Дмитрий Котеров, Игорь Симдянов

bhv®

PHP 7

- Нововведения PHP 7
- Объектно-ориентированное программирование
- Компоненты PHP и Composer
- Стандарты PSR
- PHP-FPM и nginx
- Исходные коды на GitHub

+ Материалы на www.bhv.ru

Наиболее полное руководство

В ПОДЛИННИКЕ®

Дмитрий Котеров
Игорь Симдянов

PHP 7

Санкт-Петербург
«БХВ-Петербург»
2016

УДК 004.438 PHP
ББК 32.973.26-018.1
К73

Котеров, Д. В.

К73 PHP 7 / Д. В. Котеров, И. В. Симдянов. — СПб.: БХВ-Петербург, 2016. — 1088 с.: ил. — (В подлиннике)

ISBN 978-5-9775-3725-4

Рассмотрены основы языка PHP и его рабочего окружения в Windows, Mac OS X и Linux.

Отражены радикальные изменения в языке PHP, произошедшие с момента выхода предыдущего издания: трейты, пространство имен, анонимные функции, замыкания, элементы строгой типизации, генераторы, встроенный Web-сервер и многие другие возможности. Приведено описание синтаксиса PHP 7, а также функций для работы с массивами, файлами, СУБД MySQL, memcached, регулярными выражениями, графическими примитивами, почтой, сессиями и т. д. Особое внимание уделено рабочему окружению: сборке PHP-FPM и Web-сервера nginx, СУБД MySQL, протоколу SSH, виртуальным машинам VirtualBox и менеджеру виртуальных машин Vagrant. Рассмотрены современные подходы к Web-разработке, система контроля версий Git, GitHub и другие бесплатные Git-хостинги, новая система распространения программных библиотек и их разработки, сборка Web-приложений менеджером Composer, стандарты PSR и другие инструменты и приемы работы современного PHP-сообщества.

В третьем издании добавлены 24 новые главы, остальные главы обновлены или переработаны.

На сайте издательства находятся исходные коды всех листингов.

Для Web-программистов

УДК 004.438 PHP
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Екатерина Капалыгина*
Редактор	*Анна Кузьмина*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Марины Дамбиевой*

Подписано в печать 29.04.16.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 87,72.
Тираж 1500 экз. Заказ №
"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

Первая Академическая типография "Наука"
199034, Санкт-Петербург, 9 линия, 12/28

ISBN 978-5-9775-3725-4

Оглавление

Предисловие **29**
Для кого написана эта книга 29
Исходные коды 30
Третье издание 31
Общая структура книги 31
Часть I 31
Часть II 32
Часть III 32
Часть IV 32
Часть V 33
Часть VI 33
Часть VII 33
Часть VIII 33
Часть IX 33
Часть X 33
Листинги 34
Предметный указатель 34
Благодарности от Дмитрия Котерова 34
Благодарности от Игоря Симдянова 35

Предисловие к первому изданию **39**

ЧАСТЬ I. ОСНОВЫ WEB-ПРОГРАММИРОВАНИЯ **41**

Глава 1. Принципы работы Интернета **43**
Протоколы передачи данных 43
Семейство TCP/IP 45
Адресация в Сети 46
IP-адрес 46
Версии протокола IP 47
Доменное имя 48
Порт 50
Установка соединения 51
Обмен данными 52

Терминология 52
Сервер 52
Узел 53
Порт 53
Сетевой демон, сервис, служба 53
Хост 54
Виртуальный хост 54
Провайдер 55
Хостинг-провайдер (хостер) 55
Хостинг 55
Виртуальный сервер 55
Сайт 56
HTML-документ 56
Страница (или HTML-страница) 56
Скрипт, сценарий 56
Web-программирование 56
Взаимосвязь терминов 57
World Wide Web и URL 57
Протокол 58
Имя хоста 58
Порт 58
Путь к странице 59
Резюме 59

Глава 2. Интерфейс CGI и протокол HTTP 60
Что такое CGI? 60
Секреты URL 61
Заголовки запроса и метод *GET* 62
GET 63
POST 64
Content-Type 64
Host 64
User-Agent 65
Referer 65
Content-length 65
Cookie 66
Accept 66
Эмуляция браузера через telnet 66
Метод *POST* 67
URL-кодирование 68
Что такое формы и для чего они нужны? 68
Передача параметров "вручную" 69
Использование формы 69
Абсолютный и относительный пути к сценарию 70
Метод *POST* и формы 71
Резюме 71

Глава 3. CGI изнутри ... 72
Язык C ... 72
Работа с исходными текстами на C ... 73
Компиляция программ ... 73
Передача документа пользователю ... 74
Заголовки ответа ... 74
Заголовок кода ответа ... 74
"Подделывание" заголовка ответа ... 75
Content-type ... 75
Pragma ... 75
Location ... 75
Set-cookie ... 76
Date ... 76
Server ... 76
Примеры CGI-сценариев на C ... 76
Вывод бинарного файла ... 77
Передача информации CGI-сценарию ... 78
Переменные окружения ... 78
Передача параметров методом *GET* ... 80
Передача параметров методом *POST* ... 81
Расшифровка URL-кодированных данных ... 83
Формы ... 86
Тег <*input*> — различные поля ввода ... 87
Текстовое поле *(text)* ... 87
Поле ввода пароля *(password)* ... 87
Скрытое текстовое поле *(hidden)* ... 88
Независимый переключатель *(checkbox)* ... 89
Зависимый переключатель *(radio)* ... 89
Кнопка отправки формы *(submit)* ... 90
Кнопка сброса формы *(reset)* ... 90
Рисунок для отправки формы *(image)* ... 90
Тег <*textarea*> — многострочное поле ввода текста ... 90
Тег <*select*> — список ... 91
Списки множественного выбора *(multiple)* ... 92
HTML-сущности ... 92
Загрузка файлов ... 93
Формат данных ... 94
Тег загрузки файла *(file)* ... 95
Что такое cookies и "с чем их едят"? ... 96
Установка cookie ... 98
Получение cookies из браузера ... 100
Пример программы для работы с cookies ... 100
Аутентификация ... 101
Резюме ... 103

Глава 4. Встроенный сервер PHP ... 104
Установка PHP в Windows ... 104
Переменная окружения *PATH* ... 105

Установка PHP в Mac OS X ... 107
Установка PHP в Linux (Ubuntu) ... 109
Запуск встроенного сервера ... 109
Файл hosts ... 110
Вещание вовне ... 110
Конфигурирование PHP ... 111
Резюме ... 111

ЧАСТЬ II. ОСНОВЫ ЯЗЫКА PHP ... 113

Глава 5. Характеристика языка PHP ... 115
История PHP ... 115
Что нового в PHP 7? ... 119
Пример PHP-программы ... 120
Использование PHP в Web ... 124
Резюме ... 126

Глава 6. Переменные, константы, типы данных ... 127
Переменные ... 127
Копирование переменных ... 128
Типы переменных ... 128
integer (целое число) ... 128
double (вещественное число) ... 129
string (строка текста) ... 130
array (ассоциативный массив) ... 130
object (ссылка на объект) ... 131
resource (ресурс) ... 131
boolean (логический тип) ... 131
null (специальное значение) ... 132
callable (функция обратного вызова) ... 132
Действия с переменными ... 132
Присвоение значения ... 132
Проверка существования ... 132
Уничтожение ... 133
Определение типа переменной ... 133
Установка типа переменной ... 134
Оператор присваивания ... 136
Ссылочные переменные ... 136
Жесткие ссылки ... 136
"Сбор мусора" ... 137
Символические ссылки ... 138
Ссылки на объекты ... 138
Некоторые условные обозначения ... 139
Константы ... 141
Предопределенные константы ... 142
Определение констант ... 143
Проверка существования константы ... 143
Константы с динамическими именами ... 143
Отладочные функции ... 144
Резюме ... 146

Глава 7. Выражения и операции PHP 147
Выражения 147
Логические выражения 148
Строковые выражения 148
Строка в апострофах 149
Строка в кавычках 149
Here-документ 150
Now-документ 151
Вызов внешней программы 151
Операции 151
Арифметические операции 151
Строковые операции 152
Операции присваивания 152
Операции инкремента и декремента 153
Битовые операции 153
Операции сравнения 158
Особенности операторов == и != 159
Сравнение сложных переменных 160
Операция эквивалентности 160
Оператор <=> 162
Логические операции 162
Операция отключения предупреждений 163
Особенности оператора @ 164
Противопоказания к использованию 165
Условные операции 165
Резюме 167

Глава 8. Работа с данными формы 168
Передача данных командной строки 168
Формы 170
Трансляция полей формы 171
Трансляция переменных окружения 173
Трансляция cookies 173
Обработка списков 174
Обработка массивов 175
Диагностика 176
Порядок трансляции переменных 177
Особенности флажков *checkbox* 177
Резюме 179

Глава 9. Конструкции языка 180
Инструкция *if-else* 180
Использование альтернативного синтаксиса 181
Цикл с предусловием *while* 182
Цикл с постусловием *do-while* 183
Универсальный цикл *for* 183
Инструкции *break* и *continue* 184
Нетрадиционное использование *do-while* и *break* 185

Цикл *foreach* 187
Конструкция *switch-case* 188
Инструкции *goto* 188
Инструкции *require* и *include* 189
Инструкции однократного включения 190
Суть проблемы 191
Решение: *require_once* 192
Другие инструкции 193
Резюме 193

Глава 10. Ассоциативные массивы 194
Создание массива "на лету". Автомассивы 195
Конструкция *list()* 196
Списки и ассоциативные массивы: путаница?.. 197
Конструкция *array()* и многомерные массивы 197
Массивы-константы 199
Операции над массивами 199
Доступ по ключу 199
Функция *count()* 199
Слияние массивов 200
Слияние списков 200
Обновление элементов 200
Косвенный перебор элементов массива 201
Перебор списка 201
Перебор ассоциативного массива 202
Недостатки косвенного перебора 203
Вложенные циклы 203
Нулевой ключ 203
Прямой перебор массива 204
Старый способ перебора 204
Перебор циклом *foreach* 204
Ссылочный синтаксис *foreach* 204
Списки и строки 205
Сериализация 206
Упаковка 207
Распаковка 207
Резюме 208

