

<<高性能PHP应用开发>>

图书基本信息

书名：<<高性能PHP应用开发>>

13位ISBN编号：9787115264954

10位ISBN编号：7115264953

出版时间：2011-11

出版单位：人民邮电出版社

作者：Armando Padilla, Tim Hawkins

页数：177

译者：刘霞, 盛海艳

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<高性能PHP应用开发>>

内容概要

本书是一本广受好评的PHP性能优化方面的图书，通过介绍PHP的原理和相关的工具集来实现调优性能的目的。

它分析和研究了Web应用程序的前端和后端，并系统地提升了其性能和运行效率。

本书还介绍了PHP编码最佳实践的运用以及如何使用工具来应用缓存技术。

另外书中也涉及了对Web服务器的优化和数据库的优化。

本书适合PHP开发人员阅读。

<<高性能PHP应用开发>>

作者简介

Armando

Padilla，专注于PHP技术已有13年，领导并全面参与了基于LAMP的多个网络应用程序。

Armando目前是Yahoo的高级工程师，曾负责多个著名的高流量应用，例如2010年冬奥会和2010年南非世界杯网站、Yahoo手机新闻应用程序。

Tim

Hawkins，早在1993年就创建了站点，这是世界上最早的分类门户网站之一。

之后他帮助Yahoo欧洲公司打造了很多关键产品，例如搜索、本地搜索、邮件、即时通信软件和社会化网络等。

他目前在美国的一家大型电子零售公司管理庞大的海外团队，负责开发和部署下一代电子商务应用程序。

<<高性能PHP应用开发>>

书籍目录

第1章 基准测试技术

- 1.1 PHP应用程序栈
- 1.2 基准测试实用工具
- 1.3 定义请求/响应生命周期
- 1.4 Apache Benchmark
 - 1.4.1 安装Apache Benchmark
 - 1.4.2 运行Apache Benchmark
 - 1.4.3 弄清响应的含义
 - 1.4.4 ab选项标记
 - 1.4.5 ab陷阱
- 1.5 Siege
 - 1.5.1 安装Siege
 - 1.5.2 运行Siege
 - 1.5.3 分析结果
 - 1.5.4 Siege选项标记
 - 1.5.5 测试很多URL
- 1.6 影响基准测试数字
 - 1.6.1 地理位置
 - 1.6.2 旅行的数据包
 - 1.6.3 响应的大小
 - 1.6.4 代码复杂性
 - 1.6.5 浏览器行为
 - 1.6.6 Web服务器设置
- 1.7 小结

第2章 提高客户端下载和呈现性能

- 2.1 优化响应的重要性
- 2.2 Firebug
 - 2.2.1 安装Firebug
 - 2.2.2 Firebug性能选项卡
 - 2.2.3 Console选项卡
 - 2.2.4 Net选项卡
- 2.3 YSlow
 - 2.3.1 YSlow v2规则集
 - 2.3.2 安装YSlow
 - 2.3.3 启动YSlow
- 2.4 Page Speed
 - 2.4.1 安装Page Speed
 - 2.4.2 运行中的Page Speed
- 2.5 优化工具
 - 2.5.1 JavaScript优化
 - 2.5.2 JavaScript的放置位置
 - 2.5.3 精简JavaScript
- 2.6 精简工具
- 2.7 YUI Compressor
- 2.8 Closure Compiler

