

---

*Лорел посвящается*



# ОГЛАВЛЕНИЕ

<b>Об авторе .....</b>	<b>13</b>
<b>Предисловие .....</b>	<b>14</b>
Что нужно знать об этой книге .....	14
Структура книги .....	15
Соглашения, принятые в этой книге .....	16
Использование примеров кода .....	16
Как связаться с нами .....	17
Благодарности .....	17
<b>ЧАСТЬ I.</b>	
<b>Особенности языка .....</b>	<b>19</b>
<b>Глава 1. Новый PHP .....</b>	<b>20</b>
Прошлое .....	20
Настоящее .....	21
Будущее .....	23
<b>Глава 2. Особенности .....</b>	<b>25</b>
Пространства имен .....	25
Зачем нужны пространства имен .....	28
Объявление .....	28
Импорт и псевдонимы .....	30
Полезные советы .....	32
Интерфейсы .....	34
Трейты .....	39
Зачем нужны трейты .....	39
Как создать трейт .....	41
Как использовать трейт .....	43
Генераторы .....	44
Создание генератора .....	45
Использование генератора .....	46
Замыкания .....	48
Создание .....	49
Прикрепление состояния .....	50
Расширение Zend OPcache .....	53

Включение расширения Zend OPcache .....	54
Настройка расширения Zend OPcache .....	55
Использование расширения Zend OPcache.....	56
Встроенный HTTP-сервер .....	56
Запуск сервера .....	57
Настройка сервера .....	57
Сценарии маршрутизации .....	58
Обнаружение встроенного сервера .....	58
Недостатки .....	59
Что дальше.....	59
<b>ЧАСТЬ II.</b>	
<b>Передовые технологии .....</b>	<b>61</b>
<b>Глава 3. Стандарты .....</b>	<b>62</b>
PHP-FIG приходит на помощь .....	63
Совместимость фреймворков .....	64
Интерфейсы .....	64
Автозагрузка.....	64
Стиль.....	65
Что такое PSR? .....	65
PSR-1: Базовый стиль оформления кода .....	66
PSR-2: Строгий стиль оформления кода.....	68
PSR-3: Интерфейс журналирования .....	73
Создание компонента журналирования PSR-3 .....	73
Использование компонента журналирования PSR-3 .....	74
PSR-4: Автозагрузка.....	75
Почему автозагрузка так важна .....	76
Модель автозагрузки PSR-4.....	77
Как написать автозагрузчик PSR-4 (и почему этого делать не нужно) ....	78
<b>Глава 4. Компоненты .....</b>	<b>80</b>
Почему надо использовать компоненты?.....	80
Что представляют собой компоненты? .....	81
Компоненты и фреймворки .....	83
Не все фреймворки плохи.....	84
Использование инструмента, соответствующего задаче .....	85
Поиск компонентов .....	85
Магазин .....	86
Выбор .....	87
Оставьте отзыв .....	88
Использование PHP-компонентов.....	88
Установка Composer .....	89

Как использовать Composer .....	90
Пример проекта .....	92
Composer и закрытые хранилища .....	96
Создание PHP-компонентов .....	98
Имена производителя и пакета .....	98
Пространства имен .....	99
Организация файловой системы .....	99
Файл composer.json .....	100
Файл README .....	103
Реализация компонента .....	103
Управление версиями .....	106
Размещение на сайте Packagist .....	106
Использование компонента .....	107
<b>Глава 5. Передовой опыт .....</b>	<b>109</b>
Санирование и проверка ввода, и экранирование вывода .....	110
Санирование ввода .....	110
Проверка данных .....	114
Экранирование вывода .....	115
Пароли .....	116
Не храните пароли в открытом виде .....	116
Не ограничивайте пароли ваших пользователей .....	116
Не отправляйте пароли пользователей по электронной почте .....	117
Хеширование паролей пользователей с помощью bcrypt .....	117
Программный интерфейс хеширования паролей .....	119
Программный интерфейс хеширования паролей для PHP < 5.5.0 .....	124
Даты, время и часовые пояса .....	125
Установка часового пояса по умолчанию .....	125
Класс DateTime .....	125
Класс DateInterval .....	127
Класс DateTimeZone .....	128
Класс DatePeriod .....	129
Компонент nesbot/carbon .....	130
Базы данных .....	131
Расширение PDO .....	131
Подключение базы данных и DSN .....	131
Параметризованные запросы .....	134
Результаты запроса .....	137
Транзакции .....	139
Многобайтовые строки .....	143
Кодировка символов .....	144
Отображение данных в кодировке UTF-8 .....	145
Потоки данных .....	145
Обертки потоков .....	146
Контекст потока .....	150
Фильтры потоков .....	150

