O'REILLY®

2-е издание

Разработка веб-приложений на WordPress

WordPress как фреймворк

Брайан Мессенленер
Джейсон Коулман

bhv®

УДК 004.4'236
ББК 32.973.26-018
М53

Мессенленер, Б.

М53 Разработка веб-приложений на WordPress: Пер. с англ. / Б. Мессенленер, Д. Коулман. — 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2021. — 528 с.: ил.

ISBN 978-5-9775-6753-4

Подробно рассматривается создание веб-приложений на платформе WordPress, в том числе для мобильных устройств, принципы работы таких приложений. Описана структура каталогов и базы данных, приведены типы записей, метаданных и таксономий, перечислены основные классы и функции. Уделено внимание разработке собственной темы оформления с адаптивным дизайном. Рассказывается о типах пользователей и их ролях в архитектуре WordPress. Отдельная глава посвящена работе с API-интерфейсами, объектами и вспомогательными функциями, рассматриваются вопросы безопасности веб-приложений, принципы написания безопасного кода. Изучается REST API в WordPress, JavaScript-фреймворки, способы локализации приложений. Описаны принципы построения многосайтовых сетей, оптимизации и масштабирования. Рассматривается проект Gutenberg и его возможности.

Во втором издании авторы рассматривают новые функции и возможности актуальных версий WordPress. Все примеры кода из книги доступны на веб-сервисе GitHub.

Для веб-разработчиков

УДК 004.4'236
ББК 32.973.26-018

Группа подготовки издания:

Руководитель проекта	*Павел Шалин*
Зав. редакцией	*Людмила Гауль*
Перевод с английского	*Михаила Райтмана*
Компьютерная верстка	*Натальи Смирновой*
Оформление обложки	*Карины Соловьевой*

"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

ISBN 978-1-491-99008-7 (англ.)
ISBN 978-5-9775-6753-4 (рус.)

Оглавление

Вступительное слово **15**

Предисловие **17**
Для кого предназначена эта книга 17
Для кого НЕ предназначена эта книга 18
Структура книги 18
О коде программ 20
Условные обозначения 20
Использование примеров программного кода 21

Благодарности **23**

ГЛАВА 1. Создание веб-приложений с помощью WordPress **25**
Что такое веб-сайт? 25
Что такое приложение? 25
Что такое веб-приложение? 25
Функции веб-приложения 26
Мобильные приложения 27
Прогрессивные веб-приложения 28
Зачем нужен WordPress? 29
Вы уже используете WordPress 29
С помощью WordPress легко управлять контентом 29
WordPress позволяет просто и безопасно управлять пользователями 30
Плагины 30
Гибкость важна 31
Частые обновления безопасности 31
Стоимость 32
Ответы на некоторые распространенные критические мнения о WordPress 32
Когда не следует использовать WordPress 35
Вы планируете лицензировать или продавать технологию своего сайта 35
Имеется другая платформа, которая приведет вас к цели быстрее 36
Гибкость не важна для вас 36
Ваше приложение должно работать в режиме реального времени 37
WordPress как фреймворк 37
WordPress и фреймворки Framework-View-Controller 38
Анатомия приложения WordPress 40
Что такое SchoolPress? 41
SchoolPress работает в многосайтовой сети WordPress 41
Бизнес-модель SchoolPress 41
Уровни участия и роли пользователей 42
Классы — это группы BuddyPress 42
Назначения — это CPT 42
Представления (подтип) CPT для назначений 43
Семестры являются таксономией для класса CPT 43

Департаменты являются таксономией для класса CPT ... 43
SchoolPress имеет один основной пользовательский плагин ... 43
В SchoolPress есть несколько других пользовательских плагинов ... 44
SchoolPress использует тему Memberlite ... 44

ГЛАВА 2. Основы WordPress ... 46
Структура каталогов WordPress ... 46
Корневой каталог ... 46
Структура базы данных WordPress ... 48
Таблица *wp_options* ... 49
Функции в каталоге /wp-includes/option.php ... 49
Таблица *wp_users* ... 51
Функции в каталоге /wp-includes/ ... 52
Таблица *wp_usermeta* ... 55
Таблица *wp_posts* ... 59
Функции в каталоге /wp-includes/post.php ... 60
Таблица *wp_postmeta* ... 64
Функции из каталога /wp-includes/post.php ... 64
Таблица *wp_comments* ... 68
Функции в каталоге /wp-includes/comment.php ... 69
Таблица *wp_commentsmeta* ... 73
Функции из каталога /wp-includes/comment.php ... 73
Таблица *wp_terms* ... 75
Функции в каталоге /wp-includes/taxonomy.php ... 76
Таблица *wp_termmeta* ... 79
Таблица *wp_term_taxonomy* ... 81
Функции в каталоге /wp-includes/taxonomy.php ... 81
Таблица *wp_term_relationships* ... 82
Хуки: события и фильтры ... 83
События ... 84
Фильтры ... 85
Среды разработки и хостинг ... 86
Работа локально ... 86
Выбор веб-хостинга ... 87
Среды разработки, интеграции и доставки ... 87
Расширение WordPress ... 88

ГЛАВА 3. Использование плагинов WordPress ... 89
General Public License, версия 2 ... 90
Установка плагинов WordPress ... 90
Создание собственного плагина ... 91
Структура файла плагина приложения ... 92
Каталог /adminpages/ ... 93
Каталог /classes/ ... 94
Каталог /css/ ... 94
Каталог /js/ ... 95
Каталог /images/ ... 96
Каталог /includes/ ... 96
Каталог /includes/lib/ ... 97
Каталог /pages/ ... 97

Каталоги /services/ и /scheduled/ ... 98
Файл schoolpress.php ... 98
Дополнения к существующим плагинам ... 99
Случаи из практики и примеры ... 99
Цикл WordPress ... 100
Глобальные переменные WordPress ... 100
Бесплатные плагины ... 111
Admin Columns ... 111
Advanced Custom Fields ... 111
BadgeOS ... 112
Posts 2 Posts ... 112
Members ... 113
W3 Total Cache ... 113
Yoast SEO ... 113
Премиальные плагины ... 114
Gravity Forms ... 114
BackupBuddy ... 114
WP All Import ... 115
Плагины сообщества ... 115
BuddyPress ... 115