Глава 11. Функции и области видимости 209
Пример функции 209
Общий синтаксис определения функции 211
Инструкция *return* 211
Объявление и вызов функции 213
Параметры по умолчанию 213
Передача параметров по ссылке 214
Переменное число параметров 215
Типы аргументов и возвращаемого значения 217
Локальные переменные 218

Глобальные переменные 219
Массив *$GLOBALS* 220
Самовложенность 221
Как работает инструкция *global* 221
Статические переменные 222
Рекурсия 223
Факториал 223
Пример функции: *dumper()* 223
Вложенные функции 225
Условно определяемые функции 226
Эмуляция функции *virtual()* 227
Передача функций по ссылке 228
Использование *call_user_func()* 228
Использование *call_user_func_array()* 229
Анонимные функции 229
Замыкания 230
Возврат функцией ссылки 232
Технология отложенного копирования 233
Несколько советов по использованию функций 235
Резюме 236

Глава 12. Генераторы 237
Отложенные вычисления 237
Манипуляция массивами 240
Делегирование генераторов 242
Экономия ресурсов 243
Использование ключей 244
Использование ссылки 244
Связь генераторов с объектами 245
Резюме 247

ЧАСТЬ III. СТАНДАРТНЫЕ ФУНКЦИИ PHP 249

Глава 13. Строковые функции 251
Кодировки 251
UTF-8 и PHP 255
Конкатенация строк 258
О сравнении строк 258
Особенности *strpos()* 259
Отрезание пробелов 260
Базовые функции 261
Работа с подстроками 262
Замена 262
Подстановка 263
Преобразования символов 267
Изменение регистра 269
Установка локали (локальных настроек) 270
Функции форматных преобразований 271
Форматирование текста 273

Работа с бинарными данными 274
Хэш-функции 276
Сброс буфера вывода 278
Резюме 278

Глава 14. Работа с массивами 279
Лексикографическая и числовая сортировки 279
Сортировка произвольных массивов 280
Сортировка по значениям 280
Сортировка по ключам 280
Пользовательская сортировка по ключам 281
Пользовательская сортировка по значениям 282
Переворачивание массива 282
"Естественная" сортировка 283
Сортировка списков 284
Сортировка списка 284
Пользовательская сортировка списка 285
Сортировка многомерных массивов 285
Перемешивание списка 288
Ключи и значения 288
Слияние массивов 289
Работа с подмассивами 290
Работа со стеком и очередью 291
Переменные и массивы 292
Применение в шаблонах 293
Создание диапазона чисел 294
Работа с множествами 295
Пересечение 295
Разность 295
Объединение 295
JSON-формат 296
Резюме 301

Глава 15. Математические функции 302
Встроенные константы 302
Функции округления 303
Случайные числа 305
Перевод в различные системы счисления 308
Минимум и максимум 308
Не-числа 309
Степенные функции 310
Тригонометрия 310
Резюме 312

Глава 16. Работа с файлами и каталогами 313
О текстовых и бинарных файлах 313
Открытие файла 314
Конструкция *or die()* 316
Различия текстового и бинарного режимов 316

Сетевые соединения 317
Прямые и обратные слеши 317
Безымянные временные файлы 318
Закрытие файла 319
Чтение и запись 319
Блочные чтение/запись 320
Построчные чтение/запись 320
Чтение CSV-файла 321
Положение указателя текущей позиции 322
Работа с путями 323
Манипулирование целыми файлами 325
Чтение и запись целого файла 325
Чтение INI-файла 326
Другие функции 328
Блокирование файла 329
Рекомендательная и жесткая блокировки 329
Функция *flock()* 329
Типы блокировок 330
Исключительная блокировка 330
"Не убий!" 331
"Посади дерево" 331
"Следи за собой, будь осторожен" 332
Выводы 333
Разделяемая блокировка 333
Выводы 335
Блокировки с запретом "подвисания" 335
Пример счетчика 335
Работа с каталогами 336
Манипулирование каталогами 336
Работа с записями 337
Пример: печать дерева каталогов 338
Получение содержимого каталога 339
Резюме 341

Глава 17. Права доступа и атрибуты файлов 342
Идентификатор пользователя 342
Идентификатор группы 343
Владелец файла 344
Права доступа 344
Числовое представление прав доступа 345
Особенности каталогов 345
Примеры 347
Домашний каталог пользователя 347
Защищенный от записи файл 347
CGI-скрипт 347
Системные утилиты 348
Закрытые системные файлы 348
Функции PHP 348
Права доступа 348

Определение атрибутов файла....350
Специальные функции....351
Определение типа файла....352
Определение возможности доступа....353
Ссылки....353
Символические ссылки....353
Жесткие ссылки....354
Резюме....355

Глава 18. Запуск внешних программ....356
Запуск утилит....356
Оператор "обратные апострофы"....358
Экранирование командной строки....358
Каналы....359
Временные файлы....359
Открытие канала....360
Взаимная блокировка (deadlock)....360
Резюме....362

Глава 19. Работа с датой и временем....363
Установка часового пояса....363
Представление времени в формате timestamp....363
Вычисление времени работы скрипта....364
Большие вещественные числа....364
Построение строкового представления даты....365
Построение timestamp....367
Разбор timestamp....369
Григорианский календарь....370
Проверка даты....371
Календарик....371
Дата и время по Гринвичу....373
Время по GMT....373
Хранение абсолютного времени....374
Перевод времени....375
Окончательное решение задачи....376
Резюме....376

Глава 20. Основы регулярных выражений....377
Начнем с примеров....377
Пример первый....377
Пример второй....378
Пример третий....378
Пример четвертый....379
What is the PCRE?....380
Терминология....380
Использование регулярных выражений в PHP....381
Сопоставление....381
Сопоставление с заменой....382

Язык PCRE 383
Ограничители 383
Альтернативные ограничители 384
Отмена действия спецсимволов 384
Простые символы (литералы) 385
Классы символов 386
Альтернативы 386
Отрицательные классы 387
Квантификаторы повторений 388
Ноль или более совпадений 388
Одно или более совпадений 388
Ноль или одно совпадение 389
Заданное число совпадений 389
Мнимые символы 389
Оператор альтернативы 390
Группирующие скобки 390
"Карманы" 390
Использование карманов в функции замены 392
Использование карманов в функции сопоставления 393
Игнорирование карманов 394
Именованные карманы 394
"Жадность" квантификаторов 394
Рекуррентные структуры 396
Модификаторы 396
Модификатор */i*: игнорирование регистра 396
Модификатор */x*: пропуск пробелов и комментариев 396
Модификатор */m*: многострочность 397
Модификатор */s*: (однострочный поиск) 398
Модификатор */e*: выполнение PHP-программы при замене 398
Модификатор */u*: UTF-8 399
Незахватывающий поиск 399
Позитивный просмотр вперед 399
Негативный просмотр вперед 400
Позитивный просмотр назад 400
Негативный просмотр назад 401
Другие возможности PCRE 401
Функции PHP 401
Поиск совпадений 401
Замена совпадений 404
Разбиение по регулярному выражению 407
Выделение всех уникальных слов из текста 407
Экранирование символов 409
Фильтрация массива 409
Примеры использования регулярных выражений 410
Преобразование адресов e-mail 410
Преобразование гиперссылок 411
Быть или не быть? 412
Ссылки 412
Резюме 412

Глава 21. Разные функции 413
Информационные функции 413
Принудительное завершение программы 414
Финализаторы 415
Генерация кода во время выполнения 416
Выполнение кода 416
Генерация функций 418
Другие функции 420
Резюме 420

ЧАСТЬ IV. ОСНОВЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ 421

Глава 22. Объекты и классы 423
Класс как тип данных 423
Создание нового класса 425
Работа с классами 426
Создание объекта некоторого класса 426
Доступ к свойствам объекта 426
Доступ к методам 427
Создание нескольких объектов 428
Перегрузка преобразования в строку 429
Инициализация и разрушение 430
Конструктор 430
Параметры по умолчанию 431
Старый способ создания конструктора 432
Деструктор 432
Вопрос освобождения ресурсов 433
Описание деструктора 434
Алгоритм сбора мусора 436
Циклические ссылки 437
Проблема циклических ссылок 439
Права доступа к членам класса 440
Модификаторы доступа 440
Public: открытый доступ 441
Private: доступ только из методов класса 441
Protected: доступ из методов производного класса 442
Неявное объявление свойств 442
Общие рекомендации 443
Класс — *self*, объект — *$this* 443
Пример: счетчик объектов 444
Пример: кэш ресурсов 445
Константы класса 446
Перехват обращений к членам класса 447
Клонирование объектов 449
Переопределение операции клонирования 450
Запрет клонирования 450
Перехват сериализации 451

Сериализация объектов 451
Упаковка и распаковка объектов 452
Методы *__sleep()* и *__wakeup()* 453
Резюме 458

Глава 23. Наследование 459
Расширение класса 460
Метод включения 461
Недостатки метода 462
Несовместимость типов 463
Наследование 463
Переопределение методов 464
Модификаторы доступа при переопределении 465
Доступ к методам базового класса 465
Финальные методы 465
Запрет наследования 466
Константы *__CLASS__* и *__METHOD__* 466
Позднее статическое связывание 466
Анонимные классы 468
Полиморфизм 469
Абстрагирование 470
Виртуальные методы 475
Расширение иерархии 478
Абстрактные классы и методы 479
Совместимость родственных типов 481
Уточнение типа в функциях 481
Оператор *instanceof* 482
Обратное преобразование типа 482
Резюме 483

Глава 24. Интерфейсы и трейты 484
Интерфейсы 484
Наследование интерфейсов 486
Интерфейсы и абстрактные классы 488
Трейты 488
Трейты и наследование 490
Резюме 491

Глава 25. Пространство имен 492
Проблема именования 492
Объявление пространства имен 493
Иерархия пространства имен 496
Импортирование 498
Автозагрузка классов 498
Функция *__autoload()* 498
Функция *spl_autoload_register()* 501
Резюме 502