Пользовательские фильтры потоков .....	153
Ошибки и исключения .....	156
Исключения .....	157
Обработчики исключений .....	161
Ошибки .....	162
Обработчики ошибок .....	164
Ошибки и исключения в ходе разработки .....	166
Эксплуатация .....	168

## **ЧАСТЬ III.**

### **Развертывание, тестирование и настройка ..... 171**

#### **Глава 6. Хостинг ..... 172**

Разделяемые серверы .....	172
Виртуальный выделенный сервер .....	173
Выделенный сервер .....	174
PaaS .....	175
Выбор тарифного плана хостинга .....	176

#### **Глава 7. Комплектование ..... 177**

Наша цель .....	178
Настройка сервера .....	178
Первый вход .....	178
Обновление программного обеспечения .....	179
Непривилегированный пользователь .....	180
SSH-аутентификация с помощью парных ключей .....	181
Отключение парольной аутентификации и запрет входа пользователя root .....	183
PHP-FPM .....	184
Установка .....	184
Глобальная конфигурация .....	185
Настройка пулов .....	186
nginx .....	189
Установка .....	190
Виртуальный хост .....	190
Автоматизация комплектования .....	193
Делегирование комплектования .....	194
Дополнительные материалы .....	194
Что дальше .....	195

#### **Глава 8. Настройка ..... 196**

Файл php.ini .....	196
Память .....	197

Zend OPcache .....	198
Выгрузка файлов .....	201
Максимальное время выполнения .....	202
Обслуживание сеансов .....	203
Буферизация вывода .....	204
Кэш Realpath .....	204
Что дальше .....	205
<b>Глава 9. Развертывание .....</b>	<b>206</b>
Управление версиями .....	206
Автоматизация развертывания .....	207
Сделайте развертывание простым .....	207
Сделайте развертывание предсказуемым .....	207
Сделайте развертывание обратимым .....	207
Capistrano .....	207
Как это работает .....	208
Установка .....	208
Настройка .....	209
Аутентификация .....	211
Подготовка удаленного сервера .....	211
Обработчики Capistrano .....	212
Развертывание приложения .....	213
Откат к предыдущей версии приложения .....	213
Дополнительные материалы .....	213
Что дальше .....	213
<b>Глава 10. Тестирование .....</b>	<b>214</b>
Почему мы тестируем? .....	214
Когда мы тестируем? .....	215
Перед .....	215
В процессе .....	215
После .....	216
Что мы тестируем? .....	216
Как мы тестируем? .....	216
Модульное тестирование .....	216
Разработка через тестирование (TDD) .....	217
Разработка, основанная на функционировании (BDD) .....	217
PHPUnit .....	219
Структура каталогов .....	219
Установка PHPUnit .....	220
Установка Xdebug .....	221
Настройка PHPUnit .....	222
Класс Whovian .....	223
Класс теста WhovianTest .....	224

Запуск тестов .....	227
Охват кода .....	228
Непрерывное тестирование с помощью Travis CI .....	229
Установка .....	229
Запуск .....	230
Дополнительные материалы .....	231
Что дальше.....	231
<b>Глава 11. Профилирование .....</b>	<b>232</b>
Когда следует использовать профилировщик.....	232
Типы профилировщиков.....	233
Xdebug .....	233
Настройка.....	234
Включение .....	235
Анализ .....	235
XHProf .....	235
Установка .....	236
XHGUI.....	236
Настройка.....	237
Включение .....	237
Профилировщик New Relic .....	238
Профилировщик Blackfire.....	238
Дополнительные материалы .....	238
Что дальше.....	239
<b>Глава 12. HHVM и Hack .....</b>	<b>240</b>
HHVM .....	240
PHP в Facebook .....	241
Совместимость HHVM с Zend Engine .....	243
Будет ли HHVM правильным выбором для меня? .....	243
Установка .....	244
Настройка.....	245
Расширения.....	246
Мониторинг HHVM с помощью Supervisorд.....	246
HHVM, FastCGI и Nginx .....	248
Язык Hack.....	250
Перевод с PHP на Hack .....	250
Что такое типы? .....	251
Статическая типизация.....	252
Динамическая типизация.....	253
Двойной подход языка Hack.....	254
Контроль типов в Hack .....	254
Режимы Hack .....	255
Синтаксис Hack .....	256

