выражениях. И, поверьте нам, это поможет вам в отладке вашего кода в случаях, если вы забыли использовать скобки.

Прежде чем мы перейдем к списку приоритетов операторов, вот еще одна причина, почему вам следует использовать скобки.

Представьте следующее:

Это будет -1.

```
$marbles = 4 - 3 - 2;
```

Здесь не применяются никакие правила приоритета. Результат будет либо 3, либо –1. Это вносит путаницу, когда вы пишете код. Поэтому лучше использовать скобки, как в следующих двух строках:

```
$marbles = 4 - (3 - 2);
$marbles = (4 - 3) - 2;
```

А теперь список операций с наивысшего приоритета (выполняются вначале) до самого низкого (выполняются в конце).

Оператор	Тип операций
`++ --`	инкрементные/декрементные
`*/ %`	арифметические
`+ - .`	арифметические и строковые
`< <= > >= <>`	сравнения
`== != === !==`	сравнения
`&&`	логические
`||`	логические
`= += -= *= /= .= %= &= |= ^= <<= >>=`	присвоения
`and`	логические
`xor`	логические
`or`	логические

Операторы сравнения, аналогичные тем, которые вы используете в управляющих конструкциях if, также имеют приоритеты.

№ 9. В чем заключается разница между PHP 5 и PHP 6

В момент написания этой книги PHP 5 являлся последней рабочей версией PHP. Но ведется работа над PHP 6, и он доступен для разработчиков здесь: `http://snaps.php.net/`.

Разница между PHP 4 и 5 значительно больше, чем между 5 и 6. Во многих отношениях PHP 6 предлагает усовершенствование объектно-ориентированных возможностей, представленных в 5. Другие изменения включают более широкую поддержку XML и Unicode.

Более широкая поддержка Unicode

Предположим, вашему приложению необходим вывод на греческом языке.

Представьте себе манипуляции, которые вы иногда должны производить со строками текста, такие как определение их длин или сортировка. Это легко на английском языке, но когда вы работаете с символами других языков, строковые операции становятся значительно сложнее.

Unicode — это набор символов и технологии их кодирования. В Unicode греческий символ, который выглядит как треугольник, имеет определенное числовое значение наряду с другими символами в других языках. Unicode — это стандарт. Это означает, что он имеет широкую поддержку большинства разработчиков. В Unicode каждому символу соответствует уникальное число независимо от языка, программы или платформы, на которой он используется. До появления 5-й версии у PHP не было реальной поддержки Unicode. PHP 6 расширил поддержку Unicode в своих функциях, а также в функциях, специально разработанных для создания и декодирования строк текста с использованием Unicode.

Усовершенствование объектно-ориентированной модели языка, поддержка XML и другие изменения

PHP 5 предлагает объектно-ориентированную модель, но все еще позволяет добавлять код, составленный в процедурном стиле. PHP 6 движется далее в объектно-ориентированное царство. Одно из наиболее крупных изменений заключается в том, что динамические функции больше не разрешено вызывать с использованием статического синтаксиса. Имеется большое количество важных изменений в том, как PHP обрабатывает свой объектно-ориентированный код, что делает его более совместимым с другими объектно-ориентированными языками, такими как C++ и Java.

Вот некоторые другие изменения.

- И программа просмотра, и редактор XML включены в PHP 6 как расширения, что упрощает работу с файлами XML.
- Опции `register_globals`, `magic_quotes` и `safe_mode` более недоступны в файле `php.ini`.
- Удалено расширение `ereg`, предоставлявшее другой способ построения регулярных выражений. К счастью, код `preg_match()`, описанный в этой книге, будет основным методом построения регулярных выражений в PHP 6.
- Добавлен тип данных 64-bit.
- Многомерные массивы получили возможность использовать оператор `foreach`.
- Версия PHP 6 — это, скорее, модифицированная и исправленная версия языка.

№ 10. Использование PHP-приложений, написанных другими

Далеко не всегда необходимо писать свой собственный код PHP с нуля. Иногда лучше всего использовать код, написанный кем-нибудь другим. Далее перечисляются популярные и очень успешные программные пакеты, базирующиеся на PHP, о которых вам следует подумать, если вы не хотите изобретать PHP-колесо. И все они распространяются свободно!

Drupal

Один из наиболее впечатляющих современных PHP-проектов. Drupal — это мощная система управления контентом, которая может быть использована для разработки практически любого сайта содержательного характера. NASA, The Onion, the Electronic Frontier Foundation и Popular Science — все они используют Drupal на своих сайтах. Он достаточно гибок, чтобы создать практически все что угодно со значительным содержанием. Посетите их сайт `http://drupal.org/`.

phpBB

Чемпион в области онлайновых систем обмена сообщениями (форумов), phpBB придерживается принципа «тише едешь — дальше будешь» при создании форума. Он исключительно гибок и едва ли может быть побежден в одном, что он делает очень хорошо, — управлении тематическими дискуссиями. Узнайте больше на `http://www.phpbb.com/`.

Coppermine Gallery

Если вы подумываете о создании галереи изображений, Coppermine Gallery — это то приложение, на которое нужно обратить внимание. В эру Flickr, Photobucket, Shutterfly и Snapfish создание собственного сайта со своими фотографиями — довольно странное занятие. Но возможность управления является большим достоинством, а если вы хотите полного контроля над своими фотографиями, посмотрите Coppermine Gallery на `http://coppermine-gallery.net/`.

WordPress

Один из хитов в блогосфере WordPress — это PHP-приложение, предоставляющее возможность создавать и поддерживать блог с минимумом трудностей. В этой области довольно жесткая конкуренция, поэтому вы, возможно, захотите провести небольшое исследование, но, пытаясь запустить блог, можете совершить ошибку, выбрав что-либо другое вместо WordPress. Вы можете загрузить его с `http://wordpress.org/`.

Потому что использовать чужой код не всегда так просто, как это кажется: иногда это требует навыков в PHP.

Многие программные пакеты PHP требуют локальной настройки, а для этого необходимы навыки в PHP. И это не все. Иногда вам понадобится только использовать небольшой компонент чьего-нибудь кода. В любом случае зная PHP, вы можете выбирать, а возможность выбора — это всегда хорошо!

Приложение II настройка среды разработки

Место, где разворачиваются события

Вам необходимо место, где вы могли бы практиковаться в PHP и MySQL и при этом не допустить уязвимости ваших данных. Очень полезно найти безопасное место для разработки своих PHP-приложений, прежде чем предоставлять к ним доступ из Интернета. Приложение ii содержит инструкции, как установить веб-, MySQL- или PHP-сервер для создания безопасного места для работы.

Создание среды разработки PHP-приложений

Прежде чем вы сможете поместить свое приложение на веб-сервер в Интернете, вам необходимо разработать его. И это не слишком удачная мысль — разрабатывать свое приложение на веб-сервере в Интернете, где каждый может видеть его. Вы вправе **установить программное обеспечение локально, что даст вам возможность создавать и тестировать ваше приложение, перед тем как вы выложите его в Интернет**.

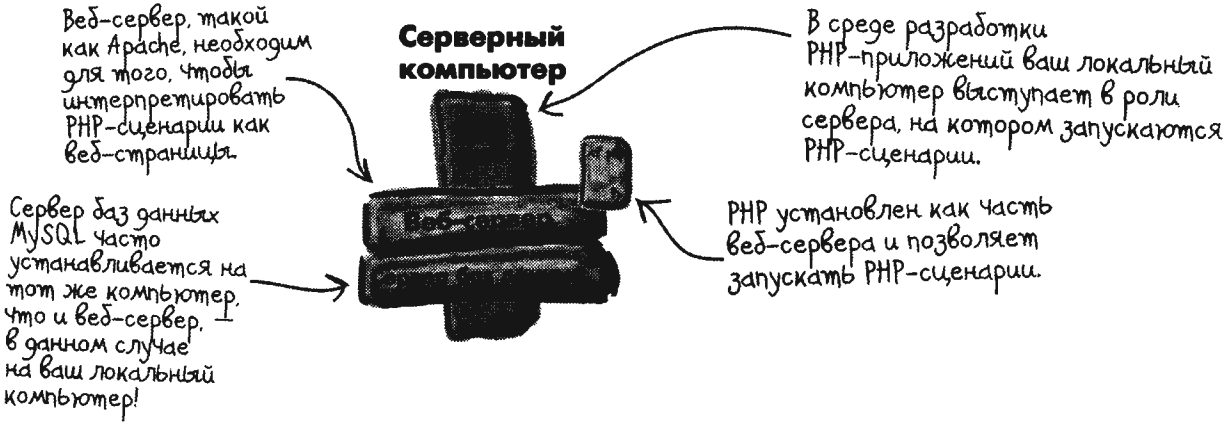
Вам необходимо установить на своем локальном компьютере три программы, чтобы создавать и тестировать ваше PHP-приложение.

1. Веб-сервер.

2. PHP.

3. Сервер баз данных MySQL.

