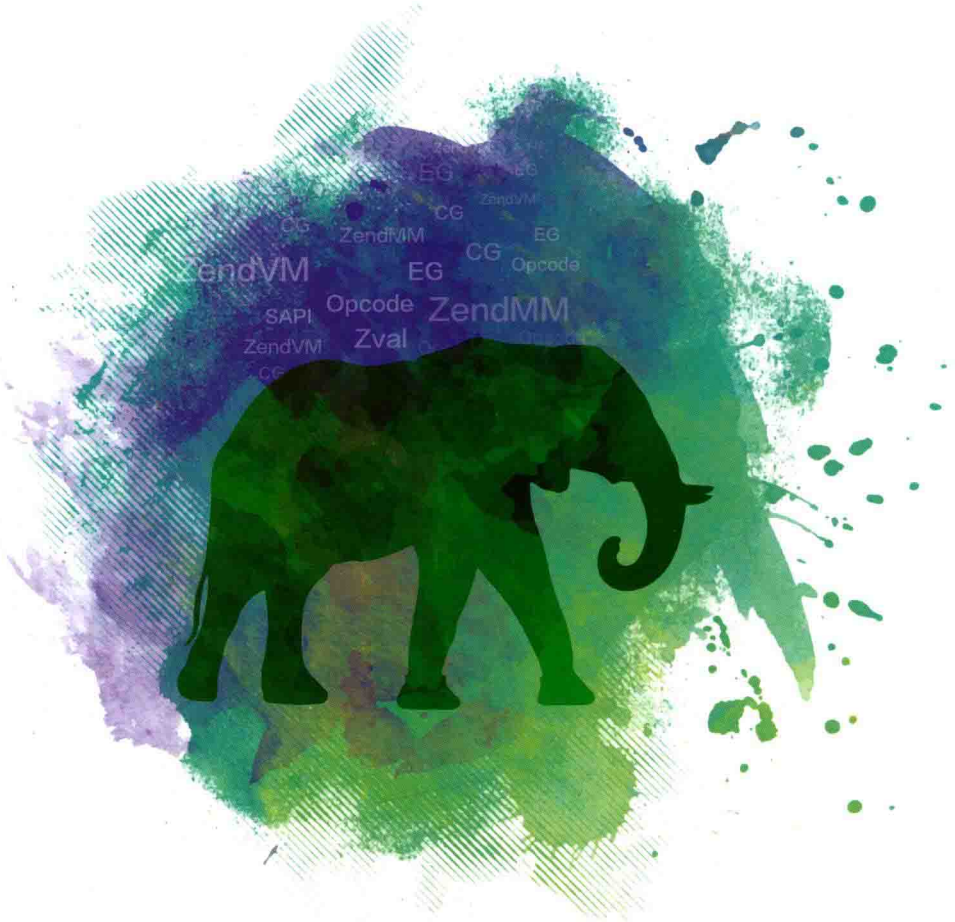
Broadview
www.broadview.com.cn

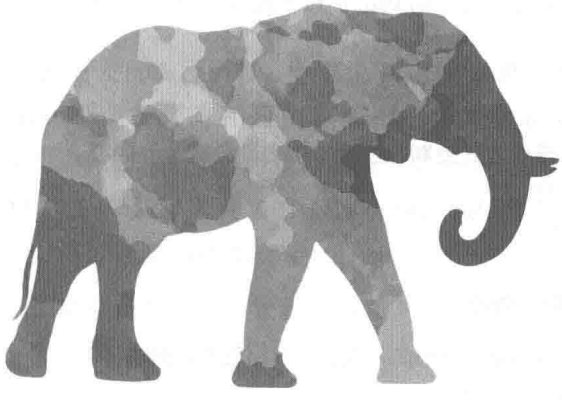


PHP7内核剖析

基于PHP7版本，围绕PHP的SAPI、数据类型、内存管理、编译与执行、函数、类及基础语法等的实现，深刻揭示PHP底层Zend引擎的实现原理

秦朋 / 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn



PHP7内核剖析

秦朋 / 著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

PHP 作为最流行的语言之一，自第一个版本发布至今的二十几年里经历了多次重大改进，PHP7 版本最大的特色在于其性能上的突破，比 PHP5 快了一倍。目前 PHP7 已经得到了广泛应用，越来越多的项目从 PHP5 迁移到了 PHP7。目前，关于 PHP 内核的资料非常有限，本书以当前最为流行的 PHP7 版本为基础，系统性地、尽可能详细地介绍 PHP 语言底层的实现，旨在帮助更多的开发者进一步理解 PHP，参与到 PHP 的实现中，为未来 PHP 的发展贡献一份力量！全书内容主要包括 PHP 数据类型的实现、PHP 的编译及执行、PHP 内存的管理、函数及面向对象的实现、PHP 基础语法的实现，以及 PHP 扩展的开发。

本书适用于有一定 C 语言基础的 PHP 高级工程师，或者想了解 PHP7 的内部实现、扩展开发的工程师。

图书在版编目（CIP）数据

PHP7 内核剖析 / 秦朋著. —北京：电子工业出版社，2017.10
ISBN 978-7-121-32810-7

Ⅰ. ①P… Ⅱ. ①秦… Ⅲ. ①PHP 语言－程序设计 Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字（2017）第 242680 号

责任编辑：陈晓猛
印　　刷：北京京科印刷有限公司
装　　订：北京京科印刷有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×980　1/16　　印张：30.75　　字数：590 千字
版　　次：2017 年 10 月第 1 版
印　　次：2017 年 10 月第 1 次印刷