Структуры данных Nаск.....	258
ННVM и Nаск против PHP .....	259
Дополнительные материалы .....	261
<b>Глава 13. Сообщество .....</b>	<b>262</b>
Местная группа PHP-разработчиков .....	262
Конференции .....	262
Наставничество.....	263
Будьте в курсе .....	263
Сайты .....	263
Списки рассылок .....	263
Твиттер .....	263
Подкасты .....	263
Юмор.....	264
<b>Приложение А. Установка PHP .....</b>	<b>265</b>
Linux.....	265
Менеджеры пакетов.....	265
Ubuntu 14.04 LTS .....	266
CentOS 7 .....	268
MAMP .....	270
Homebrew .....	273
Сборка из исходных текстов .....	277
Получение исходного кода.....	278
Windows.....	284
Скомпилированные файлы .....	285
WAMP.....	285
Zend Server .....	286
<b>Приложение Б. Локальная среда разработки.....</b>	<b>287</b>
VirtualBox.....	288
Vagrant .....	289
Команды .....	289
Боксы .....	290
Инициализация.....	290
Комплектование .....	291
Синхронизация каталогов .....	292
Быстрый старт .....	293
<b>Предметный указатель .....</b>	<b>295</b>
<b>Об обложке.....</b>	<b>303</b>





## ОБ АВТОРЕ

**Джош Локхарт** (Josh Lockhart) – создатель Slim Framework (<http://slimframework.com/>), популярного микрофреймворка для PHP, обеспечивающего быструю разработку веб-приложений и программных интерфейсов. Джош также основал и до сих пор курирует «PHP The Right Way» (<http://www.phptherightway.com/>) – популярную в PHP-сообществе инициативу, поощряющую современные методики разработки и распространяющую качественную информацию для PHP-разработчиков по всему миру.

Джош работает программистом в New Media Campaigns (<http://www.newmediacampaigns.com/>), агентстве по предоставлению полного комплекса услуг веб-дизайн, разработки и маркетинга в Каррборо, Северная Каролина. Получает удовольствие от создания приложений с использованием HTML, CSS, PHP, JavaScript, Bash и различных фреймворков управления контентом.

Окончил курс Information and Library Science (<http://sils.unc.edu/>) в Университете Северной Каролины, в Чапел-Хилл, в 2008. В настоящее время проживает в Чапел-Хилл, штат Северная Каролина, вместе со своей замечательной женой Лорел и двумя их собаками.

Джоша можно найти в Твиттере (<https://twitter.com/codeguy>). Его блог доступен по адресу <https://joshlockhart.com>, а его проекты с открытым исходным кодом – на GitHub (<https://github.com/codeguy>).



# ПРЕДИСЛОВИЕ

В Интернете можно найти миллионы электронных пособий по PHP. Большинство из них устарели и содержат описание отработавших свое технологий. Но Google продолжает выдавать ссылки на эти пособия, обеспечивая им бессмертие. Устаревшая информация не приносит пользы начинающим PHP-программистам, которые, основываясь на ней, создают медленные и ненадежные PHP-приложения. Я осознал эту проблему в 2013 году, и это явилось основной причиной появления инициативы «PHP The Right Way» (<http://www.phptherightway.com/>) по обеспечению доступа к качественной актуальной информации от авторитетных членов PHP-сообщества.

Книга «Современный PHP» служит той же цели. Эта книга не справочное руководство. Нет. Эта книга представляет собой дружеский и живой разговор между нами. Я познакомлю вас с современным языком программирования PHP. Расскажу о новейших PHP-технологиях, которыми ежедневно пользуюсь в работе и в своих проектах с открытым исходным кодом. И помогу вам начать использовать новейшие стандарты программирования, освоив которые, вы сможете распространять свои PHP-компоненты и библиотеки среди членов PHP-сообщества.

