第 13 章　PHP 加速器

PHP 代码在运行时是通过编译器首先编译成中间代码，然后再被服务器运行得到用户所需要的结果。因此，中间代码的优劣直接决定了代码的最终运行速度。目前，有一些常见的 PHP 加速器可以通过对中间代码进行优化来提高 PHP 代码的运行速度。这样，网站的访问者就可以更快的打开网页，提高工作效率和客户满意度。

本章将介绍几款常见的 PHP 加速器，读者可以根据实际情况选择其中的一款安装，可以有效的提高 PHP 的运行效率。

13.1　Zend Optimizer

Zend Optimizer 是 PHP 的开发商 Zend 公司开发的官方版 PHP 加速器，其优化效果非常明显。Zend Optimizer 可以从其官方网站 http://www.zend.com/上下载到，目前的最新版本是 3.0.2。

13.1.1　Zend Optimizer 的安装

Zend Optimizer 提供多个操作系统的支持，本节仅以 Windows 下的安装为例简要说明一下 Zend Optimizer 的安装步骤。下载后的 Zend Optimizer 文件名为 ZendOptimizer-x.x.x-Windows-i386.exe，其中 x.x.x 为版本号，安装步骤如下所示。

（1）双击 ZendOptimizer-x.x.x-Windows-i386.exe 文件，启动安装程序。

（2）在【Zend Optimizer – InstallShield Wizard】对话框上单击【Next】按钮，弹出【ZendOptimizer – 3.0.2 License Agreement】对话框。

（3）阅读并接受协议后，单击【Next】按钮，弹出【Choose Destination Folder】对话框。

（4）选择要安装的目录后，单击【Next】按钮，弹出【Choose Web server】对话框，如图 13-1 所示。

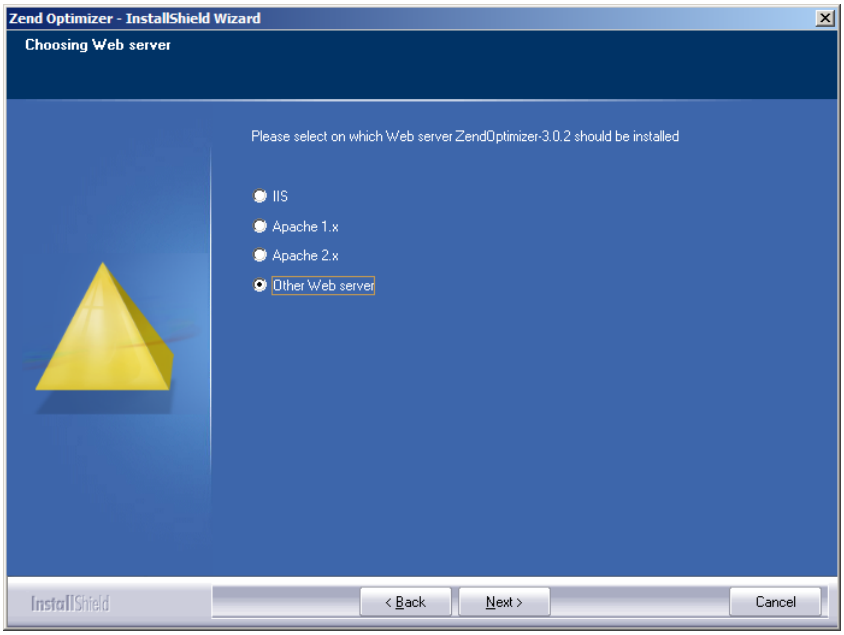

图 13-1　选择 Web 服务器类型

（5）选择服务器上使用的服务器类型后，单击【Next】按钮，弹出【Choose the php.ini folder】对话框。

（6）选择好 php.ini 所在目录后，单击【Next】按钮，弹出【Choose the Web server's root folder】对话框。

（7）选择好 Web 服务器所在目录后，单击【Next】按钮，弹出【Pre-Install Summary】对话框，这里将显示当前选择的服务器类型和安装路径，如图 13-2 所示。