定　　价：89.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

序一

PHP 是一门优秀的 Web 开发的编程语言，一直说 PHP 是世界“最优秀”的语言，其他各个语言，包括 Python/Java 等语言都有相应的源码剖析或者内核解读之类的书籍，哪怕 MySQL/Redis 等都有相应的源码解读书籍。但是目前的图书市场，除了零零碎碎的一些 PHP 内核描述的文章，真正关于内核的书籍只有英文的《*Extending and Embedding PHP*》，中文电子版的《TIPI：深入理解 PHP 内核》算是相对比较专业的描述 PHP 内核特性的书籍。

本书从整个 PHP 的内部数据结构到内存管理 GC，从 PHP 脚本的编译执行原理到扩展开发，都有详实、深入的描述，是一本国内难得的描述 PHP 内核的佳作，非常值得推荐。看完以后，对整个 PHP 的内部理解，会更上一个新档次。

秦朋是我曾经在 360 公司的同事，多年前他就表现出对 PHP 技术的巨大兴趣，经过几年的努力和对内核的深入阅读理解，终于编写了本书。我读完内容，感慨万千，对很多 PHP 内部技术细节都理解非常透彻，并且文风通俗易懂，代码翔实清晰，确实是对 PHP 下了很深的功夫，也体现了不俗的技术水平，对秦朋的努力感到骄傲和佩服。希望本书能够给中国 PHP 行业带来新的理解和血液，为 PHP 程序员们带来提高和成长，也让我国 PHP 技术真正提升到一个新的台阶。

——谢华亮（黑夜路人）

序二

认识作者，是在公司内网发现他分享了几个关于 PHP 内核的文章，后来在钉钉上直接找他交流是否能转载到我的 PHP 饭米粒的公众号上，一来二去，就熟悉了。

世面上 PHP 的书籍不少，但对于 PHP 内核分析的书很少，能分析这么透彻就更少了，以前对 PHP5.x 版本做过粗略的分析，对 PHP7 的变化其实了解并不多，当作者把电子初版给我之后，一口气看了前面几章，从 SAPI 到 ZVAL 都写得很透彻，收获颇多。

当由 PC 互联网转战到移动互联网、物联网后，PHP 的优势确实小了，一些新的语言也陆陆续续冒出，也受到了不同程度的热捧，其实这些高并发、非阻塞都不是什么高大上的概念，大多数常用语言都能实现，但目前很多人并没有修练好内功，一但碰到问题，可能就会转向那些可以直接补坑的新事物上，而不是去理解这门语言。可以预料到的是，一但在新的语言上碰到坑，又会转向另一个，周而复始，对于自己，基本没有提高，所以透过现象看本质很有必要，也就能一通百通了。