PHP — не сервер, а набор правил, согласно которым ваш веб-сервер понимает, как интерпретировать PHP-код. И веб-, и MySQL-сервер являются исполняемыми программами, которые запускаются на вашем компьютере. Имейте в виду, что мы ведем речь о конфигурировании вашего локального компьютера как веб-сервера для разработки PHP-приложения. После окончания разработки вам будет совершенно необходим онлайновый веб-сервер, на который вы загрузите свое законченное приложение с тем, чтобы другие люди могли видеть его.



Определите, что у вас есть

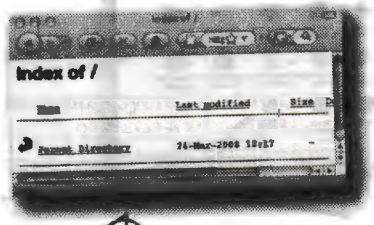
Прежде чем устанавливать любую часть комплекса для разработки PHP-приложений, вначале определите, что у вас уже установлено. Давайте посмотрим на эти три программы и на то, как вы сможете определить, что из них уже есть в вашей системе.

Платформа, используемая на вашем компьютере, сильно влияет на состав установленного программного обеспечения. Например, на Mac OS X веб-сервер устанавливается по умолчанию, в то время как на большинстве компьютеров с Windows его нет.

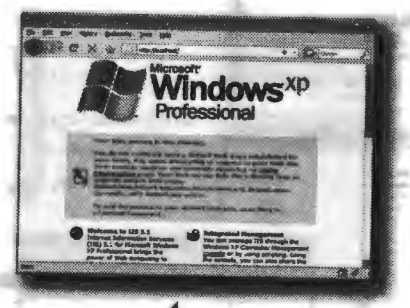
Примечание: это приложение охватывает Windows 2000, XP, Vista, Windows Server 2003/2008 или другие 32-битные операционные системы Windows. Оно применимо к Mac OS X 10.3.x и более поздним версиям.

Есть ли у вас веб-сервер?

Возможно, у вас уже есть веб-сервер, если вы используете современный PC или Mac. Чтобы быстро это узнать на обеих системах, откройте браузер и введите `http://localhost` в адресную строку. Если откроется вступительная страница, то это означает, что на вашем компьютере имеется работоспособный веб-сервер.



Если у вас Mac или Windows с установленным веб-сервером Apache, вы увидите что-то подобное.



Если у вас Windows с IIS, вы увидите что-то подобное.

У вас есть PHP? Какая версия?

Если у вас есть веб-сервер, вы можете очень легко определить, установлен ли PHP и какой версии. Создайте сценарий с именем `info.php` со следующим кодом:

```
<?php phpinfo() ?>
```

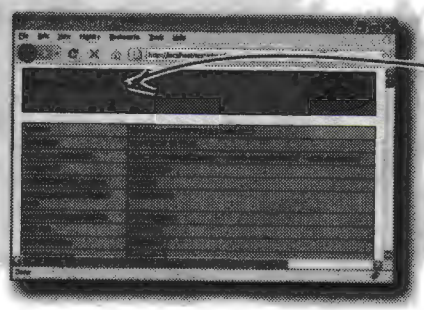
Сохраните этот файл в каталоге, который использует ваш веб-сервер. В Windows это обычно:

```
C: inetpub/wwwroot/
```

На Mac это что-нибудь вроде:

```
/Users/yourname/sites/
```

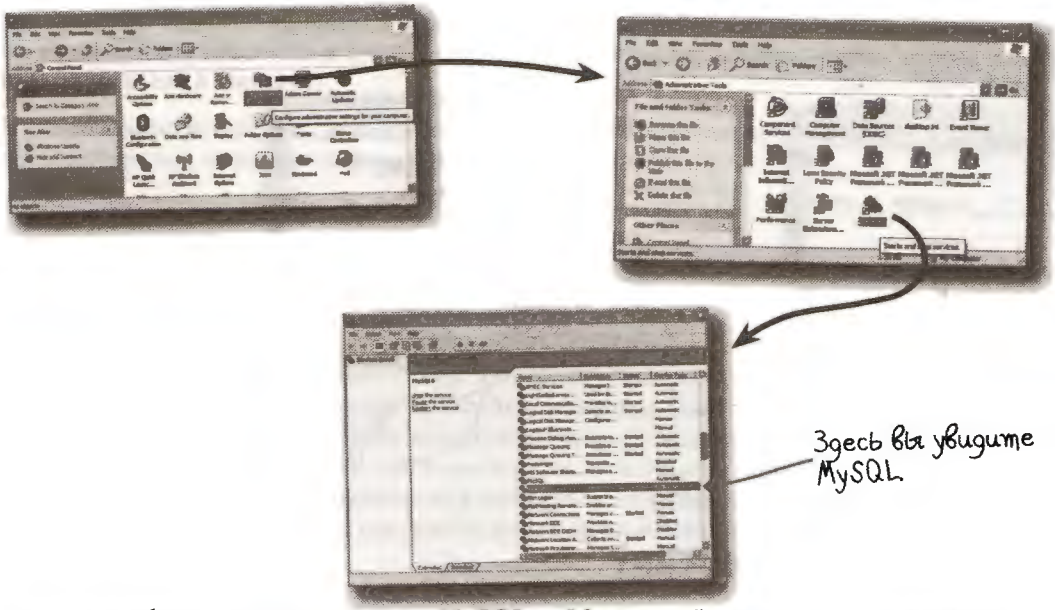
Если вы попытаетесь открыть этот файл в своем браузере, введя адрес `http://localhost/info.php`, и если у вас установлен PHP, вы увидите что-то вроде:



Это версия установленного PHP.

У вас есть MySQL? Какая версия?

В Windows вы можете узнать это, открыв `Панель управления -> Администрирование -> Службы`.



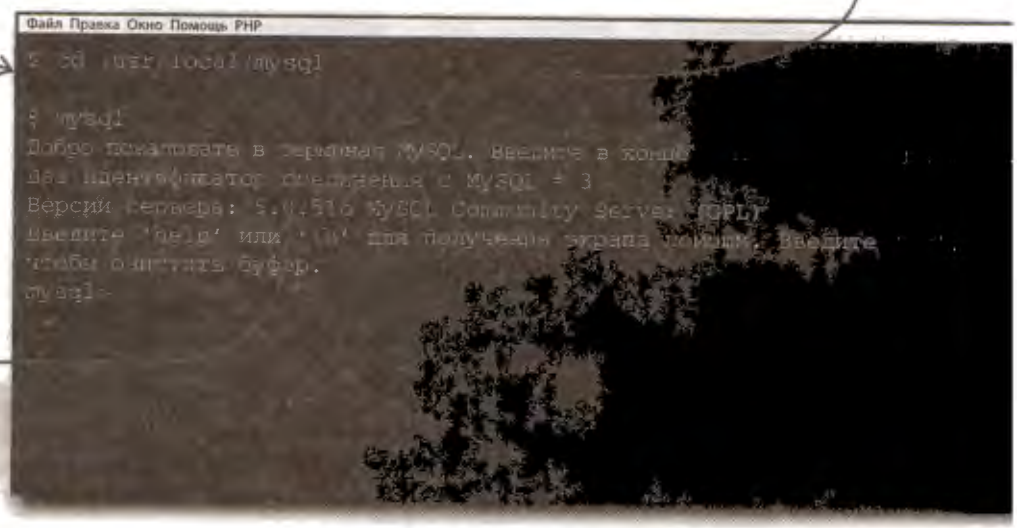
Для того чтобы узнать, установлен ли MySQL на Mac, откройте ваш терминал и введите:

```
cd /user/local/mysql
```

Если команда будет выполнена, это означает, что у вас установлен MySQL.
Для определения версии введите:

```
mysql
```

Терминал MySQL также известен под именем «монитор MySQL».



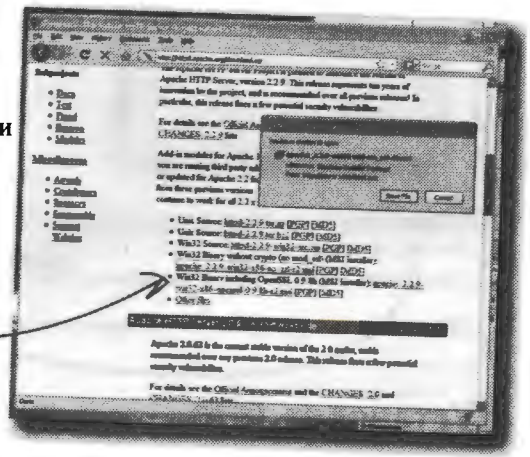
Начало установки веб-сервера

В зависимости от вашей версии Windows вы можете установить Microsoft Internet Information Server (IIS) или свободно распространяемый веб-сервер Apache. Если вам нужен сервер на Mac, скорее всего, вам лучше использовать Apache, так как он уже установлен.

Далее следует краткое описание процесса установки Apache на Windows.