Глава 26. Обработка ошибок и исключения....503
Что такое ошибка?....503
Роли ошибок....504
Виды ошибок....504
Контроль ошибок....505
Директивы контроля ошибок....505
Установка режима вывода ошибок....507
Оператор отключения ошибок....508
Пример использования оператора @....509
Предостережения....509
Перехват ошибок....510
Проблемы с оператором @....512
Генерация ошибок....513
Стек вызовов функций....514
Исключения....515
Базовый синтаксис....515
Инструкция *throw*....517
Раскрутка стека....517
Исключения и деструкторы....518
Исключения и *set_error_handler()*....519
Классификация и наследование....521
Базовый класс *Exception*....522
Использование интерфейсов....523
Исключения в PHP 7....526
Блоки-финализаторы....527
Перехват всех исключений....528
Трансформация ошибок....529
Серьезность "несерьезных" ошибок....530
Преобразование ошибок в исключения....531
Пример....531
Код библиотеки *PHP_Exceptionizer*....532
Иерархия исключений....535
Фильтрация по типам ошибок....536
Резюме....537

ЧАСТЬ V. ПРЕДОПРЕДЕЛЕННЫЕ КЛАССЫ PHP....539

Глава 27. Предопределенные классы PHP....541
Класс *Directory*....541
Класс *Generator*....544
Класс *Closure*....545
Класс *IntlChar*....548
Резюме....549

Глава 28. Календарные классы PHP....550
Класс *DateTime*....550
Класс *DateTimeZone*....551
Класс *DateInterval*....552

Класс *DatePeriod* 554
Резюме 555

Глава 29. Итераторы 556
Стандартное поведение *foreach* 556
Определение собственного итератора 557
Как PHP обрабатывает итераторы 560
Множественные итераторы 561
Виртуальные массивы 561
Библиотека SPL 563
Класс *DirectoryIterator* 563
Класс *FilterIterator* 565
Класс *LimitIterator* 566
Рекурсивные итераторы 566
Резюме 567

Глава 30. Отражения 568
Неявный доступ к классам и методам 568
Неявный вызов метода 568
Неявный список аргументов 569
Инстанцирование классов 570
Использование неявных аргументов 570
Аппарат отражений 571
Функция: *ReflectionFunction* 572
Параметр функции: *ReflectionParameter* 574
Класс: *ReflectionClass* 574
Наследование и отражения 577
Свойство класса: *ReflectionProperty* 579
Метод класса: *ReflectionMethod* 580
Библиотека расширения: *ReflectionExtension* 580
Различные утилиты: *Reflection* 581
Исключение: *ReflectionException* 581
Иерархия 582
Резюме 582

ЧАСТЬ VI. РАБОТА С СЕТЬЮ В PHP 583

Глава 31. Работа с HTTP и WWW 585
Заголовки ответа 585
Вывод заголовка ответа 585
Проблемы с заголовками 585
Запрет кэширования 586
Получение выведенных заголовков 587
Получение заголовков запроса 588
Работа с cookies 588
Немного теории 588
Установка cookie 589
Массивы и cookie 590
Получение cookie 591

Разбор URL 591
Разбиение и "склеивание" *QUERY_STRING* 591
Разбиение и "склеивание" URL 593
Пример 594
Резюме 595

Глава 32. Сетевые функции 596
Файловые функции и потоки 596
Проблемы безопасности 597
Другие схемы 598
Контекст потока 598
Работа с сокетами 601
"Эмуляция" браузера 601
Неблокирующее чтение 602
Функции для работы с DNS 603
Преобразование IP-адреса в доменное имя и наоборот 603
Резюме 604

Глава 33. Посылка писем через PHP 605
Формат электронного письма 605
Отправка письма 606
Почтовые шаблоны 607
Расщепление заголовков 608
Анализ заголовков 609
Кодировка UTF-8 611
Заголовок *Content-type* и кодировка 611
Кодировка заголовков 611
Кодирование тела письма 613
Активные шаблоны 613
Отправка писем с вложением 616
Отправка писем со встроенными изображениями 619
Резюме 621

Глава 34. Управление сессиями 622
Что такое сессия? 623
Зачем нужны сессии? 623
Механизм работы сессий 624
Инициализация сессии 625
Пример использования сессии 625
Уничтожение сессии 626
Идентификатор сессии и имя группы 627
Имя группы сессий 627
Идентификатор сессии 628
Путь к временному каталогу 629
Стоит ли изменять группу сессий? 629
Установка обработчиков сессии 630
Обзор обработчиков 630
Регистрация обработчиков 631
Пример: переопределение обработчиков 632
Резюме 634

ЧАСТЬ VII. РАСШИРЕНИЯ PHP 635

Глава 35. Расширения PHP 637
Подключение расширений 637
Конфигурационный файл php.ini 640
Структура php.ini 640
Параметры языка PHP 641
Ограничение ресурсов 643
Загрузка файлов 644
Обзор расширений 644
Резюме 645

Глава 36. Фильтрация и проверка данных 646
Фильтрация или проверка? 646
Проверка данных 649
Фильтры проверки 651
Значения по умолчанию 656
Фильтры очистки 657
Пользовательская фильтрация данных 660
Фильтрация внешних данных 661
Конфигурационный файл php.ini 663
Резюме 665

Глава 37. Работа с СУБД MySQL 666
Что такое база данных? 666
Неудобство работы с файлами 667
Администрирование базы данных 668
Язык запросов СУБД MySQL 668
Первичные ключи 671
Создание и удаление базы данных 673
Выбор базы данных 675
Типы полей 675
Целые числа 675
Вещественные числа 676
Строки 676
Бинарные данные 677
Дата и время 677
Перечисления 678
Множества 678
Модификаторы и флаги типов 678
Создание и удаление таблиц 679
Вставка числовых значений в таблицу 684
Вставка строковых значений в таблицу 686
Вставка календарных значений 687
Вставка уникальных значений 689
Механизм *AUTO_INCREMENT* 690
Многострочный оператор *INSERT* 690
Удаление данных 691
Обновление записей 692

Выборка данных 694
Условная выборка 695
Псевдонимы столбцов 699
Сортировка записей 700
Вывод записей в случайном порядке 702
Ограничение выборки 702
Вывод уникальных значений 703
Расширение PDO 704
Установка соединения с базой данных 705
Выполнение SQL-запросов 706
Обработка ошибок 707
Извлечение данных 709
Параметризация SQL-запросов 711
Заполнение связанных таблиц 712
Резюме 715

Глава 38. Работа с изображениями 716
Универсальная функция *getimagesize()* 717
Работа с изображениями и библиотека GD 718
Пример создания изображения 719
Создание изображения 720
Загрузка изображения 720
Определение параметров изображения 721
Сохранение изображения 722
Преобразование изображения в палитровое 723
Работа с цветом в формате RGB 723
Создание нового цвета 723
Текстовое представление цвета 723
Получение ближайшего в палитре цвета 724
Эффект прозрачности 725
Получение RGB-составляющих 725
Использование полупрозрачных цветов 726
Графические примитивы 727
Копирование изображений 727
Прямоугольники 728
Выбор пера 729
Линии 730
Дуга сектора 730
Закраска произвольной области 730
Закраска текстурой 731
Многоугольники 731
Работа с пикселами 732
Работа с фиксированными шрифтами 732
Загрузка шрифта 733
Параметры шрифта 733
Вывод строки 734
Работа со шрифтами TrueType 734
Вывод строки 734

Проблемы с русскими буквами 735
Определение границ строки 735
Коррекция функции *imageTtfBBox()* 735
Пример 737
Резюме 739

Глава 39. Работа с сетью 740
Подключение расширений 740
Получение точного времени 745
Отправка данных методом *POST* 745
Передача пользовательского агента 747
Резюме 748

Глава 40. Сервер memcached 749
Настройка сервера memcached 749
Хранение сессий в memcached 750
Установка соединения с сервером 751
Помещение данных в memcached 752
Обработка ошибок 753
Замена данных в memcached 754
Извлечение данных из memcached 756
Удаление данных из memcached 758
Установка времени жизни 758
Работа с несколькими серверами 758
Резюме 763

ЧАСТЬ VIII. БИБЛИОТЕКИ 765

Глава 41. Компоненты 767
Composer: управление компонентами 767
Установка Composer 768
Установка в Windows 768
Установка в Mac OS X 770
Установка в Ubuntu 770
Где искать компоненты? 770
Установка компонента 770
Использование компонента 773
Полезные компоненты 773
Компонент psySH. Интерактивный отладчик 773
Компонент phinx. Миграции 775
Инициализация компонента 776
Подготовка миграций 776
Выполнение миграций 779
Откат миграций 780
Операции со столбцами 781
Подготовка тестовых данных 782
Резюме 784

Глава 42. Стандарты PSR 785
PSR-стандарты 785
PSR-1. Основной стандарт кодирования 786
PHP-теги 786
Кодировка UTF-8 786
Разделение объявлений и выполнения действий 787
Пространство имен 788
Именование классов, методов и констант классов 788
PSR-2. Руководство по стилю кода 789
Соблюдение PSR-1 789
Отступы 789
Файлы 790
Строки 790
Ключевые слова 790
Пространства имен 791
Классы 791
Методы 792
Управляющие структуры 793
Автоматическая проверка стиля 794
PSR-3. Протоколирование 795
PSR-4. Автозагрузка 797
PSR-6. Кэширование 797
PSR-7. HTTP-сообщения 799
Базовый интерфейс *MessageInterface* 800
Тело сообщения *StreamInterface* 802
Ответ сервера *ResponseInterface* 803
Запрос клиента *RequestInterface* 804
Запрос сервера *ServerRequestInterface* 806
Загрузка файлов *UploadedFileInterface* 807
Резюме 808

Глава 43. Документирование 809
Установка 809
Документирование PHP-элементов 810
Теги 811
Типы 815
Резюме 815

Глава 44. Разработка собственного компонента 816
Имя компонента и пространство имен 816
Организация компонента 817
Реализация компонента 820
Базовый класс навигации *Pager* 821
Постраничная навигация по содержимому папки 824
Базовый класс представления *View* 827
Представление: список страниц 828
Собираем все вместе 829
Постраничная навигация по содержимому файла 830

Постраничная навигация по содержимому базы данных ... 833
Представление: диапазон элементов ... 837
Публикация компонента ... 840
Зачем разрабатывать собственные компоненты? ... 842
Резюме ... 842