另一方面，现在的人都比较浮躁，很少有年轻人能够沉下心去做深入的研究，我从业 10 多年，面试的人也众多，大多数人在工作三年左右就会遇到一个瓶颈，主要原因是对业务非常熟悉了，也没有挑战了，就想通过换个环境来找新鲜感。而有些人可以自我蜕变，从日常的业务中找到感兴趣的点深入学习，就如本书的作者一样，这给大多数人也指明了另一种突破的方式。

最后建议 PHPer 都可以精读此书，你就可以知道为什么 PHP 的一个变量类型可以变来变去，也可以知道为什么 PHP 的数组这么强大，深入之后，一定会为你打开一扇新的大门，让你在技术的道路上走得更扎实。

——王晶（滴滴技术专家，Swoole 开发者）

前　言

为什么要写这本书

PHP 作为最流行的语言之一，自第一个版本发布至今的二十几年里经历了多次重大的改进，尤其是 PHP7 版本的发布，其最大的亮点在于性能上的提升，比 PHP5 快了一倍。随着 PHP7 的不断普及，越来越多的项目从 PHP5 迁移到了 PHP7，毫无疑问，PHP7 将成为 PHP 历史上里程碑式的一个版本。我是在大学时代接触到的 PHP，初次相识就被其简洁、易用的语法所吸引了。在工作后的几年里，我一直使用 PHP 作为主要的开发语言。当然，除了 PHP，我也使用过很多其他语言，比如 C、C++、Java、Golang、Python 等，不同的语言有各自的特点、优势，让我印象最深的、也让我最喜欢的有 C、Golang、PHP。

- C

  这是我评价最高的一门语言，其强大的操控能力、简洁的语法、易于理解的处理方式无一不让我折服。编程语言本身只是控制计算机的一种工具，然而很多高级语言过度隔离了人与计算机间的联系，使得编程者并不理解计算机实际的工作机制，只能被编程语言限定在固定范围内，而 C 语言在这一点上做得恰到好处，其没有过度干预我们对计算机的操控，允许我们自由地控制内存、CPU。当然，C 语言也有很多不方便的地方，过于简单的接口使得很多操作不得不通过编写大量的代码来实现。

- Golang

  并发是我对其最大的印象，我们可以用更容易理解的方式来实现并发，但是它的内存控制没有 C 语言那么方便、灵活。

- PHP

  PHP 的底层是 C 语言实现的，这也使得它继承了很多 C 语言的基因，PHP 的简洁、易用、学习成本低等特点成就了它今天的地位。

PHP 的高度封装性与弱类型的特点使得很多操作极其简便，例如 JSON 的解析如果在 Golang 中完成，则需要定义一系列的结构体，然后才能完成解析，而在 PHP 中通过一行代码就可以完成。正是 PHP 底层的强大才得以实现如此简便的操作，那么强大的 PHP 背后到底是什么样子的呢？我想很多 PHPer 都有过这个疑问。然而让人感到沮丧的是，关于 PHP 内核的资料非常有限，已有的这些资料也不全面、系统，多数局限在理论介绍的层面上。后来我就直接去读 PHP 的源码，渐渐地发现，以前很多不理解的问题都在源码中找到了答案。本书主要的出发点是给那些想要了解 PHP 底层实现的读者一些启发，帮助更多的人理解 PHP 的实现，甚至能够参与到 PHP 的开发中，为未来 PHP 的发展贡献一份力量！

本书适合的对象

- 有一定 C 语言基础的读者。
- 想要理解 PHP 内部实现的读者。
- PHP 高级工程师。
- 对虚拟机实现感兴趣的读者。

本书不适合作为 PHP 的入门教程。书中对于基础性的、概念性的东西介绍很少，重点是源码解析。

本书的结构

本书总共分为 10 章，章节之间存在一定的衔接，建议按照先后顺序阅读。其中第 3～第 9 章为 Zend 引擎相关的内容，也是本书的核心章节。

第 1 章介绍 PHP 的基础内容。本章主要介绍 PHP 的历史发展、PHP7 的主要变化，重点讲解 PHP 的构成部分与生命周期的几个阶段。

第 2 章介绍 SAPI。本章选取了 PHP 三种常见的应用场景，介绍三个不同 SAPI 的实现：Cli、Fpm、Embed。SAPI 是 PHP 的接入层，如果只想了解 Zend 引擎的内容，那么可以跳过本章。