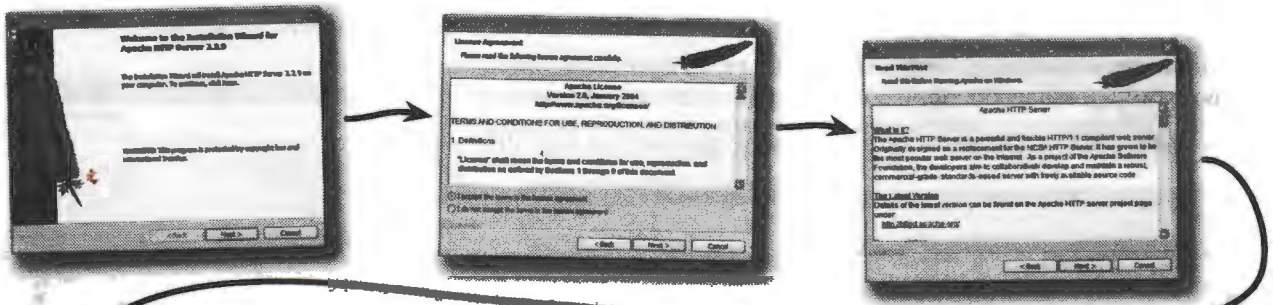
Отправляйтесь на `http://httpd.apache.org/download.cgi`.

Если вы используете Windows, мы предлагаем вам загрузить файл `apache_2.2.9-win32-x86-no_ssl-r2.msi`. **Он автоматически установит Apache, после того как вы загрузите его и дважды щелкнете нем.**

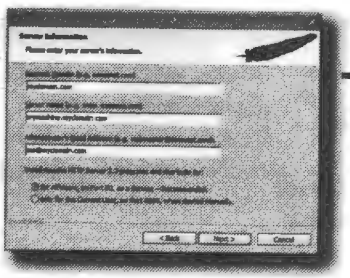


Загрузите эту версию файла и дважды щелкните на нем

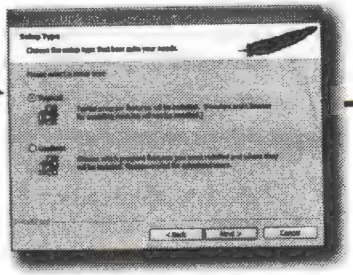
Затем вы увидите окно программы установки. Большинство инструкций понятны, и вы можете принять значения по умолчанию.



Выберите домен, которому принадлежит ваш компьютер. Если у вас нет домена, вы можете ввести `localhost`.



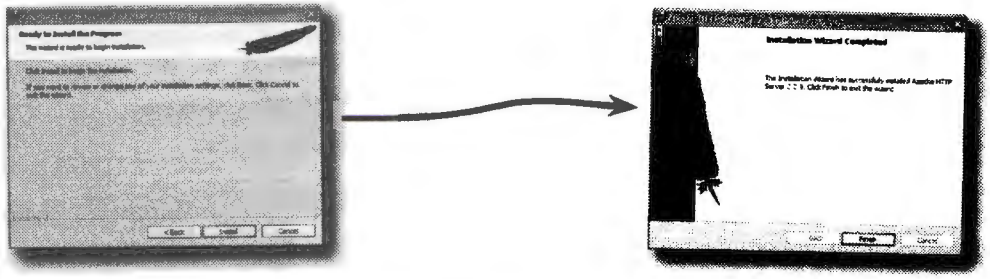
Типичный вариант установки — наилучший выбор.



Вы можете выбрать каталог по умолчанию для установки программы.

Установка Apache... завершение

Вы практически закончили. Щелкните `Install` и подождите минуту-другую для завершения установки. Вот и все!



Ваш веб-сервер установлен таким образом, что он запустится автоматически при включении компьютера. Но вы можете изменить это, останавливая сервер и запуская его вновь с использованием соответствующего диалога панели:

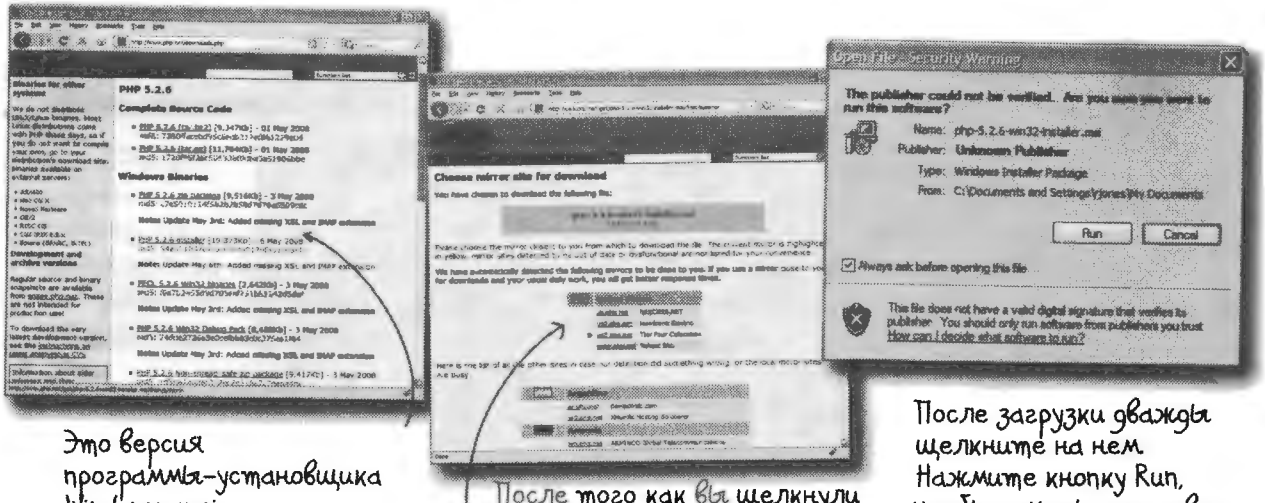
`Панель управления -> Администрирование -> Службы.`

Установка PHP

Откройте страницу по адресу: `http://www.php.net/downloads.php`.

Так же как и для Apache, если вы используете Windows, мы предлагаем загрузить программу-установщик `php_5.2.6-win32-installer.msi`.

После того как вы загрузите файл и дважды щелкнете на нем, PHP установится автоматически.



Это версия программы-установщика Windows .msi.

После того как вы щелкнули на файле, выберите ближайший сервер, на котором он размещен, и загрузите его.

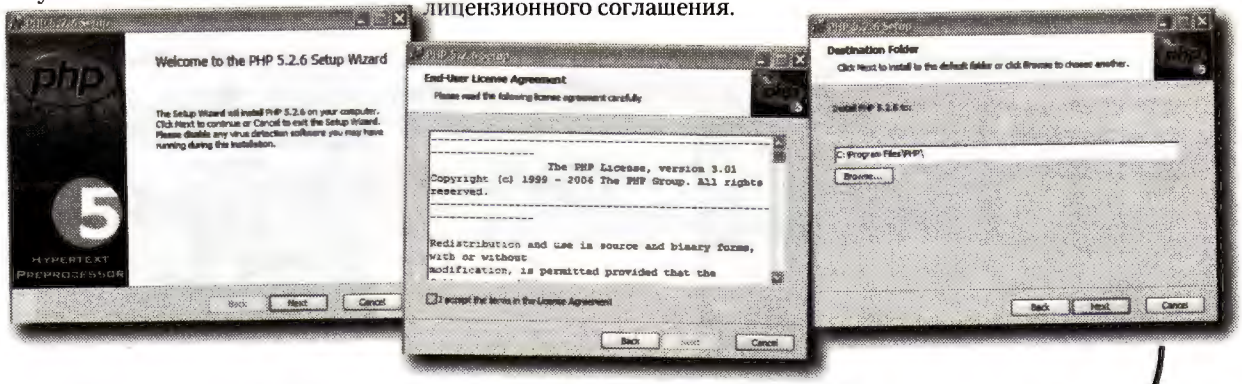
После загрузки дважды щелкните на нем. Нажмите кнопку Run, чтобы начать установку.

Стадии установки PHP

Начало основной установки.

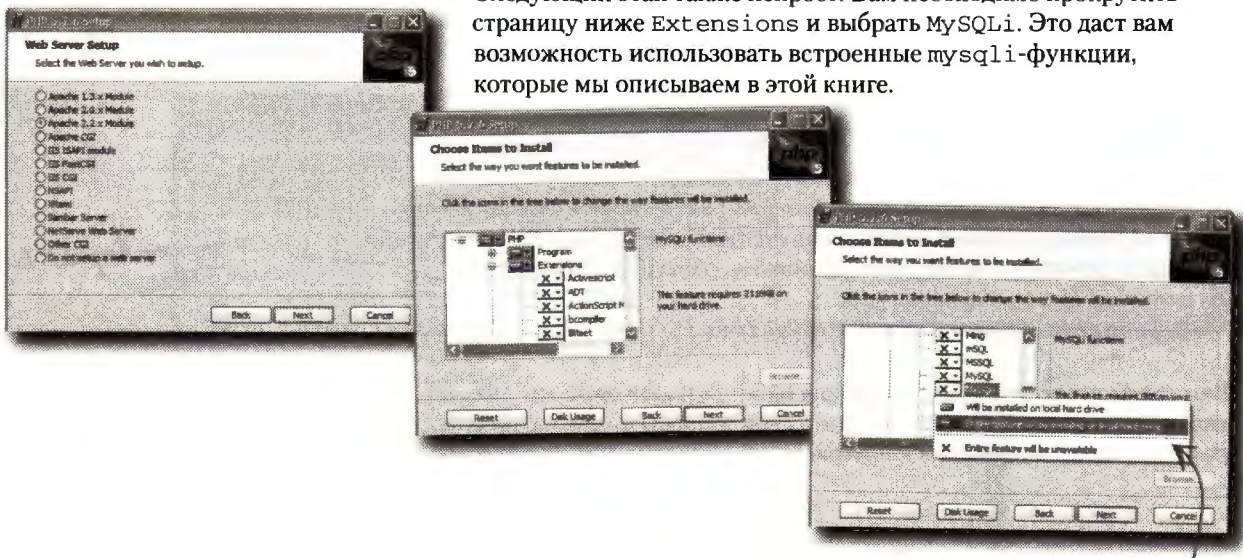
Для продолжения примите условия лицензионного соглашения.

