

Оглавление

Введение	15
Что могут PHP и MySQL	15
Что такое PHP	16
На что похож PHP	16
PHP ориентирован на работу в Сети.	17
PHP не допускает столько вольностей, сколько JavaScript.	19
PHP — это интерпретатор	19
PHP выполняется не внутри браузера	20
Что такое MySQL	21
Об этой книге	21
Macintosh и Windows	22
FTP: важная деталь	22
Структура книги	23
Внешние ресурсы	24
Недостающий компакт-диск	24
Регистрация.	24
Отзывы	24
Исправления	25
Safari Book Online	25
Об авторе	26
О творческой группе	26
Благодарности	27
От издательства	28

Часть 1. Основы PHP и MySQL

Глава 1. PHP: что, где и зачем?	30
1.1. Две ипостаси PHP — локальная и удаленная	30
HTML и CSS выполняются внутри браузера	30
JavaScript — добавляем сложность, но не программы	32

PHP — не часть браузера	33
Писать где угодно, выполнять там, где есть PHP	35
1.2. PHP — работаем локально	36
PHP на компьютере PC под Windows (инсталлируем WampServer).	37
PHP на компьютерах Macintosh (стандартная инсталляция)	41
PHP на компьютерах Macintosh (инсталляция MAMP)	45
Выбор текстового редактора	48
1.3. Пишем нашу первую программу	53
1.4. Выполняем первую программу	55
1.5. Но где же этот веб-сервер?	56
Интерпретатор PHP — это программа, к которой можно обратиться	56
Однако HTML уже близко.	57
Глава 2. Объединяем PHP и HTML	58
2.1. Сценарий или HTML?	58
Выбор по расширению	59
Интерпретация HTML как HTML	60
PHP-файл — не HTML.	62
Но ответы PHP могут быть HTML	62
2.2. PHP отвечает	64
Пишем еще один PHP-сценарий	64
Переменные	65
Выполняем все это локально	66
2.3. Удаленное выполнение PHP-сценариев	67
2.4. Выкладывание кода HTML, CSS и PHP	68
Выполняем нашу вторую программу.	69
Добро пожаловать в программирование!	72
Глава 3. Синтаксис PHP: чудной и чудесный	73
3.1. Получение информации из веб-формы	73
3.2. Создание собственных переменных.	78
3.3. Работа с текстом в PHP	81
Объединение текста	81
Поиск в тексте.	82
Изменение текста	87
Обрезка и замена текста	92
3.4. Переменная \$_REQUEST — это массив	97
Массивы могут содержать несколько значений.	97
Работа с \$_REQUEST как с массивом	99
Что делать с пользовательской информацией?	103

Глава 4. MySQL и SQL: база данных и язык	105
4.1. Что такое база данных	105
Базы данных являются постоянным хранилищем	105
База данных — это в первую очередь структура.	107
Хорошие базы данных являются реляционными.	108
4.2. Установка MySQL	110
Консольная программа mysql: ваш новый лучший друг.	111
Запустите инструмент mysql на WampServer.	111
Запустите инструмент mysql в MAMP	112
Выполните свой первый SQL-запрос.	115
4.3. SQL — язык для разговора с базами данных	120
Вход в базу данных вашего веб-сервера.	120
Использование базы данных с помощью команды USE	123
Создание таблиц с помощью инструкции CREATE	124
Удаление таблиц с помощью команды DROP	128
Вставка нескольких строк с помощью команды INSERT.	128
И в завершение — команда SELECT	129

Часть 2. Динамические веб-страницы

Глава 5. Подключение PHP к MySQL.	134
5.1. Создание простого PHP-сценария, предназначенного для подключения.	134
5.2. Выбор используемой базы данных.	140
Демонстрация таблиц базы данных с помощью команды SHOW	141
Обработка ошибок путем наблюдения за отсутствием результата.	142
Вывод SQL-результатов	143
5.3. Приведение кода в порядок с помощью нескольких файлов	147
Замена набранных вручную значений переменными	148
Абстрагирование важных значений путем помещения их в отдельный файл	149
Переменные изменяются, а константы сохраняют постоянство.	152
Создание элементарного исполнителя SQL-запросов	154
Создание HTML-формы с большим пустым полем ввода	155
Подключение к базе данных (еще раз).	155
Запуск пользовательского SQL-запроса (еще раз).	158
Ввод вашего первого запроса, основанного на применении веб-технологий	160
Обработка запросов, не выбирающих информацию с помощью команды SELECT	163