Глава 45. PHAR-архивы ... 843
Создание архива ... 843
Чтение архива ... 845
Распаковка архива ... 848
Упаковка произвольных файлов ... 848
Преобразование содержимого архива ... 851
Сжатие PHAR-архива ... 852
Утилита phar ... 854
Резюме ... 854

ЧАСТЬ IX. ПРИЕМЫ ПРОГРАММИРОВАНИЯ НА PHP ... 855

Глава 46. XML ... 857
Что такое XML? ... 858
Чтение XML-файла ... 859
XPath ... 862
Формирование XML-файла ... 863
Резюме ... 865

Глава 47. Загрузка файлов на сервер ... 866
Multipart-формы ... 867
Тег выбора файла ... 867
Закачка файлов и безопасность ... 867
Поддержка закачки в PHP ... 868
Простые имена полей закачки ... 868
Получение закачанного файла ... 870
Пример: фотоальбом ... 871
Сложные имена полей ... 872
Резюме ... 874

Глава 48. Использование перенаправлений ... 875
Внешний редирект ... 875
Внутренний редирект ... 876
Самопереадресация ... 878
Резюме ... 881

Глава 49. Перехват выходного потока ... 882
Функции перехвата ... 882
Стек буферов ... 883
Недостатки "ручного" перехвата ... 884
Использование объектов и деструкторов ... 885
Класс для перехвата выходного потока ... 886
Недостатки класса ... 889

Проблемы с отладкой ... 889
Обработчики буферов ... 890
GZip-сжатие ... 891
Печать эффективности сжатия ... 892
Резюме ... 894

Глава 50. Код и шаблон страницы ... 895
Первый способ: "вкрапление" HTML в код ... 895
Второй способ: вставка кода в шаблон ... 897
Третий способ: Model—View—Controller ... 898
Шаблон (View) ... 899
Контроллер (Controller) ... 899
Модель (Model) ... 901
Взаимодействие элементов ... 902
Активные и пассивные шаблоны ... 903
Активные шаблоны ... 903
Пассивные шаблоны ... 904
Недостатки MVC ... 906
Четвертый способ: компонентный подход ... 908
Блочная структура Web-страниц ... 908
Взаимодействие элементов ... 909
Шаблон (View) ... 911
Компоненты (Components) ... 913
Добавление записи ... 913
Показ записей ... 914
Показ новостей ... 914
Проверка корректности входных данных ... 915
Полномочия Компонентов ... 916
Достоинства подхода ... 917
Система Smarty ... 917
Трансляция в код на PHP ... 917
Использование Smarty в MVC-схеме ... 919
Инструкции Smarty ... 920
Одиночные и парные теги ... 920
Вставка значения переменной: *{$variable ...}* ... 921
Модификаторы ... 921
Перебор массива: *{foreach}...{/foreach}* ... 921
Ветвление: *{if}...{else}...{/if}* ... 922
Вставка содержимого внешнего файла: *{include}* ... 922
Вывод отладочной консоли: *{debug}* ... 923
Удаление пробелов: *{strip}...{/strip}* ... 923
Оператор присваивания: *{assign}* ... 924
Оператор перехвата блока: *{capture}* ... 924
Циклическая подстановка: *{cycle}* ... 924
Глоссарий ... 925
Резюме ... 926

Глава 51. AJAX ... 927
Что такое AJAX? ... 927
Что такое jQuery? ... 928

Обработка событий....930
Манипуляция содержимым страницы....932
Асинхронное обращение к серверу....936
AJAX-обращение к базе данных....937
Отправка данных методом *POST*....941
Двойной выпадающий список....946
Запоминание состояний флажков....949
Резюме....951

ЧАСТЬ X. РАЗВЕРТЫВАНИЕ....953

Глава 52. Протокол SSH....955
Ubuntu....956
Сервер OpenSSH....956
Установка SSH-сервера....956
Настройка SSH-сервера....956
Настройка доступа....956
Смена порта....958
Управление сервером....959
Клиент SSH....959
Обращение к удаленному серверу....959
Настройка клиента SSH....960
Псевдонимы....961
Доступ по ключу....961
Проброс ключа....963
SSH-агент ключа....964
Массовое выполнение команд....964
Загрузка и скачивание файлов по SSH-протоколу....965
Mac OS X....965
Windows....966
SSH-клиент PuTTY....966
Доступ по SSH-ключу....967
Копирование файлов по SSH-протоколу....970
Утилита pscp.exe....970
Клиент FileZilla....970
Cygwin....971
Установка....972
SSH-соединение....975
Резюме....975

Глава 53. Виртуальные машины....976
VirtualBox....977
Установка VirtualBox....977
Создание виртуальной машины....979
Установка операционной системы....981
Vagrant....985
Установка Vagrant....985
Создание виртуальной машины....986
Запуск виртуальной машины....986

Остановка виртуальной машины ... 989
Удаление виртуальной машины ... 989
Установка соединения с виртуальной машиной ... 989
Конфигурационный файл Vagrant ... 991
Управление оперативной памятью ... 991
Управление образами ... 992
Общие папки ... 992
Проброс порта ... 993
Установка программного обеспечения ... 994
Резюме ... 995

Глава 54. Система контроля версий Git ... 996
Основы Git ... 996
Установка Git ... 999
Установка в Ubuntu ... 999
Установка в Mac OS X ... 999
Установка в Windows ... 1000
Постустановочная настройка ... 1003
Локальная работа с Git-репозиторием ... 1003
Инициализация репозитория ... 1003
Клонирование репозитория ... 1004
Публикация изменений ... 1004
История изменений ... 1006
Игнорирование файлов с помощью .gitignore ... 1007
Откат по истории проекта ... 1007
Метки ... 1012
Ветки ... 1013
Разрешение конфликтов ... 1016
Удаленная работа с Git-репозиторием ... 1018
Удаленный репозиторий GitHub ... 1018
Получение изменений ... 1020
Развертывание сетевого Git-репозитория ... 1021
Резюме ... 1022

Глава 55. Web-сервер nginx ... 1023
Установка nginx ... 1024
Управление сервером ... 1024
Конфигурационные файлы ... 1025
Иерархия секций ... 1028
Виртуальные хосты ... 1029
Журнальные файлы ... 1031
Местоположения ... 1034
Резюме ... 1037

Глава 56. PHP-FPM ... 1038
Установка ... 1038
Управление сервером ... 1038
Конфигурационные файлы ... 1039
Подключение к Web-серверу nginx ... 1042
Резюме ... 1044

Глава 57. Администрирование MySQL.....1045
Установка.....1045
Управление сервером.....1046
Конфигурационный файл сервера.....1047
Выделение памяти MySQL.....1050
Пользовательский конфигурационный файл.....1054
Создание MySQL-пользователей.....1055
Удаленный доступ к MySQL.....1055
Привилегии.....1056
Восстановление утерянного пароля.....1059
Перенос баз данных с одного сервера на другой.....1060
Копирование бинарных файлов.....1060
Создание SQL-дампа.....1061
Резюме.....1062

Приложение. HTTP-коды.....1063

Предметный указатель.....1069

Предисловие

С момента первого издания книги прошло больше 10 лет. Интернет превратился из игрушки в рабочую среду, в часть повседневной инфраструктуры, наряду с дорогами, электросетями и водопроводом. Профессия Web-разработчика менялась вслед за Интернетом, появилась масса специализаций. Ушло множество мелких игроков — проекты укрупняются. Если раньше сотни сайтов ютились на нескольких серверах, современные проекты редко обходятся одним сервером. Средние проекты требуют десятков и сотен серверов, а крупные — тысячи. Интернет из клуба единомышленников превратился в бизнес.

Изменился и подход в разработке, широкие каналы связи, система контроля версий Git и сервисы бесплатного Git-хостинга вроде GitHub привели к совершенно новой среде разработки. Языки программирования окончательно оформляются в технологии, окруженные менеджерами пактов, инструментарием, сообществом, в своеобразную экосистему. Теперь не достаточно изучить лишь синтаксис языка, чтобы оставаться конкурентноспособным разработчиком, необходимо знать весь стек технологий.

Если раньше Web-разработка считалась недопрограммированием, сегодня это мейнстрим современного программирования и IT-сферы. Одно из наиболее перспективных направлений, которые может выбрать программист для приложения своих усилий. Революционные изменения в отрасли, деньги, внимание гарантированы. Интернет продолжает развиваться и перестраиваться. Государства, СМИ, общество в целом ищут способы существовать в новой среде и новой реальности. Миллионы людей осваивают новый мир. Пока этот процесс идет, а продлится он десятилетия, Web-разработчик всегда найдет точку приложения своему таланту.

Для кого написана эта книга

Если можешь не писать — не пиши.
А. П. Чехов

Книга, которую вы держите в руках, является в некотором роде *учебником* по Web-программированию на PHP. Мы сделали попытку написать ее так, чтобы даже плохо подготовленный читатель, никогда не работавший в Web и владеющий лишь основами программирования на одном из алгоритмических языков, смог получить большинство необходимых знаний и в минимальные сроки начать профессиональную работу в Web.

Заметьте еще раз: мы предполагаем, что вы *уже знакомы* с основными понятиями программирования и не будете, по крайней мере, путаться в циклах, условных операторах, подпрограммах и т. д. Программирование как таковое вообще слабо связано с конкретным языком; научившись писать на нескольких или даже на одном-единственном языке, вы в дальнейшем легко освоите все остальные.

> ***Замечание***
>
> Впрочем, даже если вы знаете только HTML и занимаетесь версткой, то можете попробовать освоить азы PHP по этой книге. Но тот факт, что HTML является *языком разметки*, а не языком программирования и не содержит многих программистских идей, сильно осложнит вам задачу.

Книга также будет полезна и уже успевшему поработать с PHP профессионалу, потому что она содержит массу подробностей по современному PHP. Мы охватываем все современные приемы разработки:

- объектно-ориентированное программирование;
- компоненты и менеджер пакетов Composer;
- исполняемые PHAR-архивы;
- сервер memcached и приемы работы с ним;
- стандарты PSR;
- протокол SSH;
- систему контроля версий Git;
- виртуальную машину VirtualBox и систему развертывания Vagrant;
- Web-сервер nginx в связке с PHP-FPM;
- нововведения PHP 7.