ГЛАВА 4. Темы ... 128
Темы и плагины ... 128
Где разместить код при разработке приложений ... 128
Где разместить код при разработке плагинов ... 129
Где разместить код при разработке тем ... 129
Иерархия шаблонов ... 130
Шаблоны страниц ... 131
Образец шаблона страницы ... 131
Использование хуков для копирования шаблонов ... 134
Когда следует использовать шаблон темы? ... 136
Функции WordPress для работы с темами ... 136
Использование переменной *locate_template* в плагинах ... 137
Файл style.css ... 139
Создание версий CSS-файлов вашей темы ... 140
Файл functions.php ... 141
Темы и CPT ... 142
Популярные фреймворки для разработки тем ... 142
Фреймворки тем WordPress ... 142
Сторонние фреймворки тем ... 144
Создание дочерней темы для Memberlite ... 144
Включение Bootstrap в тему вашего приложения ... 145
Меню ... 146
Навигационные меню ... 147
Динамические меню ... 148
Адаптивный дизайн ... 148
Определение устройства и дисплея с помощью CSS ... 149
Определение устройств и их свойств в JavaScript ... 150
Определение устройства в PHP ... 153
Последнее замечание по определению браузера ... 157

ГЛАВА 5. Пользовательские типы записей, метаданные записей и таксономия.... 158
Типы сообщений по умолчанию и CPT ... 158
Страница ... 158
Публикация ... 158
Вложение ... 158
Редакции ... 159
Элемент меню навигации ... 159
Пользовательский CSS ... 159
Наборы изменений ... 159
Кеш oEmbed ... 159
Пользовательские запросы ... 160
Повторно используемые блоки ... 160
Определение и регистрация CPT ... 160
Функция *register_post_type($post_type, $args)* ... 161
Что такое таксономия и как ее использовать? ... 168
Таксономии и метаданные постов ... 169
Создание пользовательских таксономий ... 171
Функция *register_taxonomy($taxonomy, $object_type, $args)* ... 171
Функция *register_taxonomy_for_object_type($taxonomy, $object_type)* ... 174
Использование CPT и таксономий в ваших темах и плагинах ... 175
Тема архива и файлы шаблона Single ... 175
Старый добрый класс *WP_Query* и метод *get_posts()* ... 175
Метаданные и CPT ... 179
Функция *add_meta_box($id, $title, $callback, $screen, $context, $priority, $callback_args)* ... 179
Использование метаблоков в редакторе блоков Block Editor ... 182
Пользовательские классы-оболочки для CPT ... 183
Расширение класса *WP_Post* в сравнении с созданием класса-обертки ... 186
Зачем нужны классы Wrapper? ... 186
Держите CPT и таксономии вместе ... 186
Держите все в классе-обертке ... 188
Классы Wrapper читаются лучше ... 190

ГЛАВА 6. Пользователи, их роли и возможности ... 191
Получение данных пользователей ... 192
Добавляем, обновляем и удаляем пользователей ... 194
Хуки и фильтры ... 198
Что такое роли и возможности? ... 199
Проверка роли и возможностей пользователя ... 200
Создание собственных ролей и возможностей ... 202
Расширение класса *WP_User* ... 204
Добавление полей регистрации и профиля ... 206
Настройка таблицы пользователей на административной панели ... 211
Плагины ... 213
Theme My Login ... 213
Hide the Admin Bar ... 213
Paid Memberships Pro ... 213
PMPro Register Helper ... 214
Members ... 214
WP User Fields ... 215

ГЛАВА 7. Работа с API-интерфейсами WordPress, объектами и вспомогательными функциями 216
API шорткодов 216
Атрибуты шорткода 217
Вложенные шорткоды 218
Удаление шорткодов 219
Другие полезные функции, связанные с шорткодами 220
API виджетов 220
Прежде чем добавить свой собственный виджет 221
Добавление виджетов 221
Определение области виджета 225
Встраивание виджета вне динамической боковой панели 227
Удаление виджетов с панели инструментов 229
Добавление собственного виджета на панель инструментов 231
API настроек 234
Вам действительно нужна страница настроек? 234
Не могли бы вы использовать вместо этого хук или фильтр? 235
Учет стандартов при добавлении настроек 236
Игнорирование стандартов при добавлении настроек 237
API перезаписи 238
Добавление правил перезаписи 238
Сброс правил перезаписи 239
Другие функции перезаписи 240
Функция *WP-Cron* 243
Добавление своих интервалов 245
Планирование единичных событий 246
Запуск заданий Cron с сервера 246
Использование только серверного Cron 247
Функция *WP Mail* 248
Отправка более приятных писем с помощью WordPress 249
API заголовка файла 250
Добавление заголовков файлов в ваши собственные файлы 253
Добавление новых заголовков в плагины и темы 254
Heartbeat API 255

ГЛАВА 8. Безопасность в WordPress 260
Почему это важно 260
Основные меры безопасности 261
Регулярно выполняйте обновление 261
Не используйте имя пользователя "admin" 261
Выбирайте надежный пароль 261
Примеры плохих паролей 262
Примеры хороших паролей 262
Усиление защиты в WordPress 263
Запретите администраторам редактировать плагины и темы 263
Измените префикс таблиц базы данных 263
Переместите в другое место файл wp-config.php 264
Не отображайте сообщения об ошибках авторизации 265
Не отображайте номер версии WordPress 265
Исключите возможность авторизации через страницу wp-login.php 266

Добавьте в файл *.htaccess кастомные правила, блокирующие доступ к каталогу wp-admin ... 267
SSL-сертификаты и HTTPS ... 268
Установка SSL-сертификата на сервере ... 268
Авторизация и доступ к панели администратора WordPress по протоколу SSL ... 271
Отладка проблем с протоколом HTTPS ... 272
"Атомный" способ устранения ошибок протокола SSL ... 273
Резервируйте все! ... 275
Сканируйте, сканируйте и еще раз сканируйте! ... 275
Полезные плагин для обеспечения безопасности ... 276
Плагины для блокировки спама ... 276
Плагины для резервного копирования ... 276
Плагины-брандмауэры/сканеры ... 277
Плагины для защиты авторизации и пароля ... 278
Написание безопасного кода ... 278
Проверяйте полномочия пользователей ... 278
Кастомные инструкции SQL ... 280
Валидация, санация и экранирование данных ... 280
Одноразовые коды ... 284

ГЛАВА 9. JavaScript-фреймворки и рабочий процесс ... 291
Что такое ECMAScript? ... 292
Что такое ES6? ... 293
Что такое ES9? ... 293
Что такое ESNext? ... 293
Что такое Ajax? ... 293
Что такое JSON? ... 293
jQuery и WordPress ... 294
Подключение других JavaScript-библиотек ... 295
Где следует размещать кастомный JavaScript-код ... 296
Ajax-вызовы в WordPress с использованием jQuery ... 297
Управление количеством Ajax-запросов ... 302
Heartbeat API ... 304
Инициализация ... 304
Клиентский JavaScript-код ... 305
Серверный PHP-код ... 306
Инициализация ... 307
Клиентский JavaScript-код ... 308
Серверный PHP-код ... 309
Ограничения WordPress в плане асинхронной обработки ... 310
JavaScript-фреймворки ... 311
Backbone.js ... 311
React ... 312