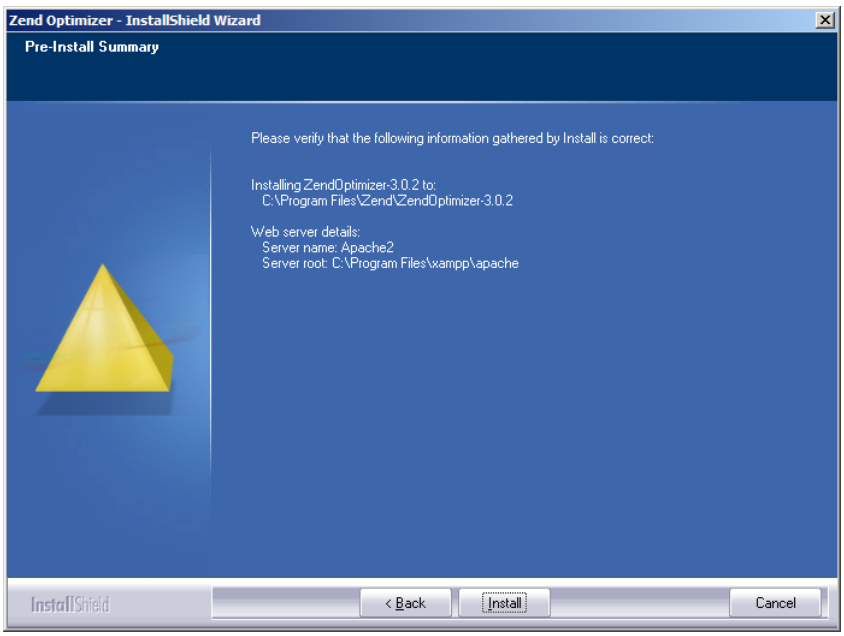

图 13-2　【Pre-Install Summary】对话框

（8）手动关闭 Web 服务器，如果不关闭服务器，Zend Optimizer 安装程序也会自动尝试关闭 Apache 服务器。但是，如果 Apache 服务器没有安装成 Windows 服务，Zend Optimizer 安装程序将无法完成关

闭操作并显示如图 13-3 所示的错误提示。

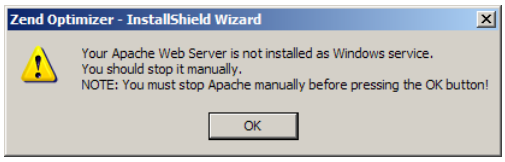

图 13-3　错误提示

（9）确认安装信息后，单击【Install】按钮开始安装。

（10）安装结束后，单击【Finish】按钮。

（11）手动启动 Apache 服务器。

安装后，可以在 PHP 安装目录下的 php.ini 中看到如下所示的代码。

```
[Zend]
; Local Variables:
; tab-width: 4
; End:
zend_extension_manager.optimizer_ts="C:\Program Files\Zend\ZendOptimizer-3.0.2\lib\Optimizer-3.0.1"
zend_extension_ts="C:\Program Files\Zend\ZendOptimizer-3.0.2\lib\ZendExtensionManager.dll"
```

同时，在浏览器上运行如下代码可以看到如图 13-4 所示的结果。

```
<?php
phpinfo();
?>
```

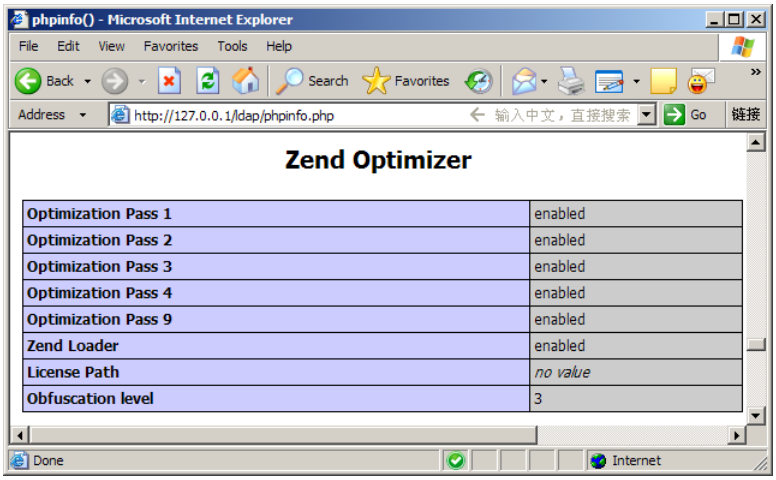

图 13-4　验证 Zend Optimizer 安装成功

13.1.2　Zend Optimizer 的配置

Zend Optimizer 的配置可以通过修改 php.ini 来实现，主要的配置参数有两项。如果需要修改某个参数，可以直接在 php.ini 中声明。

1．zend_optimizer.encoder_loader

该参数用于声明是否处理由 Zend Encoder 加密的 PHP 文件，0 表示不允许，1 表示允许。

2．zend_optimizer.optimization_level

该参数用于声明优化程度，也就是启动的优化进程数。在 Zend Optimizer 中包含 10 个优化进程，

每个优化进程的对应代码如表 13-1 所示。

表 13-1　Zend Optimizer的优化进程

优化进程名称	对应代码
不使用	0
优化进程1（Optimization Pass 1）	1
优化进程2（Optimization Pass 2）	2
优化进程3（Optimization Pass 3）	4
优化进程4（Optimization Pass 4）	8
优化进程5（Optimization Pass 5）	16
优化进程6（Optimization Pass 6）	32
优化进程7（Optimization Pass 7）	64
优化进程8（Optimization Pass 8）	128
优化进程9（Optimization Pass 9）	256
优化进程10（Optimization Pass 10）	512