Я часто буду упоминать слово «сообщество», повторяя его снова и снова. PHP-сообщество дружелюбно, приветливо и всегда готово прийти на помощь, хотя, иногда встречаются досадные исключения. Если вам захочется узнать больше об определенной особенности, упомянутой в этой книге, обратитесь в местную группу пользователей PHP с возникшими вопросами. Я уверен, что вокруг вас найдутся PHP-разработчики, которые захотят помочь вам лучше узнать PHP. Ваша местная группа PHP-пользователей это бесценный источник, который позволит вам совершенствоваться в PHP и после завершения чтения этой книги.

## **Что нужно знать об этой книге**

Прежде чем начать, я хочу остановиться на том, что вы найдете в этой книге. Во-первых, я не смогу описать *все* подходы, используемые

в РНР. На это просто не хватит времени. Вместо этого, я расскажу, *как я пользуюсь РНР*. Да, это мой личный взгляд на вещи, но я использую те же методы и стандарты, что и многие другие РНР-разработчики. Все, что вы вынесете из нашей краткой беседы, вы сможете сразу же применить в своих проектах.

Во-вторых, я предполагаю, что вы знакомы с переменными, условными операторами, циклами и так далее. От вас не требуется знание языка РНР, но вы должны, по крайней мере, иметь базовое представление об этих фундаментальных понятиях программирования. Я не буду против, если вы захватите с собой кофе (я очень люблю кофе). А я принесу с собой все, что там полагается к кофе.

И в-третьих, я не настаиваю на использовании какой-либо конкретной операционной системы. Однако примеры кода написаны мной для Linux. Bash-команды, которые вы увидите в книге, я использую в Ubuntu и CentOS, но они также будут работать в OS X. Если вы пользуетесь Windows, настоятельно рекомендую установить и настроить виртуальную машину с Linux для опробования примеров кода, прилагаемых к этой книге.

## **Структура книги**

**Часть I** посвящена описанию новых возможностей языка РНР, таких как пространства имен, генераторы и трейты (traits). Она познакомит вас с современным языком РНР и его особенностями, с которыми вы, возможно, до сих пор не сталкивались.

**Часть II** рассматривает передовые технологии, которые обязательно следует применять в своих РНР-приложениях. Вы слышали о *PSR*, но не совсем понимаете, что это такое и как им пользоваться? Вы хотите узнать, как безопасно обработать пользовательский ввод и выполнять защищенные запросы к базе данных? Тогда вам стоит ее прочесть.

**Часть III** содержит больше технических подробностей, чем первые две части. Она посвящена развертыванию, настройке, тестированию и профилированию РНР-приложений. Мы углубимся в методику развертывания с помощью Capistrano. Поговорим об инструментах тестирования, таких как РНРUnit и Travis CI. Обсудим настройку РНР и ее влияние на работу вашего приложения.

**Приложение А** содержит пошаговые инструкции по установке и настройке РНР-FPM.

**Приложение В** описывает процедуру создания локальной среды разработки максимально приближенной к среде действующего сервера. Мы познакомимся с Vagrant, Puppet, Chef и альтернативными им инструментами для быстрого начала работы.

## Соглашения, принятые в этой книге

В этой книге приняты следующие типографские соглашения:

### *Курсив*

Используется для обозначения новых терминов, адресов электронной почты, имен файлов и расширений имен файлов .

### Моноширинный

Применяется для оформления листингов программ и программных элементов внутри обычного текста, таких как имена переменных и функций, типов данных, переменных окружения, инструкций и ключевых слов .

### Моноширинный жирный

Обозначает команды или другой текст, который должен вводиться пользователем.

### Моноширинный курсив

Обозначает текст, который должен замещаться фактическими значениями, вводимыми пользователем или определяемыми из контекста .



---

---

Так обозначается совет или рекомендация.



---

---

Так будут выделены общие примечания.



---

---

Так будут выделены предупреждения и предостережения.

---

---

## Использование примеров кода

Сопроводительные материалы (примеры кода, упражнения и т. д.) можно загрузить на странице <https://github.com/codeguy/modern-php>.