ГЛАВА 10. REST API в WordPress ... 314
Что такое REST API? ... 314
API ... 314
REST ... 315
JSON ... 315
HTTP ... 315

Зачем нужен REST API в WordPress? ... 318
Использование WordPress REST API версии 2 ... 320
Обнаружение ... 320
Аутентификация ... 320
Маршруты и конечные точки ... 326
Запросы ... 326
Ответы ... 328
Добавление собственных маршрутов и конечных точек ... 329
Функция *register_rest_route($namespace, $route, $args, $override)* ... 329
Настройка плагина WordPress Single Sign-On ... 330
Добавление маршрута /wp-sso/v1/check ... 330
Подключение к нашему плагину базовой аутентификации ... 332
Использование настроенной нами конечной точки для проверки учетных данных пользователя ... 333
Популярные плагины, использующие WordPress REST API ... 334
WooCommerce ... 335
BuddyPress ... 336
Paid Memberships Pro ... 338

ГЛАВА 11. Проект Gutenberg, блоки и кастомные типы блоков ... 342
Редактор системы WordPress ... 343
Плагин Classic Editor ... 344
Блоки для контента и дизайна ... 344
Блоки для представления функциональности ... 344
Создание собственных блоков ... 345
Пример простейшего блока ... 345
Использование кастомных блоков для разработки интерфейсов приложений ... 347
Активация редактора блоков для кастомных типов постов ... 347
Категории блоков ... 348
Блоки домашнего задания ... 349
Ограничение типа блоков до определенных кастомных типов постов ... 349
Ограничение кастомного типа постов до определенных блоков ... 350
Шаблоны блоков ... 351
Сохранение данных блока в метаданных поста ... 353
Советы ... 354
Активируйте режим отладки с помощью константы *WP_SCRIPT_DEBUG* ... 354
Задавайте версию скрипта с помощью функции *filemtime()* ... 355
Дополнительные советы ... 355
Глубже изучите JavaScript, Node.js и React ... 355

ГЛАВА 12. Многосайтовые сети в WordPress ... 357
Когда целесообразна многосайтовость? ... 357
Когда лучше отказаться от многосайтовости? ... 358
Альтернативы многосайтового режима ... 359
Множество авторов или категорий на одном и том же WordPress-сайте ... 359
Кастомные типы постов ... 359
Абсолютно самостоятельные сайты ... 360
Сервис обслуживания WordPress-сайтов ... 360
Мультиарендность ... 360
Настройка многосайтовой сети ... 360

Администрирование многосайтовой сети ... 363
Панель администратора ... 363
Сайты ... 363
Пользователи ... 364
Темы ... 365
Плагины ... 365
Настройки ... 365
Обновления ... 366
Структура базы данных многосайтовой сети ... 367
Общесетевые таблицы ... 367
Индивидуальные таблицы сайтов ... 369
Совместно используемые таблицы сайтов ... 370
Сопоставление доменов ... 370
Некоторые полезные плагины для многосайтового режима ... 371
Расширение User Registration для плагина Gravity Forms ... 372
Расширение Member Network Sites для плагина Paid Memberships Pro ... 372
Плагин More Privacy Options ... 372
Плагин Multisite Global Media ... 372
Плагин Multisite Plugin Manager ... 372
Плагин Multisite Global Search ... 373
Плагин Multisite Robots.txt Manager ... 373
Плагин NS Cloner: Site Copier ... 373
Плагин WP Multi Network ... 373
Основная функциональность многосайтовости ... 373
Переменная *$blog_id* ... 374
Функция *is_multisite()* ... 374
Функция *get_current_blog_id()* ... 374
Функция *switch_to_blog($new_blog)* ... 375
Функция *restore_current_blog()* ... 375
Функция *get_blog_details($fields = null, $get_all = true)* ... 376
Функция *update_blog_details($blog_id, $details = array())* ... 377
Функция *get_blog_status($id, $pref)* ... 378
Функция *update_blog_status($blog_id, $pref, $value)* ... 378
Функция *get_blog_option($id, $option, $default = false)* ... 378
Функция *update_blog_option($id, $option, $value)* ... 379
Функция *delete_blog_option($id, $option)* ... 379
Функция *get_blog_post($blog_id, $post_id)* ... 379
Функция *add_user_to_blog($blog_id, $user_id, $role)* ... 380
Функция *wpmu_delete_user($user_id)* ... 380
Функция *create_empty_blog ($domain, $path, $weblog_title, $site_id = 1)* ... 381
Не упомянутые здесь функции ... 381

ГЛАВА 13. Локализация приложений WordPress ... 382
Нужна ли локализация вашему приложению? ... 382
Как выполняется локализация в WordPress ... 383
Определение локали в WordPress ... 383
Текстовые домены ... 384
Настройка текстового домена ... 384
Подготовка строк с помощью функций перевода ... 386
Функция *__($text, $domain = "default")* ... 387
Функция *_e($text, $domain = "default")* ... 387

Функция *_x($text, $context, $domain = "default")* ... 388
Функция *_ex($title, $context, $domain = "default")* ... 388
Сочетание перевода с экранированием ... 389
Создание и загрузка файлов перевода ... 389
Организация файлов локализации ... 390
Генерирование файла *.pot ... 390
Создание файла *.po ... 392
Создание файла *.mo ... 392
GlotPress ... 392
Использование GlotPress для ваших плагинов и тем в репозитории WordPress.org ... 393
Создание собственного сервера GlotPress ... 393

ГЛАВА 14. Оптимизация и масштабирование WordPress ... 394
Терминология ... 394
Источник или внешнее окружение? ... 396
Тестирование ... 396
Что следует тестировать ... 397
Панель отладки браузера Chrome ... 399
Инструмент Site Health системы WordPress ... 401
Apache Bench ... 402
Siege ... 409
W3 Total Cache ... 410
Настройки страничного кэширования ... 411
Минимизация ... 413
Кэширование базы данных ... 414
Объектное кэширование ... 414
Сети доставки контента ... 415
GZIP-сжатие ... 415
Хостинг ... 415
Хостинги, специально предназначенные для WordPress-сайтов ... 416
Развертывание собственного сервера ... 416
Выборочное кэширование ... 430
API для работы с транзиентами ... 431
Транзиенты в многосайтовом режиме ... 434
Повышение производительности с помощью JavaScript-кода ... 434
Кастомные таблицы ... 436
Действие в обход WordPress ... 438