zend_optimizer.optimization_level 的值是要启动的进程代码的和。例如，需要启动优化进程 1 到 4 这四个进程，则需要设置 zend_optimizer.optimization_level 为 1+2+4+8 = 15，如下所示。

```
zend_optimizer.optimization_level = 15
```

13.2　PHP Accelerator

PHP Accelerator 是一款免费的 PHP 加速器，可以通过简单的配置实现对 PHP 代码的优化。PHP Accelerator 可以从其官方网站 http://www.php-accelerator.co.uk/上下载到，目前最新的版本是 1.3.3，可以支持对 Linux、Solaris 下的 PHP 代码进行加速。目前，PHP Accelerator 尚不支持 Windows 平台下的 PHP 代码加速。

13.2.1　PHP Accelerator 的安装

本节仅以 Linux 下的 PHP Accelerator 的安装简要说明一下 PHP Accelerator 的安装步骤，其他操作系统的安装与这个类似。

（1）解压缩下载文件到一个指定目录，例如/usr/local/lib。

（2）修改 PHP 安装目录下的 php.ini 文件并增加以下内容

```
;PHP Accelerator
zend_extension="/path/php_accelerator_1.3.3.so"
```

这里，/path 是 PHP Accelerator 的所在目录。

（3）重新启动 Apache 服务器。

此时，PHP Accelerator 就安装完成了。以下代码可以验证 PHP Accelerator 是否安装成功。

```
<?php
    var_dump($GLOBALS['_PHPA']);
?>
```

其中_PHPA 是由 PHP Accelerator 建立的一个全局数组变量，运行结果如下所示。

```
array(3) {
      ["ENABLED"]=> bool(true)
      ["iVERSION"]=> int(10303)
      ["VERSION"]=> string(5) "1.3.3"
```

```
}
```

这里输出了安装的 PHP Accelerator 的版本以及是否可用的信息。如果以上信息能够被正确的输出，则说明 PHP Accelerator 已经被成功的安装成功了。需要注意的是，如果在 PHP 中输出了如下所示的错误信息，则说明当前安装的 PHP Accelerator 版本与 PHP 版本不兼容。

```
Failed loading php_accelerator_1.3.3.so:  php_accelerator_1.3.3.so: undefined symbol: _ecalloc
```

如果出现这种情况，则需要重新去网站上下载与当前 PHP 版本匹配的 PHP Accelerator。

13.2.2 PHP Accelerator 的配置

前面介绍了如何安装 PHP Accelerator，安装后，再次运行 PHP 代码，PHP Accelerator 将被自动调用并对 PHP 代码进行优化和加速处理。

如果需要对 PHP Accelerator 进行配置，可以直接编辑 PHP 安装目录下的 php.ini 文件并增添相应的参数。本节将介绍几个常用的用于 PHP Accelerator 配置的参数。

1．PHP Accelerator 的启动与关闭

启动与关闭 PHP Accelerator 可以通过在 php.ini 文件中增加 phpa 参数来实现，如下所示。

```
phpa = on
```

默认情况下 PHP Accelerator 是启动的，如果需要关闭 PHP Accelerator，只需要将 phpa 参数设置成 off 即可。

2．文件缓存目录的设置

PHP Accelerator 需要使用文件缓存目录来作为其进行优化时存放临时文件，其默认值为/tmp，如下所示。

```
phpa.cache_dir = /tmp
```

如果需要修改文件缓存目录，只需要修改 phpa.cache_dir 参数即可。

3．修改 PHP Accelerator 使用的共享内存大小

与临时文件类似，PHP Accelerator 还需要使用共享内存来为其优化时使用，其大小默认值为 8MB，如下所示。

```
phpa.shm_size = 8
```

如果需要修改共享内存的大小，只需要修改 phpa.shm_size 参数即可。需要注意的是这里只能输入整型数来指代多少兆字节的内存将被 PHP Accelerator 使用。

13.3 Turck MMCache

Turck MMCache 是一款开放源码的 PHP 加速器，与 PHP Accelerator 一样可以通过简单的配置实现对 PHP 代码的优化。PHP Accelerator 可以从其官方网站 http://turck-mmcache.sourceforge.net/上下载到，目前最新的版本是 2.4.6。从其官方网站的统计表明，该加速器的最新版本已经在性能优化方面比 Zend 官方版的加速器略胜一筹。Turck MMCache 支持多种操作系统，包括 Windows。