Выбор каталога по умолчанию, в который будет установлена программа, — это хорошая мысль.



Будьте внимательны на этом этапе. Если вы используете Apache, выберите правильную версию. Если же ISS, то, скорее всего, вам нужно выбрать модуль ISSAPI. Проверьте, какое программное обеспечение у вас установлено, чтобы точно определить, что вам необходимо.

Следующий этап также непрост. Вам необходимо прокрутить страницу ниже `Extensions` и выбрать `MySQLi`. Это даст вам возможность использовать встроенные `mysqli`-функции, которые мы описываем в этой книге.

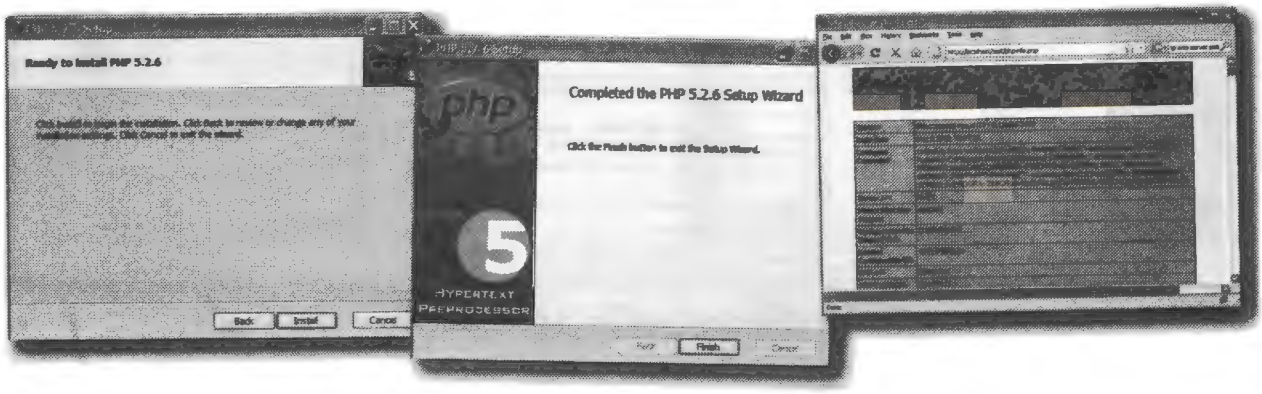


Прокрутите страницу ниже Extensions и щелкните на MySQLi. Выберите Entire feature.

Стадии установки PHP... завершение

Вот и все! Нажмите кнопку Install, а затем Done, чтобы закрыть программу-установщик.

А теперь попробуйте открыть страницу `http://localhost/info.php` и посмотрите, какая будет показана версия.



Установка MySQL

Инструкции и исправление ошибок

Еще вам необходим сервер MySQL, поэтому давайте загрузим и установим его. Официальное наименование свободно распространяемой версии сервера MySQL RDBMS — это ***MySQL Community Server***.

Далее следует перечень этапов по установке MySQL на Windows и Mac OS X. Это не является заменой превосходных инструкций, размещенных на сайте MySQL, и ***мы очень рекомендуем вам посетить этот сайт и прочитать их***! Для получения более подробных инструкций, а также руководства по устранению ошибок обратитесь по адресу:

Смотрите версию 6.0 или новее.

```
http://dev.mysql.com/doc/refman/6.0/en/windows-installation.html
```

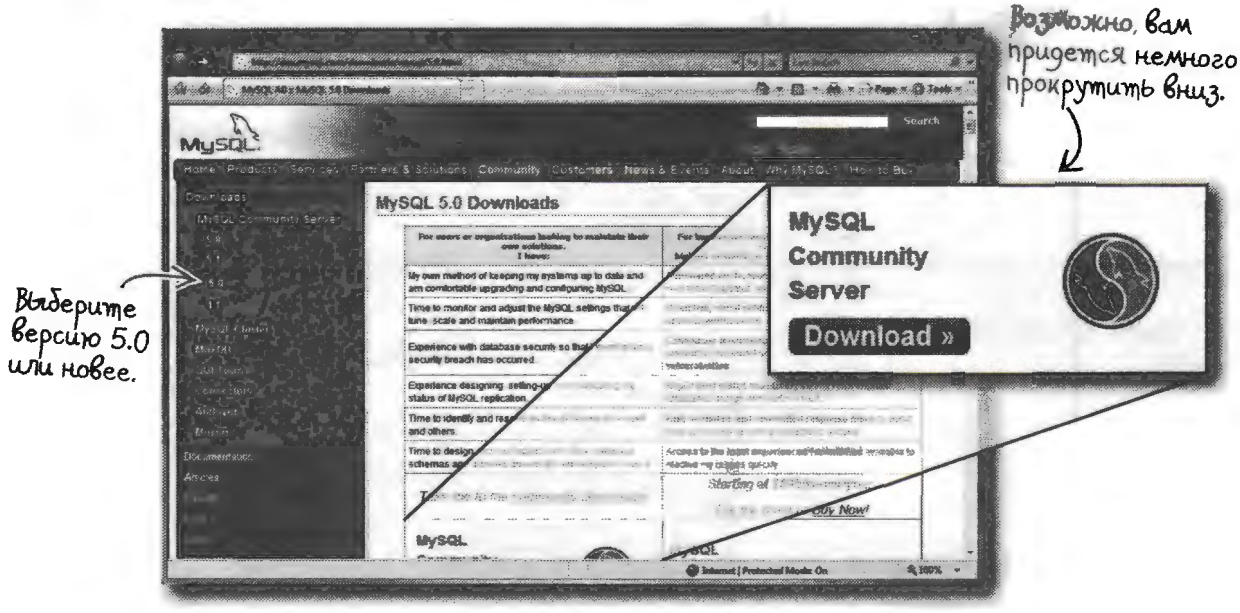
Вам также понравится браузер запросов MySQL, который мы уже упоминали. В нем вы можете вводить свои запросы и просматривать результаты их выполнения в программной оболочке, а не на консоли.

Стадии установки MySQL на Windows

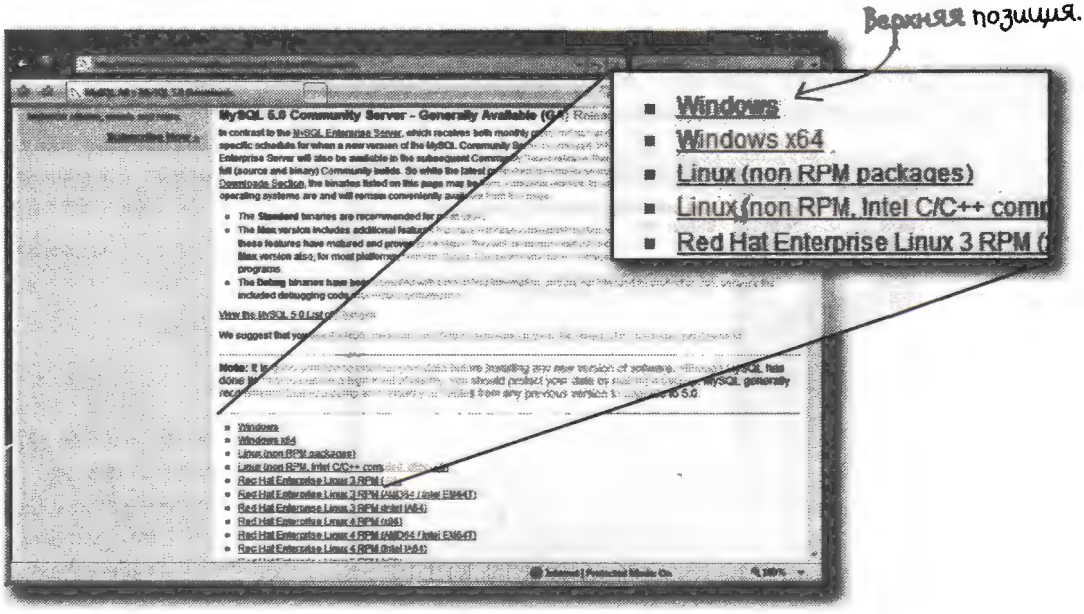
1. Откройте страницу по адресу:

   `http://dev.mysql.com/downloads/mysql/6.0.html`

   и нажмите кнопку MySQL Community Server.