ГЛАВА 15. Электронная коммерция ... 440
Выбор плагина ... 440
WooCommerce ... 440
Paid Memberships Pro ... 443
Easy Digital Downloads ... 444
Платежные системы ... 447
Торговые счета ... 448
Настройка модели SaaS с помощью Paid Memberships Pro ... 449
Модель SaaS ... 449

ГЛАВА 16. Мобильные приложения на платформе WordPress ... 464
Сценарии использования мобильных приложений ... 464

Нативные и гибридные мобильные приложения ... 465
Что такое нативное мобильное приложение? ... 465
Что такое гибридное мобильное приложение? ... 466
Почему стоит создавать гибридные приложения вместо нативных? ... 466
Cordova ... 467
Ionic Framework ... 471
Приложение-обертка ... 473
AppPresser ... 473

ГЛАВА 17. PHP-библиотеки, интеграция веб-сервисов и миграция с других платформ ... 488
PHP-библиотеки ... 488
Генерация и модификация изображений ... 489
Генерация PDF ... 491
Геолокация и геотаргетинг ... 494
Сжатие и архивация данных ... 496
Инструменты для разработки ... 500
Внешние API-интерфейсы и веб-сервисы ... 502
Elasticsearch ... 502
ElasticPress by 10up ... 502
Google Vision ... 503
Google Maps ... 503
Google Translate ... 504
Twilio ... 504
Другие популярные API-интерфейсы ... 505
Миграция ... 506
Миграция между серверами ... 507
Миграция между платформами ... 508
Руководство по привязке данных ... 510

ГЛАВА 18. Взгляд в будущее ... 511
Оглядываясь назад ... 511
REST API ... 512
Плагины WordPress будут уделять больше внимания API-интерфейсам ... 512
"Обезглавленные" версии WordPress ... 512
GraphQL ... 513
Gutenberg ... 514
Интерфейс администратора перейдет на React/Gutenberg ... 514
Gutenberg будет применяться для редактирования контента на клиентской стороне WordPress ... 514
Шаблоны блоков заменят темы оформления ... 514
Блоки заменят плагины ... 515
Доля WordPress на рынке будет колебаться ... 515
WordPress станет более популярной платформой для мобильной разработки ... 516
WordPress будет оставаться хорошим выбором для разработки любого рода приложений ... 516

Об авторах ... 517

Предметный указатель ... 518

Создание веб-приложений с помощью WordPress

Эта книга поможет вам создать что угодно с помощью WordPress: веб-сайты, темы, плагины, веб-сервисы и веб-приложения. Мы решили сосредоточиться на веб-приложениях, потому что вы можете рассматривать их как супер-сайты, использующие все методы, которые мы опишем.

Многие считают, что WordPress недостаточно мощен или не предназначен для создания веб-приложений; далее мы еще вернемся к этому вопросу. Мы много лет создавали веб-приложения на WordPress и знаем, что с его помощью вы можете создавать масштабируемые приложения.

В этой главе мы начнем с определения того, что такое веб-приложение, а затем выясним, почему WordPress является отличным фреймворком для их создания. Мы также опишем некоторые ситуации, в которых применение WordPress *не будет* лучшим способом для создания вашего веб-приложения.

Что такое веб-сайт?

Вы знаете, что такое веб-сайт: набор из одной или нескольких веб-страниц, содержащих информацию, доступ к которой осуществляется через веб-браузер.

Что такое приложение?

Нам нравится определение в Википедии (**oreil.ly/2DUKI**): "Прикладное программное обеспечение (сокращенно приложение) — это программное обеспечение, предназначенное для выполнения группы скоординированных функций, задач или действий на благо пользователя".

Что такое веб-приложение?

Веб-приложение — это всего лишь приложение, запускаемое через веб-браузер.

Обратите внимание, что в некоторых веб-приложениях технология браузера скрыта, например, когда вы интегрируете веб-приложение в собственное приложение для Android или iOS, запускаете веб-сайт как приложение в Google Chrome или активизируете приложение с помощью Adobe AIR. Однако внутри этих приложений по-прежнему существует система парсинга HTML, CSS и JavaScript.

Вы также можете думать о веб-приложении как о *веб-сайте с некоторым дополнительным функционалом приложения*. Не существует точной разделительной линии, за которой веб-сайт становится веб-приложением. Это один из тех случаев, когда данное обстоятельство совершенно очевидно для вас.

Мы *можем* лишь объяснить некоторые функции веб-приложения, дать вам несколько примеров, а затем попытаться придумать сокращенное определение, чтобы вы в целом понимали, о чем мы говорим, когда употребляем этот термин в книге.

При чтении этой книги вам встретятся ссылки на SchoolPress. SchoolPress — это веб-приложение, которое мы создаем, чтобы помочь школам и преподавателям управлять своими учениками и учебными программами. Все примеры кода направлены на функциональность, которая может существовать в SchoolPress. Более подробно об общей концепции SchoolPress мы поговорим далее в этой главе.

Функции веб-приложения

Далее перечислены некоторые функции, обычно связанные с веб-приложениями и приложениями в целом. Чем больше этих функций есть на веб-сайте, тем целесообразнее считать его веб-приложением[1].

- *Интерактивные элементы*

  Типичный веб-сайт включает в себя навигацию по страницам, прокрутку и нажатие гиперссылок. Веб-приложения также могут иметь ссылки и прокрутку, но чаще встречаются другие методы навигации по приложению.

  Веб-сайты с формами предлагают функционал совершения транзакций. Примером может служить форма обратной связи на веб-сайте или форма заявки на странице вакансий на веб-сайте компании. Формы позволяют пользователям взаимодействовать с сайтом, используя нечто большее, чем клик.

  Веб-приложения будут иметь еще больше элементов интерактивного пользовательского интерфейса (UI — user interface). Примеры включают в себя панели инструментов, элементы перетаскивания, редакторы форматированного текста и ползунки.

- *Задачи, а не содержание*

  Помните, что веб-приложения "разработаны, чтобы помочь пользователю выполнять определенные задачи". Пользователи Google Maps получают маршруты проезда. Пользователи Gmail пишут электронные письма. Пользователи Trello управляют списками. Пользователи SchoolPress добавляют комментарии по теме урока.