13.3.1 Turck MMCache 的安装

本节将以 Windows 下的 Turck MMCache 安装为例说明 Turck MMCache 的安装步骤。

（1）复制 mmcache.dll 到一个特定的目录，例如 C:\PHP\ext。

（2）修改 PHP 安装目录下的 php.ini 文件并增加以下内容

```
zend_extension_ts="path\mmcache.dll"
mmcache.shm_size="16"
mmcache.cache_dir="C:\tmp\mmcache "
mmcache.enable="1"
mmcache.optimizer="1"
mmcache.check_mtime="1"
mmcache.debug="0"
mmcache.filter=""
```

这里，path\是 Turck MMCache 的所在目录。

（3）创建 c:\tmp\mmcache 文件夹用于储存优化时的缓冲数据。

（4）重新启动 Apache 服务器。

这样，Turck MMCache 的安装就完成了。在上面代码的下半部是 Turck MMCache 的配置信息，具体含义将在下一节中介绍。

13.3.2　Turck MMCache 的配置

本节将介绍前面在 php.ini 文件中添加的配置信息的含义与配置方法，读者可以根据本节的介绍对前面的参数进行配置。

- mmcache.shm_size：该参数表示 Turck MMCache 的共享内存数，单位为 MB。设置为“0”表示使用操作系统的默认值。
- mmcache.cache_dir：该参数表示用于存放缓存数据的文件夹，默认值为“/tmp/mmcache”。
- mmcache.enable：该参数用于启动和关闭 Turck MMCache。设置为“1”表示启动 Turck MMCache，“0”表示关闭 Turck MMCache。
- mmcache.optimizer：该参数用于是否使用 Turck MMCache 的内部优化器。设置为“1”表示使用，“0”表示不使用。默认值为“1”。
- mmcache.debug：该参数用于是否使用记录调试信息。设置为“1”表示记录，“0”表示不记录。默认值为“0”。
- mmcache.check_mtime：该参数用于声明是否检查 PHP 代码已经被修改。设置为“1”表示检查，“0”表示不检查。如果需要在 PHP 代码修改后重新编译，则应该将其设为“0”。该参数为“1”时，修改 PHP 代码并不影响 PHP 代码的运行效果。默认值为“1”。
- mmcache.filter：该参数用于声明什么样的 PHP 文件将被 Turck MMCache 缓存。例如，“*.php”表示所有以.php 结尾的文件都将被缓存处理。使用“!”表示什么样的文件将不被缓存处理。例如，“!*.php”表示所有以.php 结尾的文件都将不被缓存处理。默认值为空值，表示所有的文件都将被缓存处理。

13.4　eAccelerator

eAccelerator 是另外一款源代码开放的 PHP 加速器，eAccelerator 是 Turck MMCache 的一个分支。很多 eAccelerator 的代码都是在 Turck MMCache 的基础上开发的。eAccelerator 可以从其官方网站 http://eaccelerator.net/上下载到，目前的最新版本是 0.9.5。

eAccelerator 的安装步骤与 Turck MMCache 很相似。

（1）复制 mmcache.dll 到一个特定的目录，例如 C:\PHP\ext。

（2）修改 PHP 安装目录下的 php.ini 文件并增加以下内容

```
[eAccelerator]
;extension=path\eaccelerator.dll
;eaccelerator.shm_size = "0"
;eaccelerator.cache_dir = "C:\tmp"
;eaccelerator.enable = "1"
;eaccelerator.optimizer = "0"
;eaccelerator.debug = "0"
;eaccelerator.check_mtime = "1"
;eaccelerator.filter = ""
```

这里，path\是 eAccelerator 的所在目录。

（3）创建 c:\tmp\文件夹用于储存优化时的缓冲数据。

（4）重新启动 Apache 服务器。

这样，eAccelerator 的安装就完成了。可以看到，在 php.ini 中增加的内容与 Turck MMCache 几乎完全相同。读者可以参照 13.3.2 节对 eAccelerator 进行配置。

13.5　小结

本章介绍了目前流行的几款 PHP 加速器，在服务器上安装一款合适的 PHP 加速器可以使 PHP 的运行效率进行优化。读者在实际开发的过程中可以选择一款加速器来优化 PHP 代码的运行。需要注意的是虽然 PHP 加速器可以有效的提高 PHP 的运行效率，但是在一台服务器上不要安装过多 PHP 加速器。只需要安装一两款即可，安装过多的加速器反而会对服务器的性能造成一定影响。