Данная книга призвана оказать вам помощь в решении ваших задач. Вы можете свободно использовать примеры программного кода

из этой книги в своих приложениях и в документации. Вам не нужно обращаться в издательство за разрешением, если вы не собираетесь воспроизводить существенные части программного кода. Например, если вы разрабатываете программу и используете в ней несколько отрывков программного кода из книги, вам не нужно обращаться за разрешением. Однако в случае продажи или распространения компакт-дисков с примерами из этой книги вам необходимо получить разрешение от издательства O'Reilly. Если вы отвечаете на вопросы, цитируя данную книгу или примеры из нее, получение разрешения не требуется. Но при включении существенных объемов программного кода примеров из этой книги в вашу документацию необходимо получить разрешение издательства.

Мы приветствуем, но не требуем добавлять ссылку на первоисточник при цитировании. Под ссылкой на первоисточник мы подразумеваем указание авторов, издательства и ISBN. Например: «*Modern PHP* by Josh Lockhart (O'Reilly). Copyright 2015 Josh Lockhart, 978-1-491-90501-2.».

За получением разрешения на использование значительных объемов программного кода примеров из этой книги обращайтесь по адресу [permissions@oreilly.com](mailto:permissions@oreilly.com).

## **Как связаться с нами**

С вопросами и предложениями, касающимися этой книги, обращайтесь в издательство:

O'Reilly Media, Inc.  
1005 Gravenstein Highway North Sebastopol, CA 95472  
800-998-9938 (США или Канада)

Список опечаток, файлы с примерами и другую дополнительную информацию вы найдете на странице книги [http://bit.ly/modern\\_php](http://bit.ly/modern_php).

Свои пожелания и вопросы технического характера отправляйте по адресу [bookquestions@oreilly.com](mailto:bookquestions@oreilly.com).

Мы в Facebook: <http://facebook.com/oreilly>

Мы в Twitter: <http://twitter.com/oreillymedia>

Мы в YouTube: <http://www.youtube.com/oreillymedia>

## **Благодарности**

Это моя первая книга. Когда О'Рейли (O'Reilly) предложил мне написать книгу «Современный PHP», это сильно взволновало меня и перепугало до смерти. Первое, что я сделал, исполнил танец Уолтера Хьюстона (Walter Huston). О'Рейли предложил *мне* написать книгу.

Это круто!? И тогда я спросил себя, а *смогу ли я написать так много страниц?* Написание книги долгий и трудоемкий процесс.

Конечно же, я сразу ответил «да». Я знал, что смогу написать книгу «Современный РНР», потому что у меня есть семья, друзья, сослуживцы, редакторы, рецензенты, которые и помогли мне во всем. Я хочу выразить признательность и поблагодарить всех своих сторонников за неоценимую поддержку. Без них эта книга никогда бы не была написана.

Прежде всего, я хочу поблагодарить моего редактора Эллисон Макдональд (Allyson MacDonald, @allyatoreilly) из O'Reilly Media. Элли мила, требовательна, благосклонна и умна. Она точно знает, как и когда мягко подтолкнуть меня в правильном направлении, когда я терял нить повествования. Я не могу себе представить лучшего редактора.

Я также хочу поблагодарить моих технических рецензентов Адама Феирхолма (Adam Fairholm, @adamfairholm) и Эда Финклера (Ed Finkler, @funkatron). Адам – блестящий веб-разработчик из Newfangled (<https://www.newfangled.com/>) и, пожалуй, самой известной его работой является IMVDb (<http://imvdb.com/>) – популярная база данных музыкальных клипов. Эд хорошо известен в РНР-сообществе за его невероятные навыки РНР-разработчика, его персональный подкаст /dev/hell (<http://devhell.info/>) и его заслуживающую всяческих похвал компанию Open Sourcing Mental Illness (<http://funkatron.com/osmi>). Адам и Эд указали мне на все неясные, нелогичные и ошибочные моменты в моих черновиках. Эта книга стала намного лучше благодаря их замечаниям. Я навсегда в долгу перед ними за их советы и здравый смысл. Если ошибки или неточности все же просочились в окончательный вариант рукописи, это, безусловно, моя оплошность.

Мои коллеги из New Media Campaigns (<http://www.newmediacampaigns.com/>) были для меня постоянным источником вдохновения. Джоэл, Клей, Крис, Алекс, Патрик, Эшли, Ленни, Клэр, Тодд, Паскаль, Генри и Натан – снимаю шляпу перед вами за добрые ободряющие слова.