Учет человеческого фактора	168
По возможности нужно избегать внесения изменений в то, что ввел пользователь.	169
Глава 6. Улучшение поиска с помощью регулярных выражений	173
6.1. Сопоставление строк, двойная скорость	174
Простая программа поиска в строке.	174
Поиск одной строки... или другой.	177
Учет позиции.	180
Избавление от trim и strtoupper	181
6.2. Поиск набора символов.	185
Регулярные выражения: к бесконечности и еще дальше.	188
Небольшая уборка: удалим команды echo	189
Глава 7. Создание динамических веб-страниц	191
7.1. Повторное обращение к пользовательской информации	191
7.2. Проектирование таблиц базы данных	193
В правильных таблицах баз данных всегда есть столбец id.	195
Ваш друг автоприращение.	195
ID и первичные ключи — хорошие компаньоны	196
Добавление ограничений к базе данных	198
7.3. Сохранение информации о пользователе	200
Создание SQL-запроса	202
Вставка данных о пользователе	204
Первое действие по подтверждению	206
Не нужно путать пользователей с программистами	208
7.4. Покажите мне пользователя	210
Макетирование страницы профиля пользователя.	210
Изменение структуры таблицы с помощью ALTER.	213
Создание сценария: первый проход	215
Выбор пользователя из базы данных с помощью инструкции SELECT	221
Извлечение значений из результата SQL-запроса	224
Получение ID пользователя сценарием show_user.php	226
7.5. Перенаправление и повторное обращение к сценарию, создающему новых пользователей	229
Обновление формы регистрации	229
Обновление сценария создания пользователя	232
Усовершенствование кода с помощью регулярных выражений (в очередной раз)	237

Часть 3. Переход от веб-страниц к веб-приложениям

Глава 8. Когда что-то не получается (но должно получаться)	242
8.1. Проектирование страниц ошибок	244
Что должны видеть пользователи	245
Понятие о том, когда и сколько нужно говорить	250
8.2. Поиск компромисса для страниц ошибок с помощью PHP	250
Создание страницы ошибки с кодом PHP	251
Проверка принятого решения	254
Ожидайте неожиданного	255
А теперь вас ждут проблемы безопасности и фишинга	257
8.3. Добавление отладки к приложению	260
Кто в конечном итоге использует это приложение	260
Сейчас вы меня видите, а сейчас — нет	261
Переход от <code>require</code> к <code>require_once</code>	263
8.4. Переадресация на ошибку	265
Обновление вашего сценария для использования	
<code>show_error.php</code>	265
Простота и абстракция	267
Переадресация не видит пути к файлу	270
Глава 9. Обработка изображений и решение более сложных задач	275
9.1. Изображения — это просто файлы	276
Формы HTML могут готовить почву	278
Отправка изображения пользователя на ваш сервер	281
Были ли ошибки при отправке файла?	287
Сохранение местоположения изображения в базе данных	296
9.2. Изображения, предназначенные для просмотра	301
Выбор изображения с помощью инструкции <code>SELECT</code>	
и вывод его на экран	302
Преобразование путей файловой системы в URL-адреса	304
Отображение картинки вашего пользователя: дубль два	308
9.3. А теперь совсем о другом	310
Глава 10. Двоичные объекты и загрузка изображений	312
10.1. Хранение разных объектов в различных таблицах	312
10.2. Вставка в таблицу необработанного изображения	315
Функция <code>getimagesize</code> не возвращает размер файла	318

Функция <code>file_get_contents</code> оправдывает свое название	318
Вставка изображения с помощью инструкции <code>INSERT</code>	318
10.3. Пока ваши двоичные данные вставлять небезопасно.	319
Вставка строки в переменную	320
Получение правильного ID перед перенаправлением	324
10.4. Связывание пользователей и изображений	327
Вставка изображения при вставке пользователя	328
Связывание таблиц с помощью условия <code>WHERE</code>	334
10.5. Покажите мне изображение	337
Вывод изображения.	337
Перехват и обработка ошибок	343
Тест, тест и еще раз тест	346
10.6. Встроить изображение ничуть не сложнее, чем его просмотреть	347
Вам нужен лишь идентификатор изображения	348
Сценарий может быть в виде указания в теге источника изображения (<code>src</code>)	349
10.7. Какой же подход лучше?	353
Глава 11. Вывод списков, итерация и администрирование.	355
11.1. Что вам нужно как администратору (вещи, которые никогда не меняются)	356
Пользовательский интерфейс, или Краткость по-прежнему сестра таланта.	356
Нужен также список пожеланий.	358
11.2. Вывод списка всех пользователей	359
Выбор с помощью <code>SELECT</code> нужной (на данный момент) информации	360
Создание простой страницы администрирования	362
Перебор элементов массива.	364
11.3. Удаление пользователя.	367
Разбор отдельных компонентов	368
Объединение всех составляющих.	369
Удаление пользователей не должно быть тайной операцией	371
11.4. Возражения, высказываемые вашим пользователям	375
У перенаправления есть некоторые ограничения	376
Возвращение окна предупреждения, создаваемого с помощью <code>JavaScript</code>	379
Функция <code>alert</code> прерывает действия	385
11.5. Приведение сообщений к единому стандарту	387
Создание новой сервисной функции для отображения	389