Мы не затрагиваем тестирование, не рассматриваем прожорливый и теряющий популярность Web-сервер Apache, не освещаем ни один из современных фреймворков или систем управления контентом (CMS). Впрочем, перечислять, что не вошло в книгу, можно бесконечно долго.

Исходные коды

Удивительное наблюдение: стоит только дать в книге адрес электронной почты авторов, как тут же на него начинают приходить самые разные письма от читателей. Предыдущая версия книги не была в этом отношении исключением. Основную массу писем составляли просьбы выслать исходные тексты листингов. По мере своих сил мы старались удовлетворять эти запросы, однако, конечно, сейчас так продолжаться уже не может.

Именно по этой причине *абсолютно все* исходные коды приведенных листингов теперь доступны для загрузки с GitHub.

https://github.com/igorsimdyanov/php7

Система контроля версий git и сервис GitHub подробно освещаются в *главе 54*. Если вы обнаружите неточность или опечатку, ждем ваших реквестов. Вопросы и предложения можно размещать в разделе **Issues**.

Третье издание

Вы держите в руках третье издание книги "PHP 7". Первое издание книги подготавливалось 12 лет назад для PHP 5. Релиз версии PHP 6 так никогда не увидел свет, о чем подробнее можно будет узнать в *главе 5*.

Революционные изменения в PHP и в Web-программировании потребовали кардинальной переработки книги. Мы написали 24 новые главы, дополнительно 9 глав подверглись существенной переработке. Оставшиеся главы были тщательно проработаны и обновлены. Перечислять по пунктам изменения не представляется возможным, из-за их объема.

Уважаемые читатели! Если вы хотите сделать следующее издание этой книги лучше, пожалуйста, публикуйте свои замечания по адресу:

https://github.com/igorsimdyanov/php7/issues

Практика показала эффективность такого подхода.

Общая структура книги

Книга состоит из 10 частей и 57 глав. Непосредственное описание языка PHP начинается с *части II*. Это объясняется необходимостью прежде узнать кое-что о CGI (Common Gateway Interface, общий шлюзовой интерфейс) — *часть I*. В *части III* разобраны наиболее полезные стандартные функции языка. *Часть IV* посвящена объектно-ориентированным возможностям PHP, а *часть V* — предопределенным классам. *Часть VI* освещает сетевые возможности PHP, а *часть VII* знакомит с модульной структурой интерпретатора и расширениями. *Часть VIII* посвящена управлению и созданию библиотек на PHP. *Часть IX* посвящена различным приемам программирования на PHP с множеством примеров. Наконец, *часть X* охватывает инструменты разработки и обслуживания Web-сайта.

Теперь чуть подробнее о каждой части книги.

Часть I

В ней рассматриваются теоретические аспекты программирования в Web, а также основы механизма, который позволяет программам работать в Сети. Если вы уже знакомы с этим материалом (например, занимались программированием на Perl или других языках), можете ее смело пропустить. Вкратце мы опишем, на чем базируется Web, что такое интерфейс CGI, как он работает на низком уровне, как используются возможности языка HTML при Web-программировании, как происходит взаимодействие CGI и HTML и многое другое.

В принципе, вся теория по Web-программированию коротко изложена именно в этой части книги (и, как показывают отзывы читателей книги, посвященной предыдущей версии PHP, многие почерпнули фундаментальные сведения по Web-программированию именно из этой части).

Так как CGI является независимым от платформы интерфейсом, материал не "привязан" к конкретному языку (хотя в примерах используется язык C как наиболее универ-

сальное средство программирования). Если вы не знаете C, не стоит отчаиваться: немногочисленные примеры на этом языке не настолько сложны, чтобы в них можно было запутаться. К тому же, каждое действие подробно комментируется. Большинство описанных идей будет повторно затронуто в последующих главах, посвященных уже PHP.

Последняя глава книги посвящена установке PHP на одной из основных операционных систем: Windows, Mac OS X и Linux. Встроенный Web-сервер позволяет сразу же приступить к Web-разработке. Именно он будет использоваться на протяжении всей книги вплоть до *части X*.

Часть II

Язык PHP — удобный и гибкий язык для программирования в Web. Его основам посвящена *часть II*. С помощью PHP можно написать 99% программ, которые обычно требуются в Интернете. Для оставшегося 1% придется использовать C или Java (или другой универсальный язык). Впрочем, даже это необязательно: вы сильно облегчите себе жизнь, если интерфейсную оболочку будете разрабатывать на PHP, а ядро — на C, особенно если ваша программа должна работать быстро (например, если вы пишете поисковую систему). Последняя тема в этой книге не рассматривается, поскольку требует довольно большого опыта низкоуровневого программирования на языке C, а потому не вписывается в концепцию данной книги.

Часть III

Часть III может быть использована не только как своеобразный учебник, но также и в справочных целях — ведь в ней рассказано о большинстве стандартных функций, встроенных в PHP. Мы группировали функции в соответствии с их назначением, а не в алфавитном порядке, как это иногда бывает принято в технической литературе. Что ж, думаем, книга от этого только выиграла. Содержание части во многих местах дублирует документацию, сопровождающую PHP, но это ни в коей мере не означает, что она является лишь ее грубым переводом (тем более, что такой перевод уже существует для многих глав официального руководства PHP). Наоборот, мы пытались взглянуть на "кухню" Web-программирования, так сказать, свежим взглядом, еще помня собственные ошибки и изыскания. Конечно, все функции PHP описать невозможно (потому что они добавляются и совершенствуются от версии к версии), да этого и не требуется, но львиная доля предоставляемых PHP возможностей все же будет нами рассмотрена.

Часть IV

Часть IV посвящена объектно-ориентированному программированию (ООП) на PHP. ООП в PHP постоянно развивается. Если еще несколько лет назад практически все программное обеспечение на PHP выполнялось в процедурном стиле, то в настоящий момент невозможно представить современную разработку без использования объектно-ориентированных возможностей языка.

Компоненты, из которых состоит современное приложение, завязаны на пространство имен, классы и механизм автозагрузки. В PHP 7 изменился способ обработки ошибок, который теперь интегрирован в механизм исключений.

Мы постарались полнее выделить все достоинства PHP: сокрытия данных, наследование, интерфейсы, трейты, пространство имен и автозагрузку классов.

Часть V

Часть V посвящена встроенным классам PHP. ООП в PHP появился не сразу, поэтому процесс интеграции ООП в язык занял определенное время. В главах этой части рассматриваются связь объектно-ориентированного подхода с замыканиями, генераторами, обсуждаются календарные классы, итераторы и механизм отражений.

Часть VI

Основная специализация языка PHP — разработка Web-приложений. Поэтому сетевые возможности языка выделены в отдельную часть. Главы части подробно освещают как низкоуровневую работу с сетью, так и обработку cookie, сессий, отправку почтовых сообщений.

Часть VII

PHP строится по модульному принципу. Модули-библиотеки, разработанные на языке C, называются *расширениями*. Разработано огромное количество самых разнообразных расширений, от обеспечения сетевых операций до работы с базами данных. *Часть VII* посвящена управлению расширениями PHP и освещению наиболее популярных из них.

Часть VIII

Если расширения — это часть языка PHP, то библиотеки, разработанные на PHP, называются *компонентами*. Менеджер пакетов Composer позволяет подключать, загружать, управлять версиями библиотек. Невозможно представить современную разработку на PHP без компонентов. *Часть VIII* служит путеводителем в мире компонентов, здесь же мы научимся разрабатывать и публиковать собственные компоненты, так же упаковывать компоненты в исполняемые PHAR-архивы.

Часть IX

Часть IX посвящена практическим приемам программирования на PHP. Она насыщена примерами программ и библиотек, которые облегчают работу программиста. В ней освещается работа с XML, загрузка файлов на сервер, использование перенаправления, техника отделения кода от шаблона страницы сценария, приемы AJAX-разработки.

Часть X

Заключительная *X часть* книги описывает установку и настройку средств разработки Web-программиста, в том числе утилиты для работы по SSH-протоколу, систему контроля версий Git, виртуальную машину VirtualBox и систему управления виртуальными машинами Vagrant, установку сервера базы данных MySQL, сервер PHP-FPM и Web-сервер nginx.

Листинги

Как уже говорилось ранее, тексты всех листингов книги доступны для загрузки через git-репозиторий **https://github.com/igorsimdyanov/php7/issues**. Их очень много — более 600! Чтобы вы не запутались, какой файл какому листингу соответствует, применен следующий подход.

- Определенной главе книги соответствует один и только один каталог в архиве с исходными кодами. Имя этого каталога (обычно это не очень длинный идентификатор, состоящий из английских букв) записано сразу же под названием главы.
- Одному листингу соответствует один и только один файл в архиве.
- Названия *всех* листингов в книге выглядят однотипно: "Листинг *M.N*. Необязательное название. Файл *X*". Здесь *M.N* — это номер главы и листинга, а *X* — имя файла относительно *текущего каталога главы*.
- Заглавие листинга приведено прямо в самом файле и оформлено в виде комментария в первой строке:
  - `<!-- ... --->` — для HTML-кода;
  - `##...` — для PHP-программ.

  Таким образом, мы убили сразу двух зайцев: во-первых, указали заголовок листинга в удобном месте, а во-вторых, позволили читателю сразу же узнать название листинга, открыв в текстовом редакторе соответствующий ему файл.

Теперь немного о том, как использовать файлы листингов. Большинство из них являются законченными (правда, простыми) сценариями на PHP, которые можно запустить на выполнение через тестовый Web-сервер. Напомним, что в *главе 4* освещается работа встроенного Web-сервера PHP, который может быть запущен в любой операционной системе. Таким образом, для проведения экспериментов с листингами вам достаточно просто развернуть архив в подходящий каталог.

Предметный указатель

Книга, которую вы держите в руках, содержит практически исчерпывающий указатель (индекс) по всем основным ключевым словам, встречающимся в тексте. В нем, помимо прочего, приводятся ссылки на все рассмотренные функции и константы, директивы PHP и MySQL, ключевые термины и понятия, встречающиеся в Web-программировании. Мы постарались сделать предметный указатель удобным для повседневного использования книги в качестве справочника.