И, самое главное, я хочу поблагодарить мою семью: Лорел, Итан, Тесса, Чарли, Лайза, Гленн и Лиз. Спасибо вам за поддержку, без которой я бы никогда не закончил эту книгу. Моей любимой жене Лорел отдельное спасибо за терпение. Спасибо тебе за кофе Caribou, когда я засиживался за рукописью допоздна. Спасибо за понимание, когда я писал в выходные дни. Спасибо за сохранение моего творческого настроения. Я люблю тебя сейчас и навсегда.



**ЧАСТЬ I**

---

**Особенности  
языка**



# ГЛАВА 1.

## Новый PHP

Язык PHP переживает ренессанс. PHP трансформируется в современный язык сценариев с такими полезными особенностями, как пространства имен, трейты (traits), замыкания и встроенное кэширование байт-кода. Экосистема современного языка PHP также развивается. PHP-разработчики все реже используют монолитные фреймворки и все чаще небольшие специализированные компоненты. Менеджер зависимостей Composer внес революционные изменения в построение PHP-приложений. Он вывел нас из обнесенного неприступными стенами рассадника фреймворков и позволил смешивать и сочетать взаимодействующие между собой PHP-компоненты, лучше всего соответствующие нуждам PHP-приложений. Совместимость компонентов была бы невозможной без предложенных сообществом стандартов и их курирования со стороны PHP Framework Interop Group.

Книга «Современный PHP» является вашим путеводителем по новому PHP, она расскажет, как создавать и развертывать поразительные PHP-приложения, используя стандарты сообщества, лучшие методики и совместимые компоненты.

## Прошлое

Прежде, чем приступать к знакомству с современным языком PHP, важно больше узнать о его происхождении. PHP – язык сценариев, интерпретируемый на стороне сервера. Это значит, что вы должны написать PHP-код, выгрузить его на веб-сервер и запустить с помощью интерпретатора. Язык PHP, как правило, используется в паре с веб-серверами Apache или nginx для поддержки динамического контента. Кроме того, язык PHP можно использовать для создания мощных приложений, запускаемых из командной строки (по аналогии с программами на bash, Ruby, Python и т. д.). Многие PHP-разработчики



не знают об этом и упускают из виду эту действительно интересную особенность. Но вы не должны входить в их число.

Вы можете прочитать официальную историю PHP на странице <http://php.net/manual/history.php>. Я не буду повторять, что так хорошо сказано создателем PHP Расмусом Лердорфом (Rasmus Lerdorf). Я просто скажу, что PHP имеет противоречивое прошлое. Язык PHP берет начало с коллекции CGI-сценариев, написанных Расмусом Лердорфом для отслеживания посещений его электронного резюме. Лердорф назвал свой набор CGI-сценариев «Инструментами личной домашней страницы» («Personal Home Page Tools»). Это раннее воплощение полностью отличается от сегодняшнего языка PHP, который мы знаем. Изначально инструменты PHP Лердорфа не были языком сценариев; они были инструментами поддержки элементарных переменных и автоматической интерпретации форм, основанных на встроенном синтаксисе HTML.

Между 1994 и 1998 годами, PHP претерпел множество изменений и даже несколько раз был переписан с нуля. Энди Гутманс (Andi Gutmans) и Зеев Сураски (Zeev Suraski), два программиста из Тель-Авива, объединили свои усилия с Расмусом Лердорфом, чтобы превратить PHP из небольшого набора CGI-инструментов в полноценный язык программирования с последовательным синтаксисом и базовой поддержкой объектно-ориентированного программирования. Они назвали свой конечный продукт PHP 3 и выпустили его в конце 1998 года. Новое название «PHP» не совпадало с предыдущим и представляло собой акроним от Hypertext Preprocessor (гипертекстовый процессор). PHP 3 был первой версией, которая уже походила на сегодняшний PHP. Она обеспечивала превосходную расширяемость для поддержки различных баз данных, протоколов и программных интерфейсов. Именно расширяемость PHP 3 привлекла к проекту внимание многих новых разработчиков. К концу 1998 года PHP 3 уже был установлен на ошеломляющих 10% всех веб-серверов.

## Настоящее

Сегодня язык PHP быстро развивается и поддерживается десятками программистов со всего мира, входящих в состав основной команды разработчиков. Методики разработки на PHP также меняются. В прошлом программист обычно писал сценарий на PHP и выгружал его на действующий сервер по FTP, в надежде, что он заработает. Такая дикая по современным понятиям стратегия разработки использовалась из-за отсутствия средств локальной разработки.

