

Dicas de Desenvolvimento

Trabalhando com threads no PHP

Introdução

PHP

O PHP é uma linguagem de script open source de uso geral muito utilizada. Especialmente adequada para o desenvolvimento web, e que pode ser incluída dentro do HTML.

A imagem mostra um exemplo do funcionamento de uma aplicação desenvolvida em PHP.



(<https://dicasdedevelop.files.wordpress.com/2015/06/php.jpg>).

Threads

Disponível em várias linguagens de programação, a thread é uma ferramenta que executa blocos de instruções em paralelo, aumentando o desempenho do software.

Objetivo

Criar um aplicativo em PHP para testar a performance do processamento multi threads.

Desenvolvimento

Para usar threads no PHP é necessário instalar uma extensão, nesse caso chamada de pthreads.


PTHREADS

POSIX threads é um padrão POSIX para threads, o qual define uma API padrão para criar e manipular thread. As bibliotecas que implementam a POSIX threads são chamadas de Pthreads, são muito difundidas no universo Unix e outros sistemas operacionais semelhantes, como Linux e Solaris.

Instalando a API no Windows

Considerando que o PHP já está instalado e funcional no servidor, use o comando `phpinfo()`; para verificar se a versão do PHP presente é compatível para se trabalhar com “Threads”.

PHP Extension Build	API20131226,TS,VC11
Debug Build	no
Thread Safety	enabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled



(<https://dicasdedevelop.files.wordpress.com/2015/06/table.jpg>).

Caso a opção “Thread Safety” estiver “disabled”, efetuar o download da versão que permita threads em:

<http://windows.php.net/download/>
(<http://windows.php.net/download/>).

As versões são:

“VC11 x86 Thread Safe” ou “VC11 x64 Thread Safe”, escolher a versão mais adequada para a arquitetura presente.

Efetuar o download da API binária no link:

<http://windows.php.net/downloads/pecl/releases/pthreads/>
(<http://windows.php.net/downloads/pecl/releases/pthreads/>).

Escolher o arquivo a baixar de acordo com a versão do PHP.

Extrair o arquivo zip e efetuar as cópias:

1. Copiar `php_pthreads.dll` para o diretório “bin\php\ext\”;
2. Copiar `pthreadVC2.dll` para o diretório “bin\php\”;
3. Copiar `pthreadVC2.dll` para o diretório “bin\apache\bin”;
4. Copiar `pthreadVC2.dll` para o diretório “C:\windows\system32”;
5. Incluir a linha “extension=php_pthreads.dll” no arquivo `php.ini`;

6. Reiniciar o servidor

Finalizado o procedimento, o servidor estará pronto para operar “Threads” em PHP.

No exemplo a seguir será usada a classe *Threads*, trata-se de uma classe abstrata e deve ser herdada por classes filhas para ser utilizada corretamente. Seu método principal é denominado *run*, o qual será executado em paralelo.

Foram criados dois arquivos para o teste, um arquivo HTML (index.html) para a seleção dos parâmetros e um arquivo PHP (processo.php) que recebe os parâmetros e realiza um processo simples, que contempla duas equação matemáticas (elevant um número ao quadrado e depois calcular a raiz quadrada do resultado), registra o tempo gasto, depois realiza o mesmo processamento dividido em várias threads e por fim mostra a diferença de performe do processamento.

Abaixo o código do index.html

```

1  <html>
2  <body>
3  <form action="processo.php" target="result" method="post">
4  <table border="1" width="90%" align="center">
5  <tr>
6  <td width="70%">
7  <font size=6><b>Utilizando pthread no PHP</b></font>
8  <font size=5>Processador i7 (QuadCore - 8 nucleos)</font>
9  <font size=4>Tamanho: <select id="tamanho" name="tamanho">
10 <option value=1>1 unidade</option>
11 <option value=10>10 unidades</option>
12 <option value=100>100 unidades</option>
13 <option value=1000>1000 unidades</option>
14 </select> (1000x)<br><br>
15
16 <font size=4>Exemplo: <select id="exemplo" name="exemplo">
17 <option value=2>2 threads</option>
18 <option value=4>4 threads</option>
19 <option value=5>5 threads</option>
20 <option value=8>8 threads</option>
21 <option value=20>20 threads</option>
22 </select><br><br>
23
24 <input type="submit" value="Processar"><br><br>
25 </td>
26 <td width="30%">&nbsp;</td>
27 </tr>
28 <tr>
29 <td width="70%">
30 <font size=5>Resultado</font><br>
31 <iframe name="result" src="processo.php" width="100%" height="100px">
32 </td>
33 <td width="30%">&nbsp;</td>
34 </tr>
35 </table>
36 </form>
37 </body>
38 </html>

