

PHP Tips - Optimize How To

Saturday, 24 May 2008

Last Updated Monday, 14 July 2008

Sebagai web programmer tentu saja anda ingin membuat web site yang cepat di load dan tidak membebani kinerja server.

Berikut adalah tips untuk mengoptimalkan kode-kode php anda sehingga kode php anda dapat lebih cepat di load dan di eksekusi oleh server dan tidak membebani atau menggunakan terlalu banyak resource yang disediakan oleh server.

1. Penggunaan Variable

Penggunaan Variable

- Mengoperasikan variable yang sudah di set lebih cepat 376% dibandingkan menggunakan variable yang belum di set.
- Menggunakan constant lebih lambat 146% di banding menggunakan variable.
- Menggunakan lokal variable 9.9% lebih cepat di banding menggunakan global variable.

Fungsi String

- Menggunakan 'string' lebih cepat 0.26% di banding menggunakan "string"
- Menggunakan "string" lebih cepat 4% di banding menggunakan sintak HEREDOC
- Menggunakan "string\n" lebih cepat 108% di banding menggunakan 'string'. "\n"
- Menggunakan 'string'.\$var lebih cepat 28% di banding menggunakan "string\$var"
- Menggunakan 'string '\$var.' string' lebih cepat 55% di banding menggunakan sprintf('string %s string', \$var)
- Menggunakan "\n" lebih cepat 70% di banding menggunakan chr(10)
- Menggunakan strnatcmp() lebih cepat 4.95% di banding menggunakan strcmp()
- Menggunakan strcasecmp() lebih cepat 45% di banding menggunakan preg_match()
- Menggunakan strcasecmp() lebih cepat 6.6% di banding menggunakan strtoupper(\$string) == "STRING"

- Menggunakan `strcasecmp()` lebih cepat 13% di banding menggunakan `strnatcasecmp()`
- Menggunakan `strtr($string, $string1, $string2)` lebih cepat 10% di banding menggunakan `str_replace()`
- Menggunakan `str_replace()` lebih cepat 161% di banding menggunakan `strtr($string, $array)`
- Menggunakan `stristr()` lebih cepat 10% di banding menggunakan `stripos()`
- Menggunakan `strpos()` lebih cepat 9.7% di banding menggunakan `stristr()`
- Menggunakan `isset($str{5})` lebih cepat 176% di banding menggunakan `strlen($str) > 5`
- Menggunakan `str_replace($str, $str, $str)` 2x lebih cepat 17% di banding menggunakan `str_replace(array, array, string)`
- Menggunakan `list() = explode()` lebih cepat 13% di banding menggunakan `substr($str, strpos($str))`

Fungsi Numeric

- Menggunakan `++$int` lebih cepat 10% di banding menggunakan `$int++`
- Menggunakan `(float)` lebih cepat 48% di banding menggunakan `settype($var, 'float')`

Fungsi Array

- Menggunakan `list() = $array;` lebih cepat 3.4% di banding menggunakan assigning each variable
- Menggunakan `in_array()` lebih cepat 6% di banding menggunakan `array_search`
- Menggunakan `isset($array[$key])` lebih cepat 230% di banding menggunakan `array_key_exists()`
- Menggunakan `!empty($array)` lebih cepat 66% di banding menggunakan `count($array)`

Fungsi Output

- Menggunakan `echo` lebih cepat 5% di banding menggunakan `print()`
- Menggunakan `echo ''''` lebih cepat 0.44% di banding menggunakan `echo ''''`

2. Function dan Method

Function dan Calling Method

- Menggunakan `call_user_func()` lebih lambat 54% di banding langsung memanggil function tersebut
- Menggunakan `call_user_func()` lebih lambat 59% di banding langsung memanggil sebuah static method
- Menggunakan `call_user_func()` lebih lambat 65% di banding langsung memanggil sebuah object method
- Menggunakan `function()` lebih cepat 119% di banding menggunakan `static::method()`
- Menggunakan `$this->method()` lebih cepat 116% di banding menggunakan `static::method()`
- Menggunakan `declared static::method()` lebih cepat 93% di banding menggunakan `static::method()`

Fungsi Umum

- Menggunakan Pass by reference lebih cepat 3% di banding menggunakan Return by reference
- Menggunakan No reference lebih cepat 1.7% di banding menggunakan Return by reference

3. Storage (Penyimpanan)

File System

- Menggunakan `Scandir()` lebih cepat 4% di banding menggunakan `opendir()`, `readdir()`, `closedir()`
- Menggunakan `file_get_contents()` lebih cepat 52% di banding menggunakan `fopen()`, `fread()`, `fclose()`
- Menggunakan `file_get_contents()` lebih cepat 39% di banding menggunakan `implode("\n", file())`

Fungsi Cache

- Menggunakan `xcache_set()` lebih cepat 1,645% di banding menggunakan `file_put_contents()`
- Menggunakan `xcache_set()` lebih cepat 646% di banding menggunakan `memcache->set()`
- Menggunakan `xcache_get()` lebih cepat 1,312% di banding menggunakan `memcache->get()`

4. Fungsi-Fungsi Lainnya

Fungsi Umum (General)

- Menggunakan `if elseif else` lebih cepat 0.78 % di banding menggunakan `switch`
- Menggunakan `@Error supression` lebih lambat 235% di banding tidak menggunakannya
- Menggunakan `$_SERVER['REQUEST_TIME']` lebih cepat 59% di banding menggunakan `time()`
- Menggunakan `min(array)` lebih cepat 16% di banding menggunakan `min(int, int)`
- Menggunakan `require_once()` lebih cepat 24% di banding menggunakan `include()`
- Menggunakan `require_once()` sama cepat dengan menggunakan perintah `include_once()`
- Menggunakan `include(relative path)` lebih cepat 37% di banding menggunakan `include(full path)`

- Regular Expressions

- Menggunakan `str_replace()` lebih cepat 40% di banding menggunakan `preg_replace()`
- Menggunakan `ereg('regex')` lebih cepat 17% di banding menggunakan `preg_match('/regex/')`
- Menggunakan `preg_match('/regex/i')` lebih cepat 68% di banding menggunakan `eregi('regex')`