  Некоторые приложения по-прежнему ориентированы на контент. Типичный сеанс Facebook или Twitter почти на 90% состоит из чтения. Тем не менее сами

[1] На многие идеи в этом разделе повлияли сообщения в блоге: "Что такое веб-приложение?" Доминика Хазаэль-Массо (**bit.ly/wiawa**) и Боба Бэксли (**bit.ly/wiawa2**).

приложения представляют способ просмотра контента, отличный от обычного просмотра веб-страниц.

- *Логины*

  Вход в систему и учетные записи позволяют веб-приложению сохранять информацию о своих пользователях. Эта информация предназначена для облегчения основных задач приложения и обеспечения постоянного взаимодействия. При входе в систему пользователи SchoolPress могут видеть, какие сообщения являются непрочитанными. У них также есть имя пользователя, которое идентифицирует их деятельность в приложении.

  Веб-приложения также могут иметь уровни пользователей. В SchoolPress будут администраторы, контролирующие внутреннюю работу приложения, учителя, управляющие процессом обучения и ученики, участвующие в дискуссиях в классе.

- *Возможности устройства*

  Веб-приложения, работающие на вашем телефоне, могут получить доступ к вашей камере, адресной книге, внутреннему хранилищу и информации о местоположении GPS. Веб-приложения, функционирующие на персональном компьютере, могут иметь доступ к веб-камере или локальному жесткому диску. Одно и то же веб-приложение может реагировать по-разному в зависимости от устройства, на котором оно запущено. Веб-приложения будут настраиваться для разных размеров экранов, их разрешений и возможностей.

- *Работа в автономном режиме*

  Всегда, когда это возможно, желательно, чтобы ваши веб-приложения работали в автономном режиме. Конечно, интерактивность Интернета главным образом определяет "веб" часть веб-приложения, но сайт, который все еще работает, когда вы проезжаете через туннель, будет больше походить на приложение.

  С Gmail вы можете создавать черновики писем в автономном режиме. Evernote позволяет создавать заметки офлайн, а затем синхронизировать их с Интернетом после восстановления подключения.

- *Мэшапы*

  Веб-приложения могут связывать одно или несколько веб-приложений вместе. Веб-приложение может использовать различные веб-сервисы и API для передачи и извлечения данных. У вас может быть веб-приложение, которое получает информацию о местоположении, такую как долгота и широта, из Twitter и отправлять и отмечать его на карте Google Map.

Мобильные приложения

С тех пор, как первое издание этой книги было опубликовано еще в 2012 году, веб-приложения, в частности мобильные приложения, получили широкое распространение. На большинстве веб-сайтов мобильные устройства уже преобладают

над ПК и являются крупнейшим источником трафика (Источник: Perficient, Inc. (**oreil.ly/N92kX**)).

В 2012 году типичное веб-приложение выглядело как Basecamp — менеджер проектов, доступ к которому осуществлялся через веб-браузер на вашем компьютере. В 2019 году типичное веб-приложение выглядит как Twitter — приложение для коммуникации, доступ к которому можно получить с помощью телефона iOS или Android.

Поскольку в большинстве случаев многие из пользователей будут получать доступ к вашим веб-сайтам и приложениям на мобильном устройстве, при разработке веб-приложений мы придерживаемся взгляда "прежде всего для мобильных устройств". О том, как заставить ваши приложения WordPress работать на мобильных устройствах, речь пойдет в *главе 16*. А об основах адаптивного дизайна и о том, как заставить ваши веб-сайты правильно отображаться на экранах любого размера мы расскажем в *главе 4*.

Прогрессивные веб-приложения

Прогрессивные веб-приложения (PWA) — это веб-сайты, реализующие преимущества современных функций браузера, которые ведут себя как собственные приложения для Android, iOS или ПК. В частности, веб-сайты, которые используют *сервис-воркеры* (service workers) для работы в автономном режиме, содержат файл манифеста веб-приложения для определения приложения в операционной системе (ОС) и отвечают нескольким другим требованиям, поэтому могут быть запущены как приложения прямо из браузера.

Идея PWA была разработана командой Google Chrome, но теперь поддерживается на iOS и в большинстве современных веб-браузеров. Плагин для поддержки PWA (**oreil.ly/gThAQ**) разрабатывается для работы основных функций PWA в WordPress core. С помощью этого плагина вы можете превратить ваш сайт WordPress в PWA, и это хорошая идея. Но на самом деле создание PWA — это скорее образ мышления, а не простой переход. Аналогично "функциям веб-приложения", которые мы только что описали, главный сайт PWA Google (**oreil.ly/FAwTK**) содержит контрольный список функций, необходимых в большинстве PWA (**oreil.ly/HBuTg**). Укажем следующие базовые функции:

- Сайт обслуживается по HTTPS.
- Страницы адаптированы для планшетов и мобильных устройств.
- Все URL-адреса приложения загружаются в автономном режиме.
- Метаданные предоставляются для добавления на главный экран.
- Первая загрузка быстрая даже с 3G.
- Сайт работает независимо от браузера.
- Переход между страницами не ощущается как работа с сетью.
- У каждой страницы есть URL.

В дополнение к базовым функциям есть контрольный список элементов для "примерных" PWA, который охватывает пользовательский опыт (UX) и производительность. Инструмент Google Lighthouse (**oreil.ly/GwEkb**) предоставляет автоматизированные тесты и отчеты для соответствия критериям PWA. Даже разработчики полностью нативных приложений или приложений для браузера, могут воспользоваться некоторыми советами из контрольных списков PWA и отчетов Lighthouse.

Зачем нужен WordPress?

Ни один язык программирования или программный инструмент не подойдет для любой разработки. Мы еще коснемся вопроса, почему вы *не* захотите использовать WordPress, но сейчас давайте рассмотрим некоторые ситуации, в которых создавать веб-приложения целесообразно именно с помощью WordPress.

Вы уже используете WordPress

Если вы уже используете WordPress для своего основного сайта, то можете просто добавить плагин, который вам необходим. В WordPress есть отличные плагины для электронной торговли (WooCommerce), форумов (bbPress), сайтов с подпиской (Paid Memberships Pro), функций социальных сетей (BuddyPress) и геймификации (BadgeOS).

Встраивание приложения в существующий сайт WordPress сэкономит ваше время и упростит работу для ваших пользователей. Итак, если ваше приложение достаточно простое, вы можете создать собственный плагин на своем сайте WordPress для программирования функциональности вашего веб-приложения.