第 3 章介绍数据类型。本章主要介绍 PHP 中变量的基础结构 zval，以及不同类型的结构，它们是 PHP 中最基础的、使用最频繁的数据结构，通过本章的内容你将了解 PHP 中变量的内部实现。

第 4 章介绍内存管理。本章主要介绍 PHP 变量自动回收机制的实现，以及 PHP 底层内存

池、线程安全相关的实现。通过本章的内容，你将了解变量的内存是如何进行管理的，为什么PHP中的变量不需要手动申请释放。其中内存池的实现比较独立，它的实现与tcmalloc类似；线程安全只在多线程环境下使用，常见的Fpm、Cli模式不会用到，本书其他章节介绍的内容都是非线程安全的。

第5章介绍PHP的编译与执行。本章介绍PHP代码从编译到执行的整个过程，这也是Zend引擎的核心实现。通过对本章的学习，你将了解PHP代码是如何被Zend引擎识别、执行的。

第6章介绍函数的实现。本章介绍PHP中函数的实现，这也是Zend引擎的核心部分，本章的内容与第5章相关，介绍函数的编译与执行。

第7章介绍面向对象。本章介绍面向对象相关的实现，主要包括类、对象的内部实现。

第8章介绍命名空间。本章介绍PHP中命名空间的实现，这部分内容比较简单，命名空间只涉及编译阶段。

第9章介绍基础语法的实现。本章主要介绍PHP中基础语法的实现，比如条件分支、循环结构、中断跳转、静态变量、常量、全局变量、文件加载等，这些语法涉及PHP的编译、执行，它们是PHP语言的基础组成部分。通过对本章的学习，你可以更全面地掌握PHP语言的实现。

第10章介绍扩展开发。本章的内容偏向应用性，主要介绍扩展开发中常用的一些接口、宏。

勘误与支持

因个人水平有限，以及时间比较仓促，书中难免有不足之处，还望读者批评指正。如果你对本书有比较好的建议或对书中内容有所疑惑，可与我联系。

Email：pangudashu@gmail.com；QQ群：103330909

致谢

首先感谢PHP7的主要开发者鸟哥与PHP社区的其他开发者，正是他们的智慧造就了PHP，期待未来PHP能够有更加广阔的发展空间。在这里尤其要感谢Swoole的创始人韩天峰老师，本项目有幸得到韩老师的推荐，得到了众多人的关注。另外要单独感谢陈晓猛编辑，在他耐心地指导、审稿、修改工作下，最终才有了本书的诞生。

秦朋

读者服务

轻松注册成为博文视点社区用户（www.broadview.com.cn），扫码直达本书页面。

- **下载资源**：本书如提供示例代码及资源文件，均可在下载资源处下载。
- **提交勘误**：您对书中内容的修改意见可在提交勘误处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- **交流互动**：在页面下方读者评论处留下您的疑问或观点，与我们和其他读者一同学习交流。

页面入口：http://www.broadview.com.cn/32810

目　录

第 1 章 PHP 基础架构

本章将简单介绍 PHP 的基本信息，以及 PHP 的安装、调试，同时将介绍 PHP 生命周期中的几个阶段，它们是 PHP 整个流程中比较关键的几个阶段。

1.1 简介

PHP 是一种非常流行的高级脚本语言，尤其适合 Web 开发，快速、灵活和实用是 PHP 最重要的特点。PHP 自 1995 年由 Lerdorf 创建以来，在全球得到了非常广泛的应用。