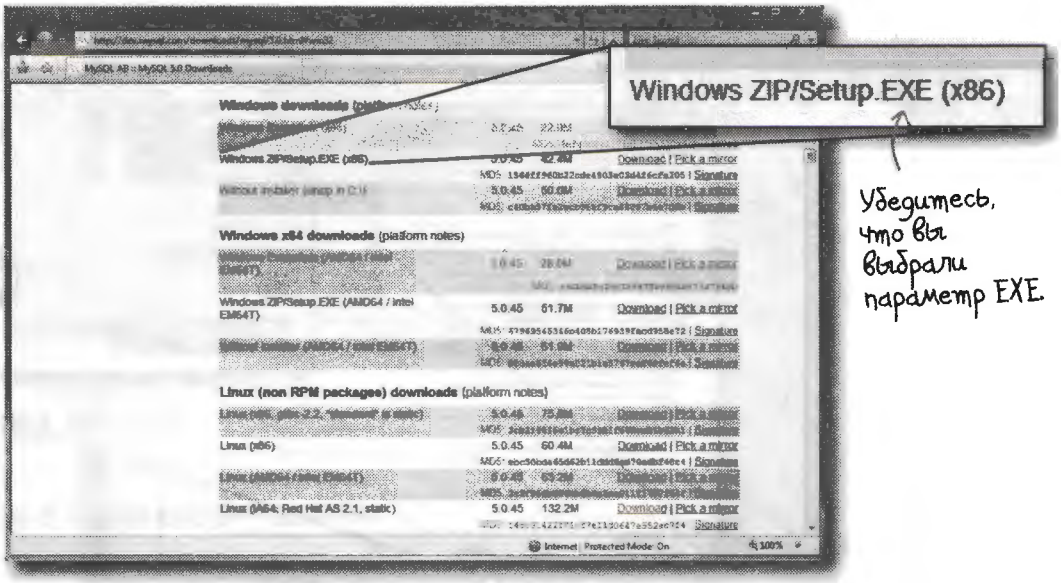


2. Выберите в списке ***Windows***.



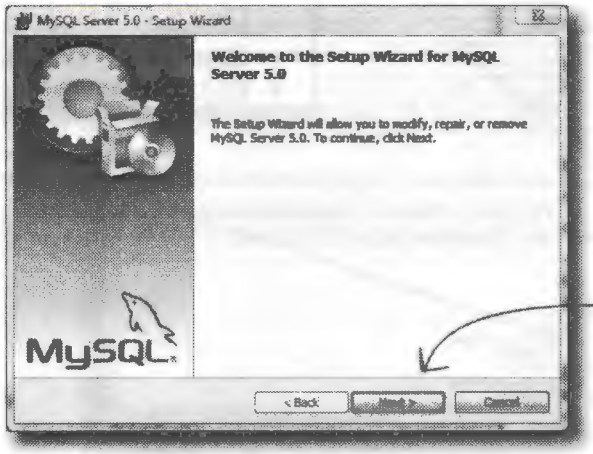
Загрузка программы-установщика

3. В секции `Windows downloads` мы рекомендуем вам выбрать опцию `Windows ZIP/Setup.EXE`, потому что в этом случае вы получаете программу, которая значительно упрощает установку. Щелкните на `Pick a Mirror`.



4. Вы увидите перечень сайтов, на которых размещены копии, доступные для загрузки. Выберите сайт, ближайший к вам.

5. По окончании загрузки дважды щелкните на файле, чтобы запустить программу установки. С этого момента программа **(Setup Wizard)** будет вести вас по процессу установки. Нажмите кнопку `Next`.



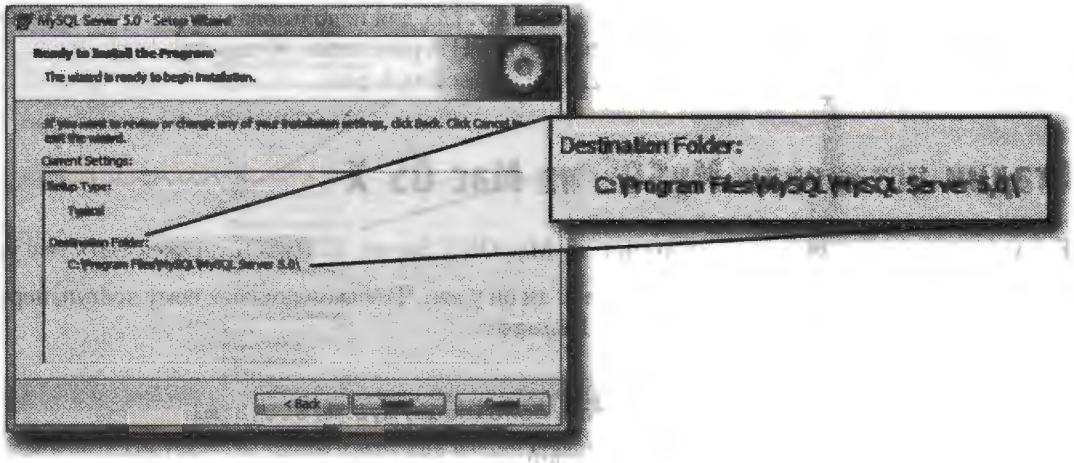
Выбор каталога, в который будет установлен MySQL

6 Вам будет предложено выбрать режим установки: **Typical**, **Complete** или **Custom**. Для решения задач, изложенных в этой книге, выберите **Typical**.

Вы можете изменить каталог, в который будет установлена программа, но мы рекомендуем использовать каталог по умолчанию:

```
C:\Program Files\MySQL\MySQL Server 6.0.
```

Нажмите кнопку `Next`.



Нажмите Install — и на этом все!

7 Вы увидите диалог **Ready to Install**, где в позиции **Destination Folder** будет указан каталог, в который устанавливается MySQL. Если он вас устраивает, нажмите `Install`. В противном случае нажмите `Back`, измените каталог и вернитесь назад.

Нажмите `Install`.

Активация PHP на Mac OS X

PHP включен в состав операционной системы Mac OS X версий 10.5+ (Leopard), но он не активирован по умолчанию. Вам необходимо открыть главный конфигурационный файл Apache и удалить символ комментария в начале строки, содержащей код активации PHP. Имя этого скрытого файла `http.conf`, он находится в каталоге, в котором размещен Apache.

Найдите следующую строку кода с символом # в начале:

```
#LoadModule php5_module        libexec/apache2/libphp5.so.
```

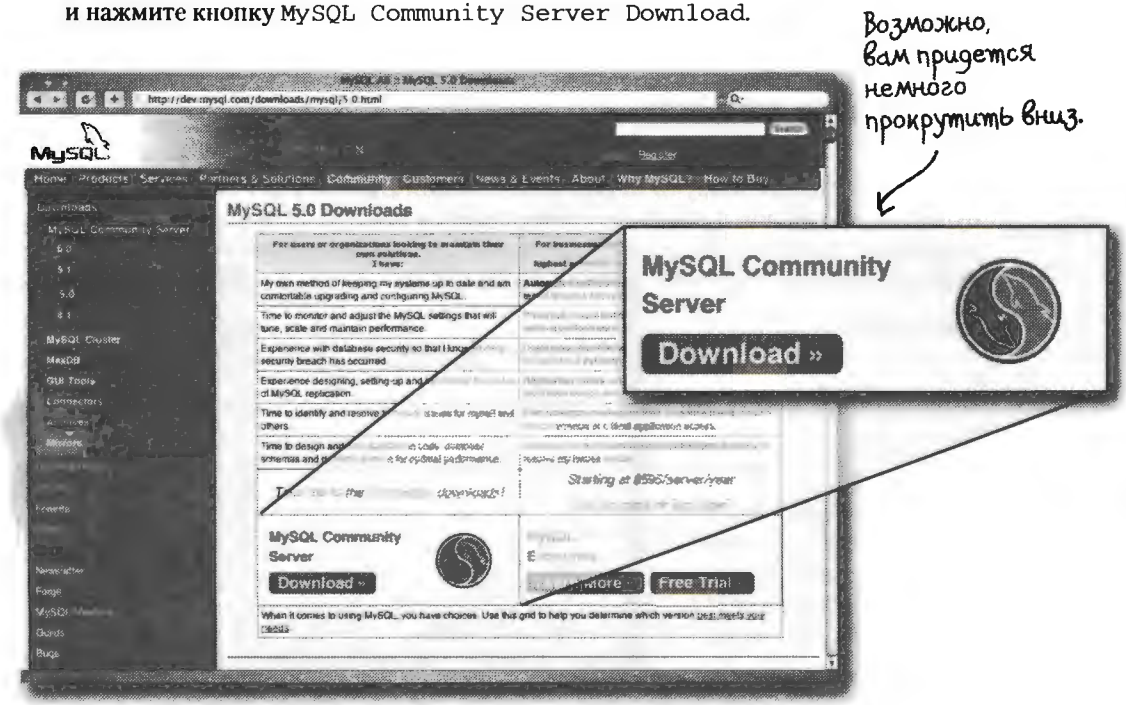
Вам необходимо удалить символ # и перезапустить сервер, чтобы активировать PHP. Владельцем файла `http.conf` является root, а это означает, что вам необходимо ввести его пароль, чтобы изменить этот файл. Возможно, вы также захотите подстроить файл `php.ini` для использования его Apache. Более подробно о том, как выполнять эти действия и активировать PHP, рассказано по адресу: `http://foundationphp.com/tutorials/php_leopard.php`.