Благодарности от Дмитрия Котерова

Редкая книга обходится без этого приятного раздела. Мы не станем, пожалуй, делать исключений и попробуем здесь перечислить всех, кто оказал то или иное влияние на ход написания книги.

Замечание

К сожалению, приятность данного момента омрачается тем фактом, что в силу линейности повествования приходится какие-то имена указывать выше, а какие-то ниже по тексту. Ведь порядок следования имен подчас не имеет ничего общего с числом "заслуг" того или иного человека. Вдобавок, всегда существует риск кого-то забыть — непреднамеренно, конечно. Но уж пусть лучше будет так, чем совсем без благодарностей.

Хочется прежде всего поблагодарить читателей первого издания книги "PHP 5", активно участвовавших в исправлении неточностей. Всего на форуме книги было опубликовано около 200 опечаток и исправлений! Мы надеемся, что благодаря этому книга стала значительно точнее, и ожидаем не меньшей активности от читателей для данного издания.

Отдельных слов благодарности заслуживают разработчики языка PHP, в сотрудничестве с которыми была написана данная книга. (Возможно, вы улыбнулись при прочтении этого предложения, однако под "сотрудничеством" мы здесь понимаем вовсе не само создание интерпретатора PHP! Речь идет о консультациях по электронной почте и *двусторонней* связи авторов книги с программистами.) Особенно хотелось бы выделить разработчика модуля DOM Роба Ричардса (Rob Richards). Кроме того, многие другие разработчики PHP (например, Маркус Боергер (Marcus Boerger), Илья Альшанецкий (Ilia Alshanetsky), Дерик Ретанс (Derick Rethans) и др.) оперативно исправляли ошибки в интерпретаторе PHP, найденные авторами книги в процессе ее написания.

Хочется также поблагодарить коллектив форума **http://forum.dklab.ru**, отдельные участники которого напрямую влияли на ход "шлифовки" книги. Например, Юрий Насретдинов прочитал и прокомментировал начальные версии глав про регулярные выражения и MySQL, а также высказал множество ценных замечаний по улучшению последний главы книги, посвященной AJAX. Антон Сущев и Ильдар Шайморданов помогали авторам в доработке *предисловия*, которое вы сейчас читаете. Наконец, многие участники форума в той или иной степени участвовали в обсуждениях насущных вопросов, ответы на которые вошли в книгу, а также занимались поддержкой проекта "Джентльменский набор Web-разработчика", позволяющего быстро установить Apache, PHP, MySQL и т. д. для отладки сразу нескольких сайтов в Windows.

Наконец, нам хотелось бы поблагодарить сотрудника издательства "БХВ-Петербург" Евгения Рыбакова, который стойко выносил все мыслимые и немыслимые превышения сроков сдачи материала, а также терпеливо отвечал на наши письма и вопросы, возникавшие по мере написания книги.

Благодарности от Игоря Симдянова

Редко программиста можно заставить писать связанные комментарии, не говоря уже о техническом тексте. Пробравшись через барьеры языка, технологий, отладки кода, они настолько погружаются в мир кодирования, что вытащить их из него и заинтересовать чем-то другим не представляется возможным.

Излагать свои мысли, "видеть" текст скорее не искусство или особый дар — это работа, которую необходимо осваивать. Выполнять эту работу учатся либо самостоятельно, либо с наставником. Я очень благодарен Зеленцову Сергею Васильевичу, моему науч-

ному руководителю, который потратил безумное количество времени в попытках научить меня писать.

Второй человек, благодаря которому вы смогли увидеть эту и множество других книг, это Максим Кузнецов — наука, война, медицина, психология, химия, физика музыка, программирование — для него не было запретных областей. Он везде был свой и чувствовал себя как рыба в воде. Он быстро жил и жизнь его быстро закончилась. Остались спасенные им люди. Эта книга в том числе и его детище.

Отдельно хотел бы поблагодарить всех друзей-программистов из команды GoPromo (**http://go-promo.ru**), с которыми мне довелось работать последнее время над сайтом газеты "Известия" (**http://izvestia.ru**) и телеканала "ЛайфНьюс" (**http://lifenews.ru**). Их профессионализм, стиль работы сильно повлияли на меня и, конечно же, на содержимое этой книги. Каждого бы хотелось поблагодарить отдельно:

- Вадима Жарко, благодаря которому я наконец-то порвал последние связи с Windows, погрузился в мир UNIX и почувствовал себя по-настоящему "дома";
- Александра Муравкина, который познакомил меня с удивительным миром Ruby и Ruby on Rails. Это переворачивает, это позволяет по-другому смотреть на программирование, в том числе на PHP-программирование;
- Дмитрия Михеева, с которым мы провели много человеко-лет над отладкой легаси-кода и работой над хитроумными SQL-конвертерами, благодаря которым легаси-код превращался в изящные, покрытые тестами и комментариями системы;
- Артема Нистратова, энциклопедические знания буквально обо всем в IT-мире постоянно служили и служат источником вдохновения. Иногда, кажется, нет такой проблемы, с которой он не может справиться, и нет такого языка программирования и сервера, о котором бы он не смог прочитать небольшую лекцию;
- Дениса Максимова, наверное, только благодаря его потрясающей работоспособности и умению глубоко погружаться в любую проблему у меня появилось время на эту книгу;
- Алексея Авдеева, гения фронт-разработки, благодаря которому я еще не теряю способность ориентироваться в мире современных JavaScript-фреймворков;
- Александра Горбунова, чье потрясающее чувство юмора не давало опускать руки при создании этой книги, а способность смотреть на вещи под самыми неожиданными углами, безусловно, оставила неизгладимый след на тексте;
- Александра Бобкова, бесконечное терпение и способность разгребать авгиевы конюшни технологического долга с одной стороны вдохновили написать книгу по современному PHP, а с другой позволили выделить на это время.

Деловая и дружелюбная атмосфера в команде GoPromo, семинары и школа программирования, которую мы ведем для всех желающих, напоминает атмосферу научной лаборатории, в которой началась моя карьера программиста.

Отдельно благодарности заслуживают читатели и посетители форума **http://softtime.ru/forum**: ваши вопросы, замечания и пожелания всегда очень важны. Даже если мы не отвечаем на них сразу, они потом находят отражение на страницах книг.

Конечно же, огромное спасибо моей жене Аленке, которая, обладая неуемным характером, терпеливо ждала, когда будет завершена книга.

Отдельная благодарность Дмитрию Котерову за то, что позволил делать со своей книгой все, что угодно — удалять, добавлять, перемещать главы. Читая несколько лет назад книгу основного "конкурента", мне всегда хотелось "не много" поправить ее. Не думал, что это когда-нибудь будет возможно.

Конечно же, огромная благодарность издательству "БХВ-Петербург" и особенно Евгению Рыбакову, который терпел нас, молодых авторов, все эти годы и продолжает терпеть. Без его усилий и усилий всей команды издательства российское IT-сообщество не досчиталось бы множества книг.

Предисловие к первому изданию

Данная книга является дальнейшим развитием и дополнением издания "Самоучитель PHP 4"[1], переработанным и дополненным новым материалом, касающимся возможностей PHP версии 5. Как много было изменено и добавлено? На этот вопрос можно ответить так. Новые (по сравнению с самоучителем) сведения составляют примерно половину объема данной книги и, главным образом, сконцентрированы в последних трех частях. Большинство глав также подверглись серьезной доработке, в основном направленной на описание новых возможностей PHP 4 и PHP 5, появившихся в языках с момента выхода первого издания.

Конечно, нельзя вести разговор о программировании, не подкрепляя его конкретными примерами на том или ином алгоритмическом языке. Поэтому главная задача книги — подробное описание языка PHP версий 4 и 5, а также некоторых удобных приемов, позволяющих создавать качественные Web-программы за короткие сроки, получая продукты, легко модифицируемые и поддерживаемые в будущем.

Основной объем материала книги применим как к PHP 5, так и к PHP 4. Различия между этими версиями, как правило, оговариваются особо. Тем не менее, пятой версии языка уделяется особое внимание, потому что в ней многие приемы программирования (особенно объектно-ориентированного) выглядят наиболее просто и изящно.

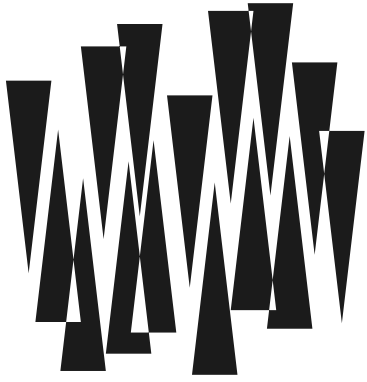
Попутно описываются наиболее часто используемые и полезные на практике приемы Web-программирования вообще, не только на PHP. Авторы постарались рассказать практически обо всем, что потребуется в первую очередь для освоения профессии Web-программиста. Но это вовсе не значит, что книга переполнена всякого рода точной технической информацией, сухими спецификациями, гигантскими таблицами и блок-схемами. Наша мысль направлена не на строгое и академическое изложение материала, а на те практические методы и приемы (часто — неочевидные сами по себе), которые позволят в значительной степени облегчить программирование.

В тексте много общефилософских рассуждений на тему "как могло бы быть, если..." или "что сделали бы сами авторы в этой ситуации...", они обычно оформлены в виде примечаний. Иногда мы позволяем себе писать не о том, что есть *на самом деле*, а как это *могло бы быть* в более благоприятных обстоятельствах. Здесь применяется метод:

---

[1] Котеров Д. В. Самоучитель PHP 4. — СПб.: БХВ-Петербург, 2001.

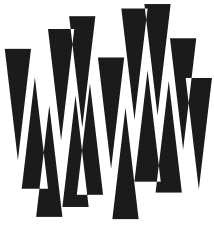
"расскажи сначала просто, пусть и не совсем строго и точно, а затем постепенно детализируй, освещая подробности, опущенные в прошлый раз". По своему опыту знаем, что такой стиль повествования чаще всего оказывается гораздо более плодотворным, чем строгое и сухое описание фактов. Еще раз: авторы не ставили себе целью написать исчерпывающее руководство в определенной области и не стремились описывать все максимально точно, как в учебнике по математике, — наоборот, во многих местах мы пытались отталкиваться от умозрительных рассуждений, возможно, немного и не соответствующих истине. Основной подход — от частного к общему, а не наоборот. Как-никак, "изобретение велосипеда" испокон веков считалось лучшим приемом педагогики.