PHP 在 1995 年早期以 Personal Home Page Tools（PHP Tools）开始对外发表第一个版本，Lerdorf 写了一些介绍此程序的文档，并且发布了 PHP1.0。在这早期的版本中，提供了访客留言本、访客计数器等简单的功能，之后越来越多的网站开始使用 PHP，并且强烈要求增加一些特性，在新的成员加入开发行列之后，Rasmus Lerdorf 在 1995 年 6 月 8 日将 PHP 公开发布，希望可以通过社群来加速程序开发与寻找错误。这个版本被命名为 PHP2，已经有了今日 PHP 的一些雏型，类似 Perl 的变量命名方式、表单处理功能，以及嵌入到 HTML 中执行的能力。程序语法上也类似 Perl，有较多的限制，不过更简单、更有弹性。PHP/FI 加入了对 MySQL 的支持，从此建立了 PHP 在动态网页开发上的地位。到了 1996 年年底，有 15000 个网站使用了 PHP。

在 1997 年，任职于 Technion IIT 公司的两个以色列程序设计师 Zeev Suraski 和 Andi Gutmans 重写了 PHP 的解析器，成为 PHP3 的基础，而 PHP 也在这个时候改称为 PHP：Hypertext Preprocessor，1998 年 6 月正式发布 PHP3。Zeev Suraski 和 Andi Gutmans 在 PHP3 发布后开始

改写 PHP 的核心，这个在 1999 年发布的解析器被称为 Zend Engine，他们也在以色列的 Ramat Gan 成立了 Zend Technologies 来管理 PHP 的开发。

在 2000 年 5 月 22 日，以 Zend Engine 1.0 为基础的 PHP4 正式发布。2004 年 7 月 13 日发布了 PHP5，PHP5 则使用了第二代的 Zend Engine。PHP 包含了许多新特色：完全实现面向对象，引入 PDO，以及许多性能方面的改进。目前 PHP5.x 仍然是应用非常广泛的一个版本。

PHP 独特的语法混合了 C、Java、Perl 及 PHP 自创新的语法，同时支持面向对象、面向过程，相比 C、Java 等语言具有语法简洁、使用灵活、开发效率高、容易学习等特点。

- 开源免费：PHP 社群有大量活跃的开发者贡献代码。
- 快捷：程序开发快，运行快，技术本身学习快，实用性强。
- 效率高：PHP 消耗相当少的系统资源，自动 gc 机制。
- 类库资源：有大量可用类库供开发者使用。
- 扩展性：允许用户使用 C/C++扩展 PHP。
- 跨平台：可以在 UNIX、Windows、Mac OS 等系统上使用 PHP。

很多人认为 PHP 非常简单，没什么技术含量，这是非常片面的认识，任何语言都有其存在的价值、有其适合的应用领域，正是 PHP 底层的强大才造就了 PHP 语言的简洁、易用，这反而更能够提现出它的优秀所在。

1.2 安装及调试

在学习 PHP 内核之前，首先需要安装 PHP，以方便在学习过程中进行调试。本书使用的 PHP 版本为 7.0.12，下载地址为 http://php.net/distributions/php-7.0.12.tar.gz，下载后使用以下命令进行编译、安装：

```
$ tar zxvf php-7.0.12.tar.gz
$ cd php-7.0.12
$ ./configure --prefix=/usr/local/php7 --enable-debug --enable-fpm
```

--enable-debug 参数为开启 debug 模式，方便我们进行调试。关于调试自然少不了 gdb 了，PHP 内核的实现虽然比较复杂，但是阶段划分比较鲜明，可以通过 gdb 在各个阶段设置断点，然后进行相应的调试。学习内核时，可以使用 Cli 模式，因为它是单线程的，方便调试，这并不影响我们对内核的学习。同时，想要弄清楚 PHP 内核，自然少不了阅读 PHP 的源码，因此本书后面介绍的内容将会非常频繁地列举源码，而对于一些概念性的解释则点到为止。本书主要的目的是引导大家自己去阅读源码、探索 PHP 的实现，而不希望只简单地通过书中的描述来