Стадии установки MySQL на Mac OS X

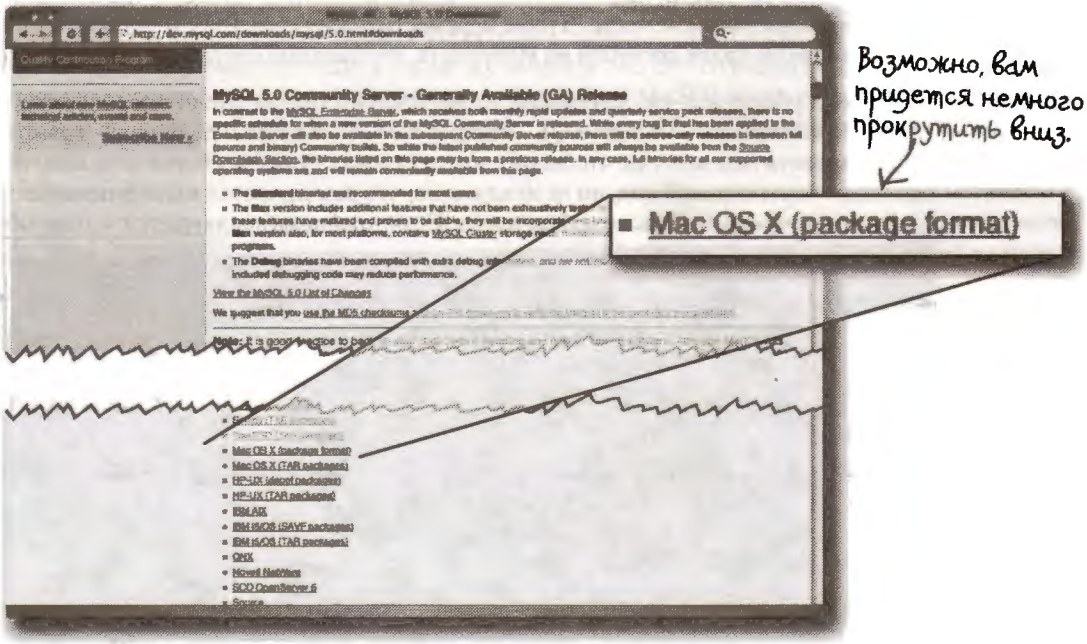
Если вы используете операционную систему Mac OS X Server, то MySQL должен быть уже установлен.

Прежде чем начинать, проверьте, установлен ли он у вас. Чтобы получить к нему доступ, перейдите в каталог **`Application/Server/MySQL Manager`**.

1. Откройте страницу по адресу:
   `http://dev.mysql.com/downloads/mysql/6.0.html`
   и нажмите кнопку `MySQL Community Server Download`.



2. Выберите в списке `Mac OS X (package format)`.



3. Выберите соответствующий пакет для вашей версии Mac OS X. Щелкните на `Pick a Mirror`.

4. Вы увидите перечень сайтов, на которых размещены копии (вы можете их загрузить). Выберите сайт, расположенный близко к вам.

5. По окончании загрузки дважды щелкните на файле, чтобы запустить программу установки. Теперь вы можете открыть окно терминала и ввести следующие команды:

```
shell> cd /usr/local/mysql
shell> sudo ./bin/mysqld_safe
```

(Введите пароль при необходимости.)
(Нажмите `Ctrl+Z`.)

```
shell> bg
```

(Нажмите `Ctrl+D` или введите команду `exit`, чтобы выйти из терминала.)
Если вы используете графические интерфейсы, такие как `phpMyAdmin`, обращайтесь к документации на это программное обеспечение, чтобы понять, как получить доступ к MySQL после успешной его установки.

Перенесите приложение с локального компьютера на постоянное место, где оно будет доступно из Интернета

Вы потратили дни или даже недели, разрабатывая ваш сайт, и наконец чувствуете, что он готов к размещению на своем постоянном месте. Чтобы переместить ваш сайт PHP и MySQL с компьютера на постоянное место, где он будет доступен из Интернета, необходимо составить небольшой план.

Для начала вы должны убедиться, что на сервере, на который вы предполагаете переместить свой сайт, установлены те же версии PHP и MySQL, которые вам необходимы. Если нет — возможно, вам придется модифицировать ваш код так, чтобы он соответствовал существующим реалиям. В этой книге в основном приведен переносимый код, но не исключено, что вам придется модифицировать его, заменив используемые функции `mysqli` на их `mysql`-аналоги. Если проблема заключается в этом, обратитесь к пункту № 1 в Приложении I за подробностями.

Если программное обеспечение на сервере, на который вы предполагаете переместить свой сайт, совместимо с вашим приложением, процесс такого перемещения достаточно прост. Ниже следуют его этапы.

Вы должны будете загрузить по FTP ваши файлы в веб-каталог сервера в Интернете.

1. Загрузите PHP-файлы с вашего локального сервера в веб-каталог на сервере в Интернете. Следите за тем, чтобы не нарушить структуру файлов, и убедитесь, что вы не потеряли ни одного каталога с дополнительными файлами.

2. Сделайте дамп вашей базы данных (мы покажем, как это делать, чуть позднее), чтобы получить MySQL-запросы, которые будут вам необходимы для создания ваших таблиц, а также `INSERT`-запросы, которые нужны для переноса данных из таблиц вашего локального сервера в таблицы сервера в Интернете.

Вам необходимо получить структуру таблиц и данные, содержащиеся в них.

3. Подключитесь к базе данных на сервере в Интернете, где вы сможете выполнить MySQL-запросы `CREATE` и `INSERT`, чтобы перенести данные из таблиц вашего локального сервера в таблицы сервера в Интернете.

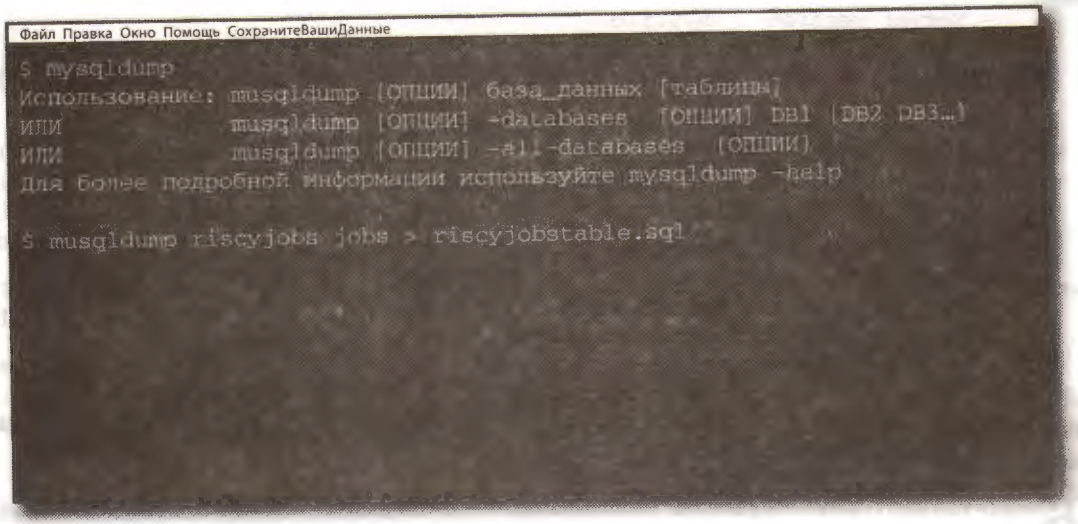
Ваш SQL-дамп предоставит вам запросы CREATE TABLE и INSERT.

4. Измените в ваших PHP-файлах код, содержащий данные по подключению к базе данных, таким образом, чтобы он указывал на сервер в Интернете. В противном случае PHP-код будет безуспешно пытаться связаться с базой данных вашего локального компьютера.

Измените вызовы mysqli_connect(), чтобы они указывали на MySQL-сервер в Интернете, связанный с вашим веб-сервером. Укажите также правильные имя пользователя и его пароль, необходимые для соединения.