Появление дубликатов — вполне ожидаемая проблема	392
Коды сценариев View и Display имеют общий характер.	394
11.6. Интеграция утилит, представлений и сообщений.	394
Вызов повторяющегося кода из сценария View.	395
Лучше использовать гибкие функции.	396
Стандартизация и объединение вывода сообщений в сценарии View.	401
Создание функции для вызова двух функций.	404
Теперь нужно просто распространить эту информацию на весь код	405

Часть 4. Безопасность и реальное окружение

Глава 12. Аутентификация и авторизация	410
12.1. Начнем со стандартной аутентификации	411
Стандартная аутентификация с использованием HTTP-заголовков	412
Стандартная аутентификация проводится... стандартно.	413
Самая худшая из всех аутентификаций	414
Получение данных о полномочиях вашего пользователя	415
Отмена не подходит для аутентификации	416
Получение пользовательских полномочий (на этот раз всерьез!)	418
12.2. Извлечение всего одинакового	422
12.3. Пароли не должны находиться в сценариях PHP	426
Обновление таблицы users	426
Работа с вновь созданными недопустимыми данными.	427
Вам нужно получить исходные имя пользователя и пароль	429
Вставка имени пользователя и пароля	432
Подключение сценария authorize.php к таблице users.	437
12.4. Пароли обеспечивают безопасность, но и сами они должны быть защищены	441
Шифрование текста с помощью функции crypt.	441
Однонаправленное шифрование с помощью функции crypt	443
При шифровании используется соль	444
Глава 13. Cookie-файлы, вопросы регистрации и избавление от примитивных окон	446

13.1. Выход за рамки стандартной аутентификации	447
Начало создания стартовой страницы	448
Управление регистрацией пользователя при входе в приложение	449
От HTTP-аутентификации к использованию cookie-файлов	452
13.2. Регистрация при входе в приложение с использованием cookie-файлов	454
Зарегистрировался пользователь или нет?	455
Пытается ли пользователь зарегистрироваться?	456
Отображение страницы	458
Перенаправление по мере необходимости	459
Регистрация пользователя при входе в приложение.	461
Пустые страницы и истечение срока действия cookie-файлов	464
Ошибки не всегда должны прерывать работу приложения	467
Настройка на повторные попытки	470
13.3. Добавление контекстно зависимых меню.	472
Установка меню	473
От HTML к сценариям	476
Отмена регистрации пользователей.	479
Требование создания cookie-файла	481
Глава 14. Авторизация и сессии	485
14.1. Моделирование групп в базе данных.	485
Добавление таблицы groups	486
Отношение «многие ко многим»	487
Проведение теста на принадлежность к группе	491
14.2. Проверка на принадлежность к группе	492
Сценарий authorize.php нуждается в функции	493
Получение списка групп	496
Последовательный перебор групп	497
Разрешить, отказать, перенаправить	499
14.3. Меню, ориентированное на принадлежность к той или иной группе	503
14.4. Введение в практику использования сессий браузера	507
Сессии находятся на серверной стороне	509
Сессии должны быть запущены	509
От \$_COOKIE к \$_SESSION	510
Сессии должны быть еще и перезапущены.	511
Переменная \$_REQUEST не включает в себя данные переменной \$_SESSION	516

Меню для любого пользователя?	517
А теперь отмените регистрацию.	518
14.5. А вы не забыли о проблеме фишинга?	519
14.6. А зачем вообще использовать cookie-файлы?	522
Приложение А. Установка PHP на Windows без WAMP.	523
Приложение Б. Установка MySQL без MAMP или WAMP.	530
Установка MySQL.	530
MySQL для Windows.	530
MySQL для Mac OS X	537