了解 PHP，所以希望大家在阅读本书时，一定要准备好一份源码以便随时查看和调试。

1.3 PHP7 的变化

PHP7 与 PHP5 版本相比有非常大的变化，尤其是在 Zend 引擎方面。为提升性能，PHP7 对 Zend 进行了深度优化，使得 PHP 的运行速度大大提高，比 PHP5.0～5.6 快了近 5 倍，同时还降低了 PHP 对系统资源的占用。下面介绍 PHP7 比较大的几个变化。

1）抽象语法树

在 PHP 之前的版本中，PHP 代码在语法解析阶段直接生成了 ZendVM 指令，也就是在 zend_language_parser.y 中直接生成 opline 指令，这使得编译器与执行器耦合在一起。编译生成的指令供执行引擎使用，该指令是在语法解析时直接生成的，假如要把执行引擎换成别的，就需要修改语法解析规则；或者如果 PHP 的语法规则变了，但对应的执行指令没有变化，那么也需要对修改语法解析规则。

PHP7 中增加了抽象语法树，首先是将 PHP 代码解析生成抽象语法树，然后将抽象语法树编译为 ZendVM 指令。抽象语法树的加入使得 PHP 的编译器与执行器很好地隔离开，编译器不需要关心指令的生成规则，然后执行器根据自己的规则将抽象语法树编译为对应的指令，执行器同样不需要关心该指令的语法规则是什么样子的。

2）Native TLS

开发过 PHP5.x 版本扩展的读者对 TSRM_CC、TSRM_DC 这两个宏一定不会陌生，它们是用于线程安全的。PHP 中有很多变量需要在不同函数间共享，多线程的环境下不能简单地通过全局变量来实现，为了适应多线程的应用环境，PHP 提供了一个线程安全资源管理器，将全局资源进行了线程隔离，不同的线程之间互不干扰。

使用全局资源需要先获取本线程的资源池，这个过程比较占用时间，因此，PHP5.x 通过参数传递的方式将本线程的资源池传递给其他函数，避免重复查找。这种实现方式使得几乎所有的函数都需要加上接收资源池的参数，也就是 TSRM_DC 宏所加的参数，然后调用其他函数时再把这个参数传下去，不仅容易遗漏，而且这种方式极不优雅。

PHP7 中使用 Native TLS（线程局部存储）来保存线程的资源池，简单地讲就是通过__thread 标识一个全局变量，这样这个全局变量就是线程独享的了，不同线程的修改不会相互影响。具体的实现在本书第 4 章会详细介绍。

3）指定函数参数、返回值类型

PHP7 中可以指定函数参数及返回值的类型，例如：

```
function foo(string $name): array {
```

```
    return [];
}
```

这个函数的参数必须为字符串，返回值必须是数组，否则将会报 error 错误。

4）zval 结构的变化

zval 是 PHP 中非常重要的一个结构，它是 PHP 变量的内部结构，也是 PHP 内核中应用最为普遍的一个结构。在 PHP5.x 中，zval 的结构是下面这个样子的：

```
struct _zval_struct {
    /* Variable information */
    zvalue_value value;     /* value */
    zend_uint refcount__gc;
    zend_uchar type;    /* active type */
    zend_uchar is_ref__gc;
};
```

type 为类型，is_ref__gc 标识该变量是否为引用，value 为变量的具体值，它是一个 union，用来适配不同的变量类型：

```
typedef union _zvalue_value {
    long lval;                  /* long value */
    double dval;                /* double value */
    struct {
        char *val;
        int len;
    } str;
    HashTable *ht;              /* hash table value */
    zend_object_value obj;
    zend_ast *ast;
} zvalue_value;
```

zval 中还有一个比较重要的成员：refcount__gc，它记录变量的引用计数。引用计数是 PHP 实现变量自动回收的基础，也就是记录一个变量有多少个地方在使用的一种机制。PHP5.x 中引用计数是在 zval 中而不是在具体的 value 中，这样一来，导致变量复制时需要复制两个结构，zval、zvalue_value 始终绑定在一起。PHP7 将引用计数转移到了具体的 value 中，这样更合理。因为 zval 只是变量的载体，可以简单地认为是变量名，而 value 才是真正的值，这个改变使得 PHP 变量之间的复制、传递更加简洁、易懂。除此之外，zval 结构的大小也从 24byte 减少到了