Создайте дамп ваших данных (и таблиц)

Вы загрузили PHP-файлы на веб-сервер в Интернете, но ваши данные все еще не на MySQL-сервере в Интернете. Если ваши таблицы полны данных, мысль о переносе их на другой MySQL-сервер выглядит довольно опасной. К счастью, в MySQL имеется программа **MySQLdump**, которая предоставляет вам простую возможность пересоздать запросы `CREATE TABLE`, с помощью которых вы пересоздадите все свои таблицы, а также все запросы `INSERT` со всеми данными в ваших таблицах. Вам просто необходимо использовать программу MySQLdump. Для того чтобы создать копию данных, которые вы сможете перенести на другой MySQL-сервер, введите следующие команды в вашем терминале:



По этой команде будет создан и отправлен в текстовый файл `riskyjobstable.sql` запрос `CREATE TABLE` для таблицы `jobs`. Если вы опустите часть `>riskyjobstable.sql`, то все запросы `CREATE TABLE` и `INSERT` просто пролетят в окне вашего терминала. Попробуйте, чтобы увидеть, что мы имеем в виду. Пользы от этого немного, но вы увидите ваши данные, пролетающие стройными рядами перед глазами.

Раз вы отправили все ваши данные в новый файл с помощью символа «более, чем», вы можете взять этот файл и использовать его содержимое как MySQL-запросы на ваш сайт в Интернете, чтобы перенести таблицы и данные.

Подготовка к использованию дампа ваших данных

Для подготовки к переносу ваших данных на сервер в Интернете выполните на нем запрос `CREATE DATABASE`. Затем выполните запрос `USE DATABASE` в вашей новой базе. Теперь вы готовы к переносу данных с локального сервера на сервер в Интернете.

Перенесите данные на сервер в Интернете, используя дамп

Вы создали файл под названием `riskyjobstable.sql`, содержащий MySQL-запросы, создающие ваши таблицы и добавляющие в них данные. Файл `riskyjobstable.sql`, скорее всего, выглядит так:

Mysqldump всегда создает DROP-запрос, перед тем как создавать запросы CREATE и INSERT.

```
-- MySQL dump 10.11
--
-- Host: localhost    Database: riskyjobs
-- ------------------------------------------------------
-- Server version    5.0.51b

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT
*/;
--
-- Table structure for table `jobs`
--

DROP TABLE IF EXISTS `jobs`;
CREATE TABLE `jobs` (
  `job_id` int(11) NOT NULL auto_increment,
  `title` varchar(200) default NULL,
  `description` blob,
  `city` varchar(30) default NULL,
  `state` char(2) default NULL,
  `zip` char(5) default NULL,
  `co_id` int(11) default NULL,
  PRIMARY KEY  (`job_id`)
) ENGINE=MyISAM AUTO_INCREMENT=14 DEFAULT CHARSET=utf8;

--
-- Dumping data for table `jobs`
--

LOCK TABLES `riskyjobs` WRITE;
/*!40000 ALTER TABLE `riskyjobs` DISABLE KEYS */;
INSERT INTO `riskyjobs` VALUES (8,'Custard Walker','Мы ищем
добровольцев, желающих проверить теорию, утверждающую, что
по заварному крему можно ходить. \r\n\r\nМы предполагаем
заполнить плавательный бассейн заварным кремом и предложим
вам походить по его поверхности. \r\n\r\n Заварной крем
и некоторые другие жидкости относятся к классу неньютоновских
жидкостей. При воздействии на эти жидкости высокого
давления (возникающего под вашими ногами во время ходьбы)
их вязкость возрастает в значительных пределах, но при этом
они не перестают оставаться жидкостями. \r\n\r\nПолотенце
предоставляется. Необходим собственный купальный костюм.
\r\n\r\n Примечание: Если вы не будете двигаться в течение
длительного времени, то начнете медленно погружаться в крем.
Мы не несем никакой ответственности за возможные последствия;
```

Если вы знаете, что таблица jobs не существует, то можете игнорировать DROP-запрос.

Mysqldump создает единый запрос INSERT, который добавляет все строки в таблицу.

Это запрос CREATE TABLE.

Вы можете игнорировать запрос LOCK, скопировать и вставить (через буфер обмена) текст дампа, начиная с запроса INSERT.

Функция mysqldump создает по одному запросу INSERT для добавления к таблице каждой записи.

Скопируйте весь текст файла .sql и вставьте его в ваш MySQL-терминал или в окно запросов вашего графического MySQL-клиента (например, phpMyAdmin).

В результате запросы из файла будут исполнены. В примере на этой странице файл дампа содержит запросы CREATE TABLE и INSERT. В процессе выполнения файл говорит вашему MySQL-серверу удалить любую существующую таблицу и запереть (LOCK, то есть не позволить никому использовать) таблицу, пока в нее добавляются новые данные.

Подключитесь к вашему серверу в Интернете

Вы переместили ваши PHP-файлы на сайт в Интернете. С помощью MySQLdump вы получили структуру вашей таблицы в виде запроса CREATE TABLE и массив ваших данных в виде запроса INSERT, а также исполнили эти запросы на сервере в Интернете. Таким образом, ваши данные перемещены.

Остался один маленький шаг. PHP-код на вашем сайте не позволяет соединиться с MySQL-сервером в Интернете.

Вам необходимо изменить строку с параметрами соединения в вашей функции `mysqli_connect()`, чтобы эти параметры указывали на MySQL-сервер в Интернете. Везде в PHP-коде, где вы вызываете функцию `mysqli_connect()`, вы должны сделать такие изменения.

```
$dbc = mysqli_connect('localhost', 'myusername', 'mypassword', 'mydatabase')
  or die('Error connecting to MySQL server.');
```

Здесь должен быть IP-адрес вашего сайта в Интернете. Localhost будет только в том случае, если ваш MySQL-сервер размещен на том же компьютере, что и ваши PHP-файлы.

Здесь должно быть имя базы данных, которую вы создали на вашем сервере в Интернете.

А это имя пользователя и его пароль, позволяющие вам соединяться с MySQL-сервером в Интернете.

Готово!

- Вы загрузили ваши PHP-файлы на веб-сервер.
- Вы создали дамп ваших данных и сохранили их в файле `.sql`.
- Вы исполнили запросы из файла `.sql` на вашем MySQL-сервере в Интернете и изменили ваш PHP-файл, чтобы была возможность соединяться с базой данных на MySQL-сервере в Интернете.

Теперь ваш сайт должен быть виден в Интернете!

Приложение III расширьте возможности своего PHP

Добейтесь еще большего

Да, вы можете создавать отличные приложения с использованием PHP и MySQL. Но вы знаете, что должно быть что-то, что позволит делать это еще лучше. В этом коротком приложении мы покажем вам, как установить такие расширения языка, как mysqli и графическая библиотека GD. Затем мы упомянем еще несколько расширений PHP, которые вас, возможно, также заинтересуют. Потому что иногда хорошо хотеть большего.

Расширение возможностей вашего PHP

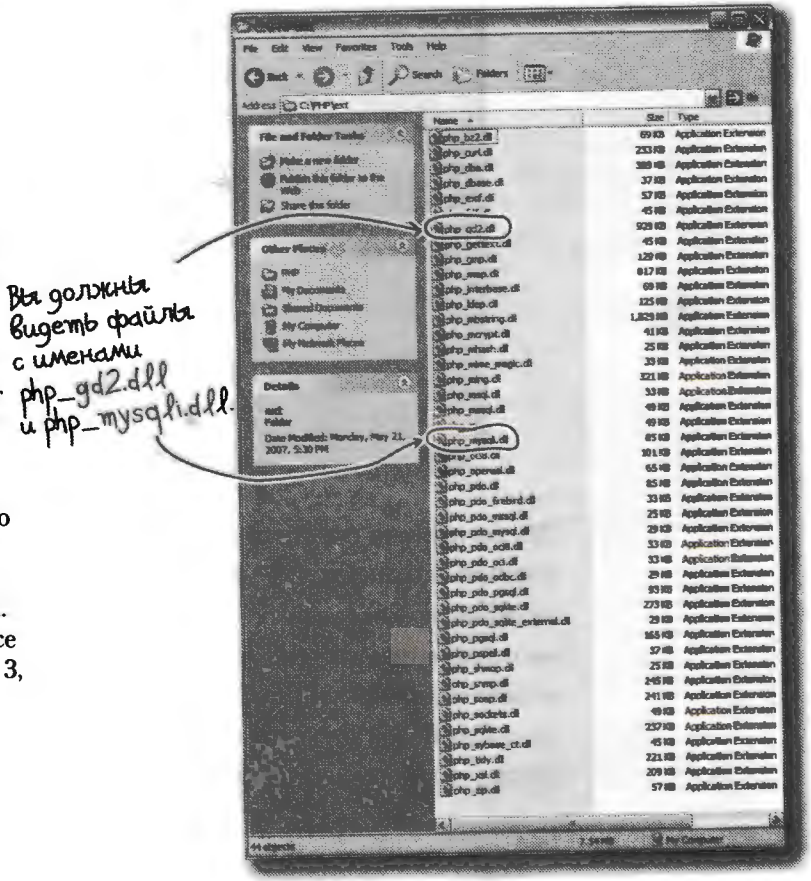
В этой книге обсуждается установка и модуля `mysqli`, и графической библиотеки GD на компьютеры под управлением операционной системы Windows. В этом приложении мы покажем вам, как узнать, какие модули имеются в вашем распоряжении, где достать эти расширения в том случае, если они у вас отсутствуют, и как установить их на компьютеры под управлением операционной системы Windows. К сожалению, установка этих расширений на компьютеры под управлением операционных систем Mac или Linux — процесс немного более сложный. Подробней об этом в конце данного приложения.