В настоящее время, мы отказались от FTP, заменив его инструментами управления версиями. Программное обеспечение управления версиями, например Git, позволяет управлять историей развития программного кода, его ветвлением и слиянием ветвей. Локальные среды разработки стали идентичны действующим серверам, благодаря инструментам виртуализации, таким как Vagrant, и инструментам удаленного комплектования, таким как Ansible, Chef и Puppet. Мы используем специализированные PHP-компоненты с помощью менеджера зависимостей Composer. Наш PHP-код соответствует общественным стандартам PSR, курируемых PHP Framework Interop Group. Мы тщательно тестируем свой код с помощью инструментов, таких как PHPUnit. Разворачиваем свои приложения с помощью менеджера PHP-процессов FastCGI на веб-серверах, таких как nginx. И увеличиваем производительность приложений с помощью кэширования байт-кода.

Современный язык PHP поддерживает множество новых методов, которые могут быть вам неизвестны, они либо являются новыми для PHP либо уже имелись в старых версиях PHP и претерпели существенные изменения. Но это не должно вас волновать. Я подробно остановлюсь на каждом таком методе в этой книге.

Я очень рад, что PHP теперь имеет официальную проектную спецификацию, которой так не хватало до 2014 года.



---

Большинство зрелых языков программирования имеют спецификацию. Говоря простыми словами, спецификация PHP – это набор правил, определяющих, что именно значит быть языком PHP. Спецификация предназначена для разработчиков, создающих программы для анализа, интерпретации и выполнения кода PHP. Она не предназначена для разработчиков приложений и веб-сайтов на PHP.

---

Сара Гоулман (Sara Golemon) и Facebook анонсировали первую проектную спецификацию PHP на конференции OSCON O'Reilly в 2014 году. Официальный анонс можно найти во внутреннем списке рассылки PHP (<http://bit.ly/php-internals>), а спецификацию PHP – на GitHub (<http://bit.ly/php-langspec>).

Официальная спецификация языка PHP становится еще более важной, с учетом появления нескольких конкурирующих между собой движков PHP. Оригинальным движком PHP является Zend Engine (<http://www.zend.com/en/company/community/php/>) – PHP-

интерпретатор, написанный на С и введенный в РНР 4. Движок Zend Engine был создан Расмусом Лердорфом, Энди Гутманом и Зеев Су-раски. Сегодня Zend Engine является основным вкладом компании Zend в РНР-сообщество. Однако в настоящее время появился еще один крупный движок – HipHop Virtual Machine от Facebook. Спецификация языка гарантирует, что оба движка будут обладать базовой совместимостью.



---

РНР-движок – это программа для анализа, интерпретации и выполнения РНР-кода (например, Zend Engine или HipHop Virtual Machine). Его не следует путать с РНР – названием языка РНР в общем смысле.

---

## Будущее

Движок Zend Engine развивается быстрыми темпами, обрстая новыми функциями и улучшая производительность. Я объясняю ускоренное прогрессирующее развитие движка Zend Engine появлением новых конкурентов, а именно движка *HipHop Virtual Machine* от компании Facebook и языка программирования *Hack*.

Hack является новым языком программирования, надстройкой над РНР. Он вводит статическую типизацию, новые структуры данных и дополнительные интерфейсы, сохраняя при этом, обратную совместимость с существующей динамической типизацией в РНР. Язык Hack ориентирован на разработчиков, которые ценят быстроту разработки, присущую РНР, но нуждаются в предсказуемости и стабильности статической типизации.



---

Мы обсудим преимущества и недостатки динамической и статической типизации в этой книге ниже. Разница между ними состоит в моменте проверки типов интерпретатором РНР. Динамические типы проверяются во время выполнения, а статические – во время компиляции. Дополнительную информацию вы найдете в главе 12.

---

Движок HipHop Virtual Machine (HHVM) – это интерпретатор языков РНР и Hack, выполняющий *динамическую компиляцию* (just-in-time, сокращенно JIT) для улучшения производительности приложений и экономии памяти.

Конец ознакомительного фрагмента.  
Приобрести книгу можно  
в интернет-магазине  
«Электронный универс»  
[e-Univers.ru](http://e-Univers.ru)