ЧАСТЬ I

Основы Web-программирования

Глава 1.	Принципы работы Интернета
Глава 2.	Интерфейс CGI и протокол HTTP
Глава 3.	CGI изнутри
Глава 4.	Встроенный сервер PHP



ГЛАВА 1

Принципы работы Интернета

Сеть Интернет представляет собой множество компьютеров, соединенных друг с другом кабелями, а также радиоканалами, спутниковыми каналами и т. д. Однако, как известно, одних проводов или радиоволн для передачи информации недостаточно: передающей и принимающей сторонам необходимо придерживаться ряда соглашений, позволяющих строго регламентировать передачу данных и гарантировать, что эта передача пройдет без искажений. Такой набор правил называется *протоколом передачи*. Упрощенно, протокол — это набор правил, который позволяет системам, взаимодействующим в рамках сети, обмениваться данными в наиболее удобной для них форме. Следуя сложившейся в подобного рода книгах традиции, мы вкратце расскажем, что же собой представляют основные протоколы, используемые в Интернете.

> ***Замечание***
>
> Иногда мы будем называть Интернет "Сетью" с большой буквы, в отличие от "сети" с маленькой буквы, которой обозначается вообще любая сеть, локальная или глобальная. Тут ситуация сходна со словом "галактика": нашу галактику часто называют Галактикой с прописной буквы, а "галактика" со строчной буквы соответствует любой другой звездной системе. На самом деле, сходство Сети и Галактики идет несколько дальше орфографии, и, думаем, вы скоро также проникнетесь этой мыслью.

Протоколы передачи данных

Необходимость некоторой стандартизации появилась чуть ли не с самого момента возникновения компьютерных сетей. Действительно, подчас одной сетью объединены компьютеры, работающие под управлением не только различных операционных систем, но нередко имеющие и совершенно различную архитектуру процессора, организацию памяти и т. д. Именно для того, чтобы обеспечивать передачу между такими компьютерами, и предназначены всевозможные протоколы. Давайте рассмотрим этот вопрос чуть подробнее.

Разумеется, для разных целей существуют различные протоколы. К счастью, нам не нужно иметь представление о каждом из них — достаточно знать только тот, который мы будем использовать в Web-программировании. Таковым для нас является *протокол TCP* (Transmission Control Protocol, протокол управления передачей данных), а точнее, *протокол HTTP* (Hypertext Transfer Protocol, протокол передачи гипертекста), бази-

рующийся на TCP. Протокол HTTP как раз и задействуется браузерами и Web-серверами.

Заметьте, что один протокол может использовать в своей работе другой. В мире Интернета эта ситуация является совершенно обычной. Чаще всего каждый из протоколов, участвующих в передаче данных по сети, реализуется в виде отдельного и по возможности независимого программного обеспечения или драйвера. Среди них существует некоторая иерархия, когда один протокол является всего лишь "надстройкой" над другим, тот, в свою очередь — над третьим, и т. д. до самого "низкоуровневого" драйвера, работающего уже непосредственно на физическом уровне с сетевыми картами или модемами. На рис. 1.1 приведена примерная схема процесса, происходящего при отправке запроса браузером пользователя на некоторый Web-сервер в Интернете. Прямоугольниками обозначены программные компоненты: драйверы протоколов и программы-абоненты (последние выделены жирным шрифтом), направление передачи данных указано стрелками. Конечно, в действительности процесс гораздо сложнее, но нам сейчас нет необходимости на этом останавливаться.

Обратите внимание, что в пределах каждой системы протоколы на схеме расположены в виде "стопки", один над другим. Такая структура обуславливает то, что часто семейство протоколов обмена данными в Интернете называют *стеком TCP/IP* (стек в переводе с английского как раз и обозначает "стопку").

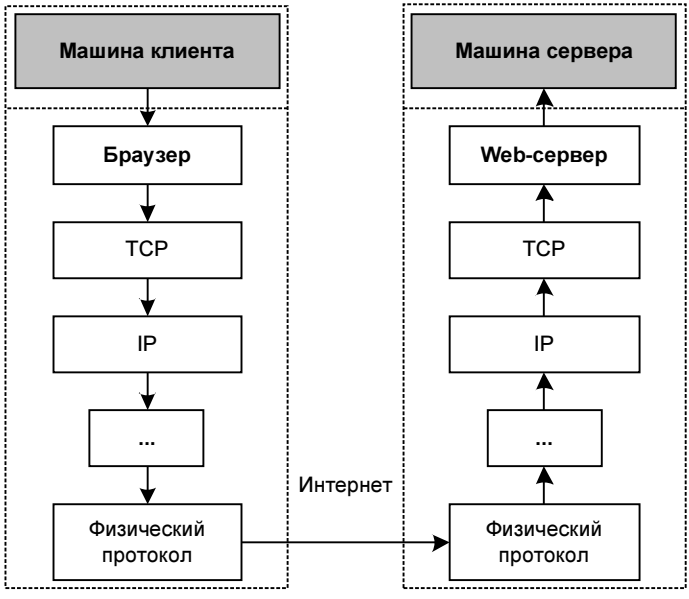


Рис. 1.1. Организация обмена данными в Интернете

Каждый из протоколов в идеале "ничего не знает" о том, какой протокол "стоит над ним". Скажем, протокол IP (который обеспечивает несколько более простой сервис по сравнению с TCP, например, не гарантирует доставку сообщения адресату) не использует возможности протокола TCP, а TCP, в свою очередь, "не догадывается" о существовании протокола HTTP (именно его задействует браузер и понимает Web-сервер; на схеме протокол HTTP не обозначен).

Применение такой организации позволяет заметно упростить ту часть операционной системы, которая отвечает за поддержку работы с сетью. Но не пугайтесь. Нас будет интересовать в конечном итоге всего лишь протокол самого высокого уровня, "возвышающийся" над всеми остальными протоколами, т. е. HTTP и то, как он взаимодействует с протоколом TCP.

Семейство TCP/IP

Вот уже несколько десятков лет основной протокол Интернета — TCP. Как часто бывает, этот выбор обусловлен скорее историческими причинами, нежели действительными преимуществами протокола (впрочем, преимуществ у TCP также предостаточно). Он ни в коей мере не претендует на роль низкоуровневого — наоборот, в свою работу TCP вовлекает другие протоколы, например *IP* (в свою очередь, IP также базируется на услугах, предоставляемых некоторыми другими протоколами). Протоколы TCP и IP настолько сильно связаны, что принято объединять их в одну группу под названием *семейство TCP/IP* (в него включается также протокол UDP, рассмотрение которого выходит за рамки этой книги). Приведем основные особенности протокола TCP, входящего в семейство.

- Корректная доставка данных до места назначения гарантируется — разумеется, если такая доставка вообще возможна. Даже если связь не вполне надежна (например, на линии помехи оттого, что в кабель попала вода, замерзшая зимой и разорвавшая оболочку провода), "потерянные" фрагменты данных посылаются снова и снова до тех пор, пока вся информация не будет передана.
- Передаваемая информация представлена в виде потока — наподобие того, как осуществляется обмен с файлами практически во всех операционных системах. Иными словами, мы можем "открыть" соединение и затем выполнять с ним те же самые операции, к каким привыкли при работе с файлами. Таким образом, программы на разных машинах (возможно, находящихся за тысячи километров друг от друга), подключенных к Интернету, обмениваются данными так же непринужденно, как и расположенные на одном компьютере.
- Протокол TCP/IP устроен так, что он способен выбрать оптимальный путь распространения сигнала между передающей и принимающей стороной, даже если сигнал проходит через сотни промежуточных компьютеров. В последнем случае система выбирает путь, по которому данные могут быть переданы за минимальное время, основываясь при этом на статистическую информацию работы сети и так называемые таблицы маршрутизации.
- При передаче данные разбиваются на фрагменты — пакеты, которые и доставляются в место назначения по отдельности. Разные пакеты вполне могут следовать различными маршрутами в Интернете (особенно если их путь пролегает через десятки серверов), но для всех них гарантирована правильная "сборка" в месте назначения (в нужном порядке). Как уже упоминалось, принимающая сторона в случае обнаружения недостачи пакета запрашивает передающую систему, чтобы та передала его еще раз. Все это происходит незаметно для программного обеспечения, эксплуатирующего TCP/IP.

В Web-программировании нам вряд ли придется работать с TCP/IP напрямую (разве что в очень экзотических случаях), обычно можно использовать более высокоуровне-

вые "языки", например HTTP, служащий для обмена информацией между сервером и браузером. Собственно, этому протоколу посвящена значительная часть книги. Его мы рассмотрим подробно чуть позже. А пока поговорим еще немного о том, что касается TCP/IP, чтобы не возвращаться к этому впоследствии.

Адресация в Сети

Машин в Интернете много, а будет — еще больше. Так что вопрос о том, как можно их эффективно идентифицировать в пределах всей сети, оказывается далеко не праздным. Кроме того, практически все современные операционные системы работают в многозадачном режиме (поддерживают одновременную работу нескольких программ). Это значит, что возникает также вопрос о том, как нам идентифицировать конкретную систему или программу, желающую обмениваться данными через Сеть. Эти две задачи решаются стеком TCP/IP при помощи IP-адреса и номера порта. Давайте посмотрим, как.

IP-адрес

Любой компьютер, подключенный к Интернету и желающий обмениваться информацией со своими "сородичами", должен иметь некоторое уникальное имя, или *IP-адрес*. Вот уже 30 лет среднестатистический IP-адрес выглядит примерно так:

127.12.232.56

Как мы видим, это — четыре 8-разрядных числа (принадлежащих диапазону от 0 до 255 включительно), разделенные точками. Не все числа допустимы в записи IP-адреса: ряд из них используется в служебных целях (например, адрес 127.0.0.1 (его еще часто называют localhost) выделен для обращения к локальной машине — той, на которой был произведен запрос, а число 255 соответствует широковещательной рассылке в пределах текущей подсети). Мы не будем здесь обсуждать эти исключения детально. Возникает вопрос: ведь компьютеров и устройств в Интернете миллиарды (а прогнозируются десятки миллиардов), как же мы, простые пользователи, запросив IP-адрес машины, в считанные секунды с ней соединяемся? Как "оно" (и что это за "оно"?) узнаёт, где на самом деле расположен компьютер и устанавливает с ним связь, а в случае неверного адреса адекватно на это реагирует? Вопрос актуален, поскольку машина, с которой мы собираемся связаться, вполне может находиться за океаном, и путь к ней пролегает через множество промежуточных серверов.