Примечание: это приложение охватывает Windows 2000, XP, Vista, Windows Server 2003/2008 или другие 32-битные операционные системы Windows.

Если вы используете Windows, ваша задача упрощается

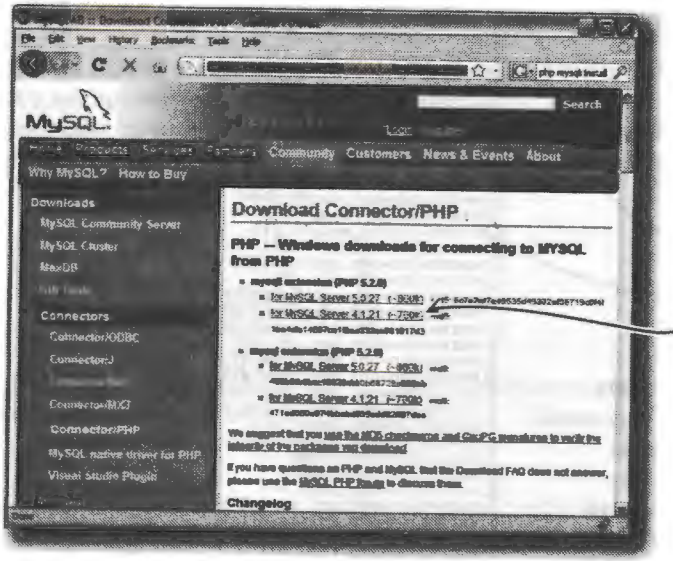
Возможно, оба расширения (и `mysqli`, и графическая библиотека GD) уже установлены на вашем компьютере. И даже если нет, их добавление осуществляется довольно просто. Мы покажем вам, как проверить, что у вас установлено, и, если одно из этих расширений или оба отсутствуют, как получить и активировать их.

1. Вначале определим, имеются ли модуль `mysqli` и графическая библиотека GD на вашем компьютере. Для того чтобы сделать это, откройте каталог, в котором установлены расширения PHP. Обычно это каталог с именем `C:/PHP/ext`, хотя на вашем компьютере может быть и другое имя. Найдите в этом каталоге файлы с именами `php_gd2.dll` и `php_mysqli.dll`. Обычно они уже установлены в PHP версии 5 и более поздних и требуют только активации. Если они у вас имеются — все отлично, переходите к этапу 3, если нет — к этапу 2.



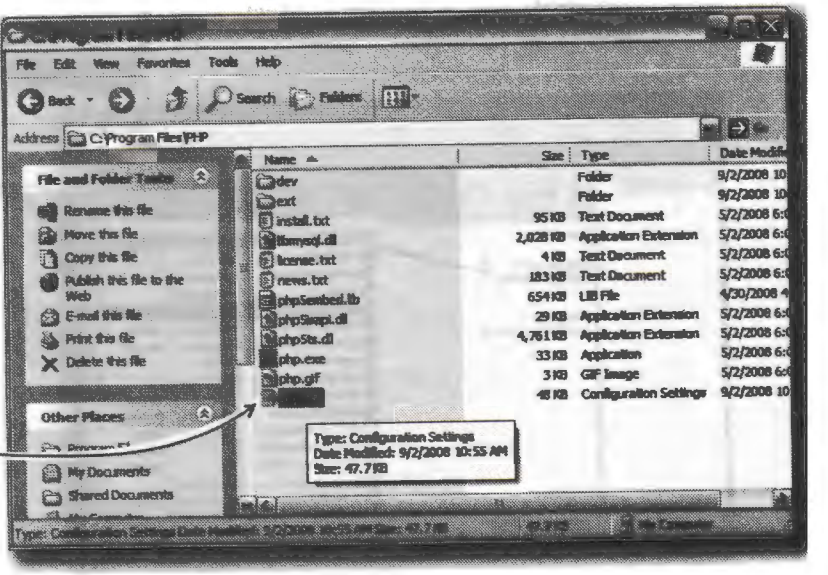
2 Если у вас отсутствует либо файл `php_gd2.dll`, либо `php_mysqli.dll`, вам необходимо получить их. Не исключено, что они у вас имеются, но если все же нет, вы можете загрузить файл `php_gd2.dll` с сайта `http://www.libgd.org/Downloads`. Загрузите файл на свой компьютер и скопируйте его в каталог `ext` той папки, в которую установлен PHP. В нашем примере это `C:/PHP/ext`.

Модуль `mysqli` вы можете загрузить с сайта `MySQL.com`. Вначале откройте страницу `http://www.mysql.com`. Нажмите ***Download (вверху) –> Connectors (в меню слева) –> MySQL native drivers for PHP –> Download php_mysqli.dll*** для PHP 5.2.1 (Windows). Убедитесь, что это ваша версия.



Выберите версию mysqli, соответствующую версии установленного у вас PHP.

3 К этому моменту вы должны иметь в каталоге `C:/PHP/ext` файлы `php_gd2.dll` и `php_mysqli.dll`. Нам необходимо указать в файле конфигурации `php.ini`, что PHP должен использовать эти модули. Для того чтобы сделать это, перейдите в каталог, в котором размещен этот файл, и откройте его в текстовом редакторе.



Иногда программа установки устанавливает PHP в каталог C:\Program Files\PHP. Найдите ваш файл php.ini и откройте его в текстовом редакторе.

4 Найдите в файле `php.ini` строку

```
extension=php_gd2.dll
```

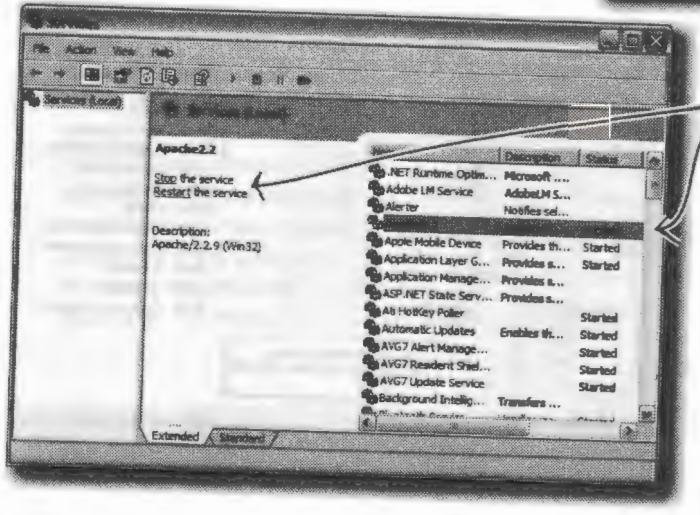
и строку

```
extension=php_mysqli.dll
```

Если в начале любой из этих строк имеется символ точки с запятой (;) или хеш (#), это означает, что соответствующие модули дезактивированы. Удалите эти символы и сохраните файл.

Удалите точку с запятой в начале этих двух строк, если такие символы там имеются. Затем сохраните файл.

5 На последнем этапе перезапустите веб-сервер Apache, чтобы все внесенные вами в конфигурацию изменения возымели эффект. Чтобы сделать это, перейдите в панель управления операционной системы, затем в каталог «Администрирование» и запустите программу «Службы». Вы должны увидеть следующее:



Выберите Apache и щелкните кнопкой мыши на ссылке «Перезапустить».

Выберите Apache и щелкните кнопкой мыши на ссылке «Перезапустить» в меню слева. Теперь все функции `mysqli` и графической библиотеки GD должны работать корректно.

И на Mac...

К сожалению, это немного труднее. Добавление модулей на Mac требует перекомпиляции исходных файлов PHP с передачей информации о включаемых модулях в виде аргументов. Имеется слишком много различных комбинаций операционных систем Mac и версий PHP, чтобы включить их в короткое приложение. Предлагаем отличное руководство, которое поможет вам установить графическую библиотеку GD. Вы можете найти его по адресу:

```
http://macoshelp.blogspot.com/2008/02/adding-gd-library-for-mac-os-x-leopard.html
```

Это будет работать, только если у вас версия OS X (Leopard) и PHP версии 5. Если у вас другие версии этих программ или возникают какие-либо проблемы, то можете поискать более детальную информацию и комментарии относительно ваших условий на этом сайте или на сайте графической библиотеки GD по адресу:

```
http://www.libgd.org/
```

За помощью по перекомпиляции PHP при установке модуля `mysqli` на Mac OS мы рекомендуем обратиться по адресу:

```
http://dev.mysql.com/downloads/connector/php-mysqlnd/
```

Имейте в виду, что перекомпиляция PHP при установке модуля mysqli и графической библиотеки GD необходима только в том случае, если вы хотите запустить веб-сервер на вашем локальном компьютере, работающем под управлением операционной системы Mac OS. Но если вы используете свой Mac для написания PHP-кода, который затем загружаете на другой компьютер, тогда этих проблем у вас не будет.

Указатель

Знаки

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

W

X

Y

А

Б

В

Д

З

И

К

Т

У

Ф

Х

Ц

Ш

Э