Когда вы довольны своим сайтом на WordPress, не поддавайтесь искушению, если люди говорят, что вам нужно перейти на что-то другое, чтобы добавить определенные функции на ваш сайт. Это, скорее всего, неправда. Вам не нужно выбрасывать всю работу, которую вы уже проделали в WordPress, и последующие доводы — это веские причины придерживаться WordPress.

С помощью WordPress легко управлять контентом

Разработанный сначала как платформа для ведения блогов, с введением пользовательских типов записей (англ. CPT — Custom Post Type) в версии 3.0 WordPress развился в полностью функциональную систему управления контентом (англ. CMS — Content Management System). Любая страница или сообщение может быть отредактировано администратором через панель инструментов, доступ к которой можно получить через ваш веб-браузер. О работе с CPT вы узнаете в *главе 5*.

WordPress упрощает добавление и редактирование контента с помощью редактора WYSIWYG (What You See Is What You Get — "Что видишь, то и получаешь"), поэтому вам не нужно привлекать веб-дизайнеров каждый раз, когда вы хотите внести простые изменения в свой сайт. Вы также можете создавать собственные меню и элементы навигации для своего сайта, не касаясь программного кода.

Если ваше веб-приложение сфокусировано на фрагментах контента (например, наше приложение SchoolPress ориентировано на назначение пользователям заданий и их обсуждение), API пользовательских типов постов для WordPress (описанный в *главе 5*) позволяет легко настроить этот пользовательский контент и управлять им.

Даже приложения, более ориентированные на задачи, обычно имеют несколько страниц с информацией, документацией и продажами. Использование WordPress для вашего приложения даст вам возможность управления приложением и всем вашим контентом из одного места.

WordPress позволяет просто и безопасно управлять пользователями

В WordPress есть все необходимое для добавления на сайт как администраторов, так и конечных пользователей.

Помимо управления доступом к контенту, система ролей и возможностей в WordPress расширяема и позволяет вам контролировать, какие *действия* доступны для определенных групп пользователей. Например, по умолчанию пользователи с ролью участника могут *добавлять* новые сообщения, но не могут *публиковать* их. Точно так же вы можете создавать новые роли и возможности для управления доступом к вашим пользовательским функциям.

Вы можете использовать плагины, такие как Paid Memberships Pro, чтобы расширить встроенное управление пользователями и назначать членов разных уровней и контролировать доступ пользователей к контенту. Например, вы можете создать уровень, чтобы предоставить участникам с платным аккаунтом доступ к премиум-контенту на вашем сайте WordPress.

Плагины

В репозитории WordPress (**wordpress.org/plugins/**) имеется более 55 тыс. бесплатных плагинов. Существует множество дополнительных плагинов, как бесплатных, так и коммерческих, на различных сайтах в Интернете. Если у вас появляется идея для расширения вашего сайта, велика вероятность, что для этого уже есть плагин, который сэкономит ваше время и деньги.

Существует несколько необходимых плагинов, которые мы задействуем практически на каждом создаваемом нами сайте и веб-приложении.

Для большинства разрабатываемых веб-сайтов вы, наверное, хотите кэшировать вывод для ускорения просмотра, использовать такие инструменты, как Google Analytics для отслеживания посетителей, создавать карты сайта и настраивать набор страниц для поисковой оптимизации (англ. SEO — Search Engine Optimization), а также выполнять ряд других общих задач.

Есть много хорошо поддерживаемых плагинов для всех этих функций. Мы предлагаем наши любимые в этой книге. Вы можете найти их список на веб-сайте этой книги (**bwawwp.com/plugins/**).

Гибкость важна

WordPress — полноценный фреймворк, способный на многое. Кроме того, WordPress построен на основе технологий PHP, JavaScript и MySQL, поэтому все, что вы можете встроить в PHP/MySQL (а это почти все), может быть достаточно легко встроено в ваше приложение WordPress.

WordPress и PHP/MySQL в целом не идеально подходят для любой задачи, но они пригодны для широкого круга задач. Наличие одной платформы, которая будет расти вместе с вашим бизнесом, позволит вам быстрее выполнять задачи и перестраиваться. Например, вот типичный пример сайта запуска стартапа Lean, работающего на WordPress:

- Объявите о своем стартапе с помощью одностраничного сайта.
- Добавьте форму для сбора адресов электронной почты.
- Добавить блог.
- Сосредоточьтесь на SEO и оптимизируйте весь контент.
- Отправляйте посты блога в Twitter и Facebook.
- Добавьте форумы.
- Используйте плагин Paid Memberships Pro, чтобы позволить участникам платить за доступ.
- Добавьте пользовательские формы, инструменты и поведение приложения для участников с платной подпиской.
- Обновите пользовательский интерфейс, используя JavaScript и фреймворки.
- Настройте сайт и сервер для масштабирования.
- Локализуйте сайт/приложение для разных стран и языков.
- Добавьте поддержку Progressive Web App.
- Запустите обертки для iOS и Android для приложения.

Суть, почему нужно идти по этому пути, состоит в том, что на каждом этапе у вас есть *одна и та же база данных пользователей* и *одна и та же платформа разработки*.

Частые обновления безопасности

Тот факт, что WordPress используется на миллионах сайтов, делает его целью хакеров, пытающихся найти уязвимости в его безопасности. Некоторые из этих хакеров были успешны в прошлом, однако разработчики WordPress быстро отслеживают уязвимости и выпускают обновления для их устранения. Получается, будто миллионы людей постоянно тестируют и исправляют ваше программное обеспечение.

Базовая архитектура WordPress делает применение этих обновлений быстрым и безболезненным процессом, который могут выполнять даже начинающие веб-

пользователи. Если вы хорошо знаете, как настроить WordPress и обновить его до последних версий, когда они станут доступны, WordPress станет гораздо более безопасной платформой для вашего сайта, чем все остальное. Мы обсудим безопасность более подробно в *главе 8*.

Стоимость

WordPress — бесплатный продукт. PHP бесплатен. MySQL бесплатен. Большинство плагинов тоже бесплатны.

Серверы и хостинг стоят денег, но в зависимости от того, насколько велико ваше веб-приложение и сколько трафика вы получаете, это может быть относительно недорого. Если вам требуются пользовательские функции, которых нет ни в одном из существующих плагинов, вам, возможно, придется заплатить разработчику за создание нового. Или, если вы сами разработчик, это будет стоить вам времени.

Ответы на некоторые распространенные критические мнения о WordPress

Некоторые высококвалифицированные критики WordPress могут сказать, что это не очень хорошая основа для создания веб-приложений и это вообще не фреймворк. При всем уважении к тем, кто придерживается подобных мнений, нужно объяснить, почему мы не согласны. Далее приведены некоторые распространенные критические замечания.