<<高性能PHP应用开发>>

- 2.8.1 减少资源请求
- 2.8.2 使用服务器端压缩
- 2.9 图像压缩
- 2.10 Smush.it
- 2.11 小结
- 第3章 PHP代码优化
 - 3.1 PHP最佳实践
 - 3.1.1 PHP的经济性
 - 3.1.2 require与require_once
 - 3.1.3 提前计算循环长度
 - 3.1.4 使用foreach、for、while循环访问数组元素
 - 3.1.5 文件访问
 - 3.1.6 更快速地访问对象属性
 - 3.2 使用VLD、strace和Xdebug一探究竟
 - 3.2.1 用VLD查看Opcode函数
 - 3.2.2 使用strace进行C级跟踪
 - 3.3 发现瓶颈
 - 3.3.1 Xdebug 2: PHP调试工具
 - 3.3.2 验证安装
 - 3.3.3 安装基于GUI的工具
 - 3.4 小结
- 第4章 Opcode缓存
 - 4.1 回顾路线图
 - 4.2 PHP的生命周期
 - 4.3 Opcode缓存工具
 - 4.3.1 Alternative PHP Cache
 - 4.3.2 XCache
 - 4.3.3 用XCache缓存
 - 4.3.4 XCache设置
 - 4.3.5 eAccelerator
 - 4.3.6 eA设置
 - 4.4 小结
- 第5章 变量缓存
 - 5.1 应用程序的性能路线图
 - 5.2 实现变量缓存的价值
 - 5.3 示例项目: 创建表
 - 5.3.1 获取记录
 - 5.3.2 计算读取数据库的开销
 - 5.4 APC缓存
 - 5.4.1 将数据添加到缓存中
 - 5.4.2 对APC进行基准测量
 - 5.5 Memcached
 - 5.5.1 安装Memcached
 - 5.5.2 启动Memcached服务器
 - 5.5.3 在PHP中使用Memcached
 - 5.6 小结
- 第6章 选择正确的Web服务器

<<高性能PHP应用开发>>

- 6.1 选择适合你的Web服务器程序包
 - 6.1.1 安全性和稳定性非常重要
 - 6.1.2 找到具有丰富知识的工程师非常重要
 - 6.1.3 你的网站主要是静态内容
 - 6.1.4 你在托管服务中托管
 - 6.1.5 你正在使用不常见的PHP扩展
- 6.2 Web服务器的使用情况图表
- 6.3 Web服务器请求的处理
- 6.4 Web服务器硬件
- 6.5 对Web服务器进行分类
- 6.6 Apache HTTPD
 - 6.6.1 Apache Daemon命令行
 - 6.6.2 Apache多处理模块
- 6.7 了解Apache模块
 - 6.7.1 添加动态Apache模块
 - 6.7.2 删除动态Apache模块
- 6.8 关于Apache的最后几点
- 6.9 lighttpd
 - 6.9.1 安装lighttpd
 - 6.9.2 lighttpd配置设置
 - 6.9.3 比较静态负载内容
 - 6.9.4 在lighttpd上安装PHP
- 6.10 Nginx
 - 6.10.1 安装Nginx
 - 6.10.2 Windows安装
- 6.11 Nginx作为静态Web服务器
 - 6.11.1 安装FastCGI PHP
 - 6.11.2 Nginx基准测试
- 6.12 小结
- 第7章 优化Web服务器和内容交付
 - 7.1 测定Web服务器的性能
 - 7.2 了解应用程序的内存占用情况
 - 7.3 优化Apache中的进程
 - 7.3.1 控制Apache客户端(PreforkMPM)
 - 7.3.2 优化内存使用和防止产生交换
 - 7.4 其他Apache配置调整
 - 7.4.1 使用.htaccess文件和AllowOverride
 - 7.4.2 使用FollowSymlinks
 - 7.4.3 使用DirectoryIndex
 - 7.4.4 关闭HostnameLookup
 - 7.4.5 启用Keep-Alive
 - 7.4.6 使用mod_deflate压缩内容
 - 7.5 扩展到单台服务器之外
 - 7.5.1 使用Round-Robin DNS
 - 7.5.2 使用负载均衡器
 - 7.5.3 使用直接服务器返回
 - 7.5.4 在服务器场的成员之间共享会话