В деталях вопрос определения пути к адресату довольно сложен. Однако нетрудно представить себе общую картину, точнее, некоторую ее модель. Предположим, что у нас есть 1 миллиард (10^9) компьютеров, каждый из которых напрямую соединен с 11 (к примеру) другими через кабели. Получается этакая паутина из кабелей, не так ли? Кстати, это объясняет, почему одна из наиболее популярных служб Интернета, базирующаяся на протоколе HTTP, названа *WWW* (World Wide Web, или Всемирная паутина).

> ***Замечание***
>
> Следует заметить, что в реальных условиях, конечно же, компьютеры не соединяют друг с другом таким большим количеством каналов. Вместо этого применяются всевозможные внутренние таблицы, которые позволяют компьютеру "знать", где конкретно располагаются

некоторые ближайшие его соседи. То есть любая машина в Сети имеет информацию о том, через какие узлы должен пройти сигнал, чтобы достигнуть самого близкого к ней адресата. А если не обладает этими знаниями, то получает их у ближайшего "сородича" в момент загрузки операционной системы. Разумеется, размер таких таблиц ограничен и они не могут содержать маршруты до всех машин в Интернете (хотя в самом начале развития Интернета, когда компьютеров в Сети было немного, именно так и обстояло дело). Потому-то мы и проводим аналогию с одиннадцатью соседями.

Итак, мы сидим за компьютером номер 1 и желаем соединиться с машиной example.com с некоторым IP-адресом. Мы даем нашему компьютеру запрос: выясни-ка у своих соседей, не знают ли они чего о example.com. Он рассылает в одиннадцать сторон этот запрос (считаем, что это занимает 0,1 с, т. к. все происходит практически одновременно — размер запроса не настолько велик, чтобы сказалась задержка передачи данных), и ждет, что ему ответят.

Что же происходит дальше? Нетрудно догадаться. Каждый из компьютеров окружения действует по точно такому же плану. Он спрашивает у *своих* десятерых соседей, не слышали ли они чего о example.com. Это, в свою очередь, занимает еще 0,1 с. Что же мы имеем? Всего за 0,2 с проверено уже $11 \times 10 = 110$ компьютеров. Но это еще не все, ведь процесс нарастает лавинообразно. Нетрудно подсчитать, что за время порядка 1 с мы "разбудим" 11×10^9 машин, т. е. в 11 раз больше, чем мы имеем!

Конечно, на самом деле процесс будет идти медленнее: с одной стороны, какие-то системы могут быть заняты и не ответят сразу. С другой стороны, мы должны иметь механизм, который бы обеспечивал, чтобы одна машина не "опрашивалась" многократно. Но все равно, согласитесь, результаты впечатляют, даже если их и придется занизить для реальных условий хоть в 100 раз.

Замечание

В действительности дело обстоит куда сложнее. Отличия от представленной схемы частично заключаются в том, что компьютеру совсем не обязательно "запрашивать" всех своих соседей — достаточно ограничиться только некоторыми из них. Для ускорения доступа все возможные IP-адреса делятся на четыре группы — так называемые адреса подсетей классов A, B, C и D. Но для нас сейчас это не представляет никакого интереса, поэтому не будем задерживаться на деталях. О TCP/IP можно написать целые тома (что и делается).

Версии протокола IP

Если во время расцвета Интернета в 1990-х годах любой пользователь, получал свой персональный IP-адрес, и даже мог закрепить его за собой, то со временем адресов стало не хватать. В настоящий момент наблюдается дефицит адресного пространства IP-адресов.

Благодаря ужесточению правил их выдачи, введению новых способов организации адресного пространства (NAT, CIDR) ситуацию удается пока держать под контролем. По оптимистичным прогнозам, растягивать адресное пространство можно до 2050 года.

В любом случае IP-адрес — дефицитный ресурс, получить который не просто. Поэтому в ближайшие годы нас ожидает смена версии протокола с текущей IPv4 на новую IPv6, в которой адрес расширяется с 32 до 128 бит (шестнадцать 8-разрядных чисел). С переходом на IPv6 будет достаточно выделяемых персональных IP-адресов каждому жителю планеты и каждому производимому устройству.

Современные операционные системы и программное обеспечение уже подготовлены для такого перехода. Многие компании уже используют IPv6 для организации своих внутренних сетей.

В процессе знакомства с Web-программированием вам постоянно будут встречаться IP-адреса в новом IPv6-формате, например:

```
2a03:f480:1:23::ca
::1
```

IP-адрес в IPv4-формате разбивается на 8-битные группы, разделенные точкой. В IPv6 запись в десятичном формате была бы слишком громоздкой, поэтому 128-битный адрес разбивается на 16-битные группы, разделяемые двоеточием. Цифры представляются в шестнадцатеричном формате, т. е. изменяются не только от 0 до 9, но и от A до F. Ведущие нули в группах отбрасываются, поэтому, если в адресе встречается запись `00ca`, она заменяется записью `ca`. Более того, группы, полностью состоящие из нулей, сжимаются до символа `::`. Таким образом, полный IPv6-адрес:

```
2a03:f480:0001:0023:0000:0000:0000:00ca
```

сначала сокращается до:

```
2a03:f480:1:23:0:0:0:ca
```

а затем до:

```
2a03:f480:1:23::ca
```

IPv4-адрес локального хоста 127.0.0.1 соответствует IPv6-адресу

```
0000:0000:0000:0000:0000:0000:0000:0001
```

В краткой форме его можно записать как

```
::1
```

Доменное имя

И все-таки обычным людям довольно неудобно работать с IP-представлением адреса, особенно с новыми IPv6-адресами. Действительно, куда как проще запомнить символьное имя, чем набор чисел. Чтобы облегчить простым пользователям работу с Интернетом, придумали систему *DNS* (Domain Name System, служба имен доменов).

> ***Замечание***
>
> Общемировая DNS представляет собой распределенную базу данных, способную преобразовать доменные имена машин в их IP-адреса. Это не так-то просто, учитывая, что Интернет насчитывает миллиарды компьютеров. Поэтому мы не будем в деталях рассматривать то, как работает служба DNS, а займемся больше практической стороной вопроса.

Итак, при использовании DNS любой компьютер в Сети может иметь не только IP-адрес, но также и символическое имя. Выглядит оно примерно так:

www.example.msu.ru

То есть это набор слов (их число произвольно), опять же разделенных точкой. Каждое такое сочетание слов называется *доменом N-го уровня* (например, **ru** — домен первого уровня, **msu.ru** — второго, **example.msu.ru** — третьего и т. д.) — рис. 1.2.

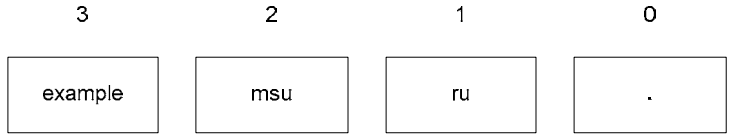


Рис. 1.2. Домены нулевого, первого, второго и третьего уровня

Вообще говоря, полное DNS-имя выглядит немного не так: в его конце обязательно стоит точка, например:

www.example.msu.ru.

Именно такое (вообще-то, и только такое) представление является правильным, но браузеры и другие программы часто позволяют нам опускать завершающую точку. В принятой нами терминологии будем называть эту точку *доменом нулевого уровня*, или *корневым доменом*.

Нужно заметить, что одному и тому же IP-адресу вполне может соответствовать сразу несколько доменных имен. Каждое из них ведет в одно и то же место — к единственному IP-адресу. Благодаря протоколу *HTTP 1.1* (мы вскоре кратко рассмотрим его особенности) Web-сервер, установленный на машине и откликающийся на какой-либо запрос, способен узнать, какое доменное имя ввел пользователь, и соответствующим образом среагировать, даже если его IP-адресу соответствует несколько доменных имен. В последнее время HTTP 1.1 применяется практически повсеместно — не то, что несколько лет назад, поэтому все больше и больше серверов используют его в качестве основного протокола для доступа к Web.

Интересен также случай, когда одному и тому же DNS-имени сопоставлены несколько разных IP-адресов. В этом случае служба DNS автоматически выбирает тот из адресов, который, по ее мнению, ближе всего расположен к клиенту, или который давно не использовался, или же наименее загружен (впрочем, последняя оценка может быть весьма и весьма субъективна). Эта возможность часто задействуется, когда Web-сервер становится очень большим (точнее, когда число его клиентов начинает превышать некоторый предел) и его приходится обслуживать сразу нескольким компьютерам. Такая схема используется, например, на сайте компании Netscape.

Как же ведется поиск по DNS-адресу? Для начала он преобразуется специальными DNS-серверами, раскиданными по всему миру, в IP-адрес. Давайте посмотрим, как это происходит. Пусть клиентом выдан запрос на определение IP-адреса машины **www.example.com.** (еще раз обратите внимание на завершающую точку — это не конец предложения). Чтобы его обработать, первым делом посылается запрос к так называемому корневому домену (точнее, к программе — DNS-серверу, запущенному на этом домене), который имеет имя "**.**" (на самом деле его база данных распределена по нескольким компьютерам, но для нас это сейчас несущественно). Запрос содержит команду: вернуть IP-адрес машины (точнее, IP-адрес DNS-сервера), на котором расположена информация о домене **com**. Как только IP-адрес получен, по нему происходит аналогичное обращение с просьбой определить адрес, соответствующий домену **example** внутри домена **com** внутри корневого домена "**.**". В конце у предпоследней машины запрашивается IP-адрес поддомена **www** в домене **example.com.**.

Каждый домен "знает" все о своих поддоменах, а те, в свою очередь, — о своих, т. е. система имеет некоторую иерархичность. Корневой домен, как мы уже заметили, при-