WordPress подходит только для блогов

Многие люди считают, что поскольку WordPress впервые был создан для ведения блога, он хорош только для этой цели.

Подобные заявления были верны несколько лет назад, но с тех пор WordPress внедрил мощную функциональность CMS, что делает его полезным для других сайтов, ориентированных на контент. WordPress в настоящее время является самой популярной из используемых CMS с долей рынка более 60%[2]. На рис. 1.1 показан слайд из презентации Мэтта Мулленвега "Состояние WordPress" с конференции WordCamp San Francisco 2013. Перевернутая пирамида слева изображает WordPress 2006 года, где бóльшая часть кода посвящена приложению блога, и есть немного кода CMS и платформы, на котором она держится. Правая пирамида представляет текущее состояние платформы WordPress, где бóльшая часть кода находится в самой платформе, поверх которой расположен слой CMS, а поверх уровня CMS — приложение для блога. Сейчас WordPress является гораздо более устойчивой платформой, чем несколько лет назад.

[2] W3Tech (**bit.ly/w3techs**) регулярно проводит исследования по использованию различных систем управления контентом.

Рис. 1.1. Диаграммы из презентации Мэтта Мулленвега "Состояние WordPress" 2013 год. WordPress не всегда был таким устойчивым

С помощью API пользовательских типов записей можно настроить установку WordPress для поддержки других типов контента, кроме постов или страниц блога. Мы подробно рассмотрим это в *главе 5*.

WordPress предназначен только сайтов, ориентированных на контент

Подобно людям с позицией "только для блогов", некоторые скажут, что WordPress предназначен только для управления контентом сайтов.

Во-первых, даже если бы WordPress был применим только к контентным сайтам и приложениям, на него приходилось бы большое количество приложений. Главный экран вашего телефона, вероятно, содержит множество приложений, основанных на контенте, таких как Netflix, Twitter, Facebook, Reddit и Evernote. Это очень популярные приложения, поддерживаемые гигантскими компаниями. Сейчас мы не говорим, что эти приложения работают на WordPress, но мы предполагаем, что *можно* создать приложение, похожее на это, с использованием WordPress в качестве фреймворка.

Во-вторых, как мы подробно рассмотрим в этой книге, WordPress — это отличная платформа для создания более интерактивных веб-приложений. Основной функцией, позволяющей выбрать WordPress в качестве основы, является API плагинов, который позволяет вам понять, как работает WordPress по умолчанию, и что-то изменить. Вам доступны не только тысячи плагинов в репозитории WordPress и других местах в Интернете, API плагинов позволяет вам написать собственные пользовательские плагины, чтобы WordPress делал что угодно с помощью PHP/MySQL.

WordPress не масштабируется

Некоторые из людей, которые так говорят, будут указывать на установку WordPress по умолчанию, работающую на хостинге нижнего уровня, и они отмечают, как сайт замедляется или "падает" при большой нагрузке, и, таким образом, приходят к выводу, что WordPress не масштабируется.

Или, может быть, когда мы предложили вам создать сайт, такой как Facebook, с использованием WordPress, вы справедливо насмехались над этой идеей.

Если вы намереваетесь создать приложение в масштабе Facebook, эта книга не для вас. Спросите своего технического директора, какая часть их бюджета в миллиард долларов выделена вашему приложению и каких инженеров вам нужно переманить из Google и Amazon, чтобы создать собственное решение.

В действительности многие сайты с большим трафиком работают на WordPress. WordPress.com работает на том же базовом программном обеспечении, что и любой сайт WordPress, и является одним из самых популярных сайтов в мире.

По мере расширения вашего приложения вам необходимо будет обновлять и заменять отдельные компоненты, чтобы соответствовать новому масштабу. Проблемы с масштабированием WordPress такие же, как и при масштабировании любого приложения: кэширование страниц и данных, более быстрая обработка вызовов базы данных и повышение производительности сети. Крупные сайты, такие как WordPress.com, TechCrunch и блог *New York Times*, стали использовать WordPress. Точно так же большинство уроков по масштабированию приложений PHP/MySQL в целом применимы и к WordPress. Мы подробно расскажем о масштабировании приложений WordPress в *главе 14*.

WordPress небезопасен

Как и для любого продукта с открытым исходным кодом, при использовании WordPress есть компромисс в отношении безопасности.

С одной стороны, поскольку WordPress очень популярен, он часто становится целью хакеров, ищущих уязвимости безопасности. А поскольку исходный код открытый, уязвимости будет легче обнаружить.

С другой стороны, поскольку WordPress является продуктом с открытым исходным кодом, вы узнаете, если эксплойты станут общедоступными, и кто-то другой, вероятно, выпустит исправление для вас.

Мы чувствуем себя более уверенно, зная, что есть множество людей, пытающихся использовать WordPress, и столько же людей, которые работают над тем, чтобы защитить WordPress от взломов. Мы не верим в "безопасность через неизвестность", если только в качестве дополнительной меры. Мы бы предпочли, чтобы дыры в безопасности нашего программного обеспечения появлялись открыто, а не оставались незамеченными до самого худшего момента.

В *главе 8* более детально рассматриваются вопросы безопасности, в том числе дан список рекомендаций по повышению безопасности установки WordPress и описаны способы безопасной разработки.

Плагины WordPress ужасны

API плагинов в WordPress и тысячи плагинов, которые были разработаны с его использованием, являются "секретным соусом" и, по нашему мнению, основной причиной того, что WordPress стал настолько популярным и настолько успешным в качестве веб-платформы.

Некоторые люди скажут: "Конечно, есть тысячи плагинов, но они все ужасны". Ну да, некоторые из плагинов *действительно* не очень полезны.

Но есть много плагинов, которые определенно не являются чепухой, например AppPresser, разработанный одним из авторов этой книги Брайаном Мессенленером. Если вам нравится WordPress для управления своим письменным контентом или

интернет-магазином, то плагин и платформа AppPresser — вот самый быстрый способ получить данный контент или сохранить в мобильном приложении.

Плагин Paid Memberships Pro, разработанный другим автором Джейсоном Коулманом, тоже хорош. Использование Paid Memberships Pro для управления биллингом и участниками позволит вам сосредоточить усилия по разработке на основной функциональности вашего приложения, а не на том, как интегрировать ваш сайт с платежной системой.

Многие плагины делают что-то очень простое (например, скрывают панель администратора от обычных пользователей), работают именно так, как рекламируется, и на самом деле вовсе не ужасны.