<<高性能PHP应用开发>>

- 7.5.5 与共享文件系统共享资产
- 7.5.6 与独立资产服务器共享资产
- 7.5.7 与内容分发网络共享资产
- 7.6 使用分布式架构的陷阱
 - 7.6.1 缓存一致性问题
 - 7.6.2 缓存版本问题
 - 7.6.3 用户IP地址跟踪
 - 7.6.4 多米诺骨牌或级联失败效应
 - 7.6.5 部署失败
- 7.7 监视应用程序
- 7.8 小结
- 第8章 数据库优化
 - 8.1 MySQL简介
 - 8.2 了解MySQL存储引擎
 - 8.2.1 MyISAM：原始引擎
 - 8.2.2 InnoDB：专业级的选择
 - 8.2.3 选择存储引擎
 - 8.3 了解MySQL如何使用内存
 - 8.3.1 InnoDB与MyISAM内存使用的比较
 - 8.3.2 每服务器与每连接(线程)内存使用的比较
 - 8.4 查找配置文件
 - 8.4.1 Mysqltuner.pl：优化数据库服务器的内存
 - 8.4.2 示例服务器可能出现的问题
 - 8.4.3 优化InnoDB
 - 8.5 找到有问题的查询
 - 8.6 分析有问题的查询
 - 8.7 PHP数据库应用程序的建议
 - 8.7.1 保持独立的读写连接
 - 8.7.2 默认使用“utf_8”(多字节Unicode)字符集
 - 8.7.3 使用“UTC”日期格式
 - 8.8 小结
- 附录A 在Windows上安装Apache、MySQL、PHP和PECL
- 附录B 在Linux上安装Apache、MySQL、PHP和PECL

<<高性能PHP应用开发>>

章节摘录

版权页：插图：8.7.1 保持独立的读写连接开始就创建两个数据库连接是一个好的方法，一个用于读取，一个用于写入，并且允许不同的数据库服务器连接它们。

如果你只有一个服务器，则将它们设置为彼此相同。

当你进行应用程序的编码时，可以把更改数据的任何查询（UPDATE、INSERT、DELETE等）都写成使用写入连接，纯SELECT或读取查询则一律使用读取连接。

如果你需要升级你的应用程序，则可以将数据库服务器分离到其他计算机上，并通过复制来连接它们。

但若要实现这一工作，必须确保所有写入都指向你的主要服务器，所有读取都指向适当的从属服务器。

通过使用两个连接，可以轻松重新配置你的应用程序以支持大量不同的扩展选项，使用一个或多个从属服务器来增加查询带宽。

从一开始就实现这种方案只需要很少的努力，但之后却会大大增加你的选择。

<<高性能PHP应用开发>>

编辑推荐

《高性能PHP应用开发》：全球已有超过百万的程序员从事PHP开发，而任何认真的程序员均需要了解如何提升PHP项目的性能。

《高性能PHP应用开发》专门研究了这一课题。

《高性能PHP应用开发》将深入探讨在应用程序运行中起重要作用的所有技术和组件。

现在是数秒间就决定能否留住用户的时代，所有人都必须把优化作为项目路线图的必备环节。

但到底应该分析应用程序中的哪些组件呢？

应该怎样优化，又该如何测量应用程序的执行性能呢？

这些正是《高性能PHP应用开发》要回答的问题，《高性能PHP应用开发》的内容还包括：为什么应该优化某个特定的组件，为什么优化某个函数会比优化另一个更有效果，如何寻找和使用面向开源社区的优化工具，如何部署缓存软件和Web服务器软件。

此外，《高性能PHP应用开发》还会讲解更多高级技巧，包括：使用Xdebug来分析一些没有实现最佳运行效率的函数；比较不同的PHP函数所执行的opcode，从而搜索到运行效率最高的函数；当应用程序正在为用户提供服务时，使用strace来分析Apache。

读完《高性能PHP应用开发》后，读者会对从哪里开始优化形成完整的认识。

最重要的是，在未来优化PHP应用程序时，将会拥有趁手的工具来助一臂之力。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>