10 件在 PHP 7 中不要做的事情

[来源](https://segmentfault.com/a/1190000006912969)

原文：[10 Things Not To Do In PHP 7](https://kinsta.com/blog/10-things-not-to-do-in-php-7/)

译者：[飞龙](https://github.com/wizardforcel)

协议：[CC BY-NC-SA 4.0](http://creativecommons.org/licenses/by-nc-sa/4.0/)

## 1. 不要使用mysql\_函数

这一天终于来了，从此你不仅仅“不应该”使用mysql\_函数。PHP 7 已经把它们从核心中全部移除了，也就是说你需要迁移到好得多的mysqli\_函数，或者更灵活的 PDO 实现。

## 2. 不要编写垃圾代码

这一条可能易于理解，但是会变得越来越重要，因为 PHP 7 的速度提升可能会隐藏你的一些问题。不要仅仅满足于你的站点速度，因为迁移到 PHP 7 才让它变快。

为了理解速度有多重要，以及如何把事情做得更好，请看一看我们的文章[速度优化入门指南](https://kinsta.com/learn/page-speed/)。

作为一名开发者，你应该总是确保按需加载脚本，尽可能连接它们，编写高效的数据库查询，尽可能使用缓存，以及其它。

## 3. 不要在文件末尾使用 PHP 闭合标签

你可以看一看，当一个文件以 PHP 代码结尾时，WordPress 多数核心代码都把末尾的 PHP 标签去掉了。实际上，Zend 框架特别[禁止了它](https://framework.zend.com/manual/1.12/en/coding-standard.php-file-formatting.html)。PHP 并不需要文件末尾的闭合标签，并且我们可以通过去掉它来保证不会在后面添加任何的空白字符。

## 4. 不要做不必要的引用传递

我个人不喜欢引用传递。我知道有时候它很实用，但是其它情况下它使代码变得难懂，并且更难预测结果。

据说一些人认为它使代码运行更快，但是根据[一些 PHP 高级程序员](http://schlueters.de/blog/archives/125-Do-not-use-PHP-references.html)所说，这并不正确。

说明引用为什么不好的一个例子是，PHP 内建了shuffle()和sort()。它们修改原始数组，而不是返回处理后的数组，这很不合逻辑。

## 5. 不要在循环中执行查询

在循环中执行查询非常浪费。它给你的系统施加不必要的压力，并且可能能够在循环外部更快获得相同结果。当我遇到需要这样的情况时，我通常会使用两个分离的查询来解决问题，我会使用它们来构建数据数组。之后我会遍历数组，并不需要在这个过程中执行查询。

由于 WordPress 适用于这里，它可能有一些例外。虽然get\_post\_meta() 会从数据库获取大量数据，如果你正在遍历某个特殊博文的元数据你可以在循环中使用它。这是因为当你第一次调用它的时候，WordPress实际上会获取所有元数据并缓存它们。后续的调用使用这些缓存数据，没有数据库的调用。

弄懂这些的最佳方式是阅读函数文档，以及使用类似 [Query Monitor](https://wordpress.org/plugins/query-monitor/) 的工具。

## 6. 不要在 SQL 查询中使用\*

当然，这个更像 MySQL 的问题，但是我们习惯在 PHP 中编写 SQL 代码，所以都差不多。无论如何，如果可以避免的话，不要在 SQL 查询里使用通配符，尤其是数据库有很多列的时候。

你应该明确指定需要哪些行，并且仅仅获取它们。这有助于减少所用资源，保护数据，以及让事情变得尽可能清晰。

对于 SQL，你需要了解所有可用的函数，并且尽可能测试其速度。在计算均值、求和或计算类似数值时，要使用 SQL 函数而不是 PHP 函数。如果你不确定某个查询的速度，测试它并且尝试一些其它的编译 -- 之后使用最好的那个。

## 7. 不要信任用户输入

信任用户输入是不明智的。始终校验、过滤、转义、检查并留好退路。用户数据存在三个问题：我们开发者并没有考虑每种可能性，它通常不正确，以及它可能是蓄意破坏。

经过周密考虑的系统可以防护这些威胁。要确保使用类似filter\_var()的内建函数检查适当的值，以及在处理数据库时转义（或预编译）。

WordPress 拥有一些函数来解决问题。详见文章[校验、转义和过滤用户数据](https://codex.wordpress.org/Validating_Sanitizing_and_Escaping_User_Data)。

## 8. 不要故作聪明

你的目标应该是编写优雅的代码，来更清晰地表达你的意图。你可能能够通过将任何东西缩短为一个单词的变量，使用多层的三元逻辑，以及其它手段，从每个页面中优化 0.01 秒。但这只会给你和你周围的人产生大麻烦。

合理命名变量，为代码编写文档，优先选择清晰而不是简洁。甚至还可以更好，使用标准的面向对象代码，它本身或多或少就是文档，不需要一大堆内联数值。

## 9. 不要重新发明轮子

PHP 到现在为止有很长时间了，网站被造出来的时间更长。很可能无论你需要造出什么，一些人之前早就造出来了。不要害怕向他人寻求支持，[Github](http://github.com/)是你的好朋友，[Composer](https://getcomposer.org/)也是，[Packagist](https://packagist.org/)也是。

从日志工具到调色工具，从性能分析器到单元测试框架，从 Mailchimp API 到 Twitter Bootstrap，每个东西都可以通过按下按键（或者敲下命令）来获取，使用它们吧！

## 10. 不要忽略其它语言

如果你是个 PHP 程序员，现在有个好机会去至少了解 HTML、CSS、JavaScript 和 MySQL。当你能够更好地处理这些语言时，就是重新学习 JavaScript 的时机了。JavaScript 并不是 jQuery，你应该合理地学习 JavaScript 来更高效地使用它。

我也打算向你推荐学习面向对象的 PHP，它可以节省时间，并且在代码规模更大时会变得更好。对于类似 C# 和 Java 的语言，在你了解 OOP 之后，它们也更易于理解。

通过了解包管理器、构建脚本、CoffeeScript、LESS、SASS、YAML 、脚本引擎和其它强大的工具来扩展你的知识面。我强烈向你推荐看一看其它框架，尤其是 [Laravel](http://laravel.com/)。

当你使用它们出色完成任务时，学习 Ruby、RoR、Android、iPhone 和 Windows Phone 应用开发如何？你可能会认为这毫无意义，因为它们在你的舒适区和工作所需范围之外，但是这就是它们的意义。每种语言都有一些要学习的实用的东西，以及从没碰到的新知识。所有 PHP 顶级开发者都懂得很多其它编程语言，这并非偶然。