Темы и плагины, найденные в репозитории WordPress.org, тщательно проверяются сообществом на предмет безопасности и качества кода. Общеизвестно, что обзор на темы WordPress.org (**oreil.ly/WgTyD**) более строгий и всеобъемлющий, чем в других платформах. Проект Tide (**oreil.ly/iR94e***)* работает над добавлением автоматических тестов для плагинов и репозиториев тем, что приведет к увеличению качества плагинов и обновлений, а также позволит обнаружить проблемы совместимости и безопасности быстрее.

Даже дрянные плагины можно исправить, переписать или заимствовать для улучшения их работы. Иногда проще переписать плохой плагин, чем исправлять его. Однако в данном случае вы окажетесь все же на шаг ближе к цели, чем при написании с нуля.

Никто не заставляет вас использовать плагины WordPress, не проверяя их самостоятельно. Если вы создаете серьезное веб-приложение, то должны самостоятельно проверить код плагина, исправить его в соответствии со своими стандартами и продолжить разработку.

Когда не следует использовать WordPress

WordPress не является универсальным решением для любого приложения. В этом разделе описываются несколько случаев, в которых вы *не хотели бы* использовать WordPress для создания своего приложения.

Вы планируете лицензировать или продавать технологию своего сайта

WordPress использует Общедоступную лицензию GNU версии 2 (GPLv2), которая содержит ограничения на то, как вы распространяете программное обеспечение, созданное с его помощью. А именно, вы не можете ограничивать то, что люди делают с вашим программным обеспечением, когда вы продаете или распространяете его.

Это сложная тема, но основная идея заключается в том, что если вы продаете только свое приложение или предоставляете *доступ* к нему, то вам не нужно беспоко-

иться о GPLv2. Однако если вы продаете или распространяете исходный код вашего приложения, то GPLv2 будет применяться к распространяемому вами коду.

Например, если мы размещаем SchoolPress на наших собственных серверах и продаем учетные записи для доступа к приложению, это не считается распространением, а GPLv2 никак не влияет на наш бизнес.

Но если мы захотим разрешить школам устанавливать программное обеспечение для запуска на своих собственных серверах, то нам необходимо предоставить им исходный код. Это будет считаться актом распространения. Наши клиенты смогут на законных основаниях бесплатно раздавать наш исходный код, даже если мы изначально взяли с них плату за программное обеспечение. Мы должны учитывать лицензию GPLv2, которая не позволяет нам ограничивать то, что пользователи делают с кодом после его загрузки.

Имеется другая платформа, которая приведет вас к цели быстрее

Если у вас есть команда опытных разработчиков Ruby, то целесообразно создавать веб-приложение именно на Ruby. Если есть платформа, фреймворк или пакет, включающий 80% функций Ruby, необходимых для вашего веб-приложения, и WordPress не имеет ничего подобного, то вам, вероятно, следует предпочесть эту другую платформу.

Гибкость не важна для вас

Одна из главных особенностей сайта WordPress — возможность быстрой замены частей сайта в соответствии с вашими потребностями. Например, если "лайки" Facebook перестали приносить трафик, вы можете удалить плагин Facebook Connect и установить плагин для Pinterest.

Как правило, обновление вашей темы или замена плагинов на сайте WordPress будет быстрее, чем разработка функций с нуля на другой платформе. Однако в тех случаях, когда оптимизация и производительность важнее, чем возможность быстрого обновления приложения, лучшим выбором будет программирование нативного приложения или программирование прямо на PHP.

Если ваше приложение служит для выполнения *одной простой задачи*, то вы, скорее всего, захотите построить его на более низком уровне. Например, сервер лицензий Paid Memberships Pro представляет собой один файл JSON с дополнительной информацией и небольшим скриптом для проверки лицензионных ключей и доставки сжатых файлов. Джейсон создал этот сервер лицензий на PHP с большим объемом кэширования. Сервер лицензий функционирует на DigitalOcean Sroplet за 10 долларов в месяц и обслуживает более 80 тыс. сайтов, работающих под управлением Paid Memberships Pro.

Точно так же, если у вас есть ресурсы масштаба Facebook, то вы можете позволить себе создавать все вручную и использовать собственные компиляторы PHP-to-C и

нативные компоненты iOS, чтобы сократить время загрузки вашего веб-сайта и приложения на несколько миллисекунд.

Ваше приложение должно работать в режиме реального времени

Одним из потенциальных недостатков WordPress, о котором мы поговорим позже, является его зависимость от типичной архитектуры веб-сервера. В стандартной настройке WordPress пользователь посещает URL-адрес, который связывается с веб-сервером (например, Apache) по HTTP, запускает скрипт PHP для генерации страницы, а затем возвращает пользователю полную страницу.

Существуют способы улучшить производительность этой архитектуры с использованием методов кэширования и/или оптимизированных настроек сервера. Вы можете сделать WordPress асинхронным с помощью вызовов Ajax или доступа к базе данных с помощью альтернативных клиентов. Однако если ваше приложение должно работать в режиме реального времени и быть полностью асинхронным (например, приложение, похожее на чат, или многопользовательская игра), мы советуем хорошенько подумать, подойдет ли вам WordPress.

Многие разработчики WordPress, включая Мэтта Мулленвега, основателя и духовного лидера WordPress, понимают это ограничение. Все больше функциональности WordPress переносится в JavaScript, где вычисления могут быть перенесены в браузер, а фреймворки, такие как REACT, позволяют создавать высокоинтерактивные события. Новый редактор Gutenberg, добавленный в WordPress 5.0, является лучшим примером этого шага и свидетельствует о грядущих событиях. Но в данный момент вы столкнетесь с трудностями, пытаясь заставить WordPress работать асинхронно с той же производительностью, что и нативное приложение или что-то полностью построенное на Node.js либо других технологиях, специально предназначенных для приложений реального времени.

WordPress как фреймворк

Системы управления контентом, такие как WordPress, Drupal и Joomla, часто остаются вне обсуждения фреймворков, но на самом деле WordPress (в частности) действительно отлично подходит для того, для чего предназначены фреймворки: для быстрого создания приложений.

В течение нескольких минут вы можете настроить WordPress и получить полнофункциональное приложение с регистрацией пользователей, управлением сеансами, управлением контентом и панелью для мониторинга активности сайта.

Различные API, общие объекты и вспомогательные функции, описанные в этой книге, позволяют быстрее программировать сложные приложения, не беспокоясь об интеграции систем более низкого уровня.

На рис. 1.2 показан правый треугольник из презентации Мэтта Мулленвега "Состояние WordPress" 2013 года, изображающий устойчивую платформу WordPress со