```

Abaixo o código do processo.php

```

1  <?php
2  set_time_limit(0);
3  date_default_timezone_set("Brazil/East");
4
5  // Iniciamos o "contador"
6  list($usec3, $sec3) = explode(' ', microtime());
7  $script_start3 = (float) $sec3 + (float) $usec3;
8
9  if (!empty($_POST["tamanho"]))
10 {
11
12     $tamanho = $_POST["tamanho"] * 1000000;
13     $exemplo = $_POST["exemplo"];
14     $vlr = $tamanho / $exemplo;
15
16     // Iniciamos o "contador"
17     list($usec, $sec) = explode(' ', microtime());
18     $script_start = (float) $sec + (float) $usec;
19     $soma = 0;
20     for ($x = 1; $x <= $tamanho; $x++)
21     {
22         $soma=sqrt(pow($x,2));
23     }
24     $soma = 0;
25     // Terminamos o "contador" e exibimos
26     list($usec, $sec) = explode(' ', microtime());
27     $script_end = (float) $sec + (float) $usec;
28     $elapsed_time = round($script_end - $script_start);
29
30     // Exibimos uma mensagem
31     echo 'Processo Simples: ', $elapsed_time, ' segundos';
32
33     class AguardaRand extends Thread {
34
35         // ID da thread (usado para identificar a ordem que
36         protected $id, $vlr;
37
38         // Construtor que apenas atribui um ID para identi
39         public function __construct($id, $vlr) {
40             $this->id = $id;
41             $this->vlr = $vlr;
42         }
43
44         // Metodo principal da thread, que sera acionado c
45         public function run() {
46             $soma = 0;
47             for ($x = 1; $x <= $this->vlr; $x++)
48             {
49                 $soma=sqrt(pow($x,2));
50             }
51             $soma = 0;
52         }
53     }
54
55     // Iniciamos o "contador"
56     list($usec2, $sec2) = explode(' ', microtime());
57     $script_start2 = (float) $sec2 + (float) $usec2;

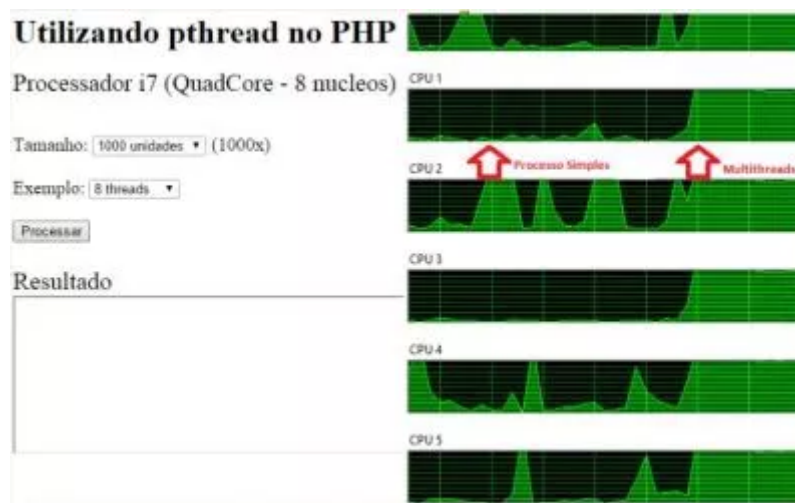
```

```

58
59 // Criar um vetor com 10 threads do mesmo tipo
60 $vetor = array();
61 for ($id = 0; $id < $exemplo; $id++) {
62     $vetor[] = new AguardaRand($id,$vlr);
63 }
64
65 // Iniciar a execucao das threads
66 foreach ($vetor as $thread) {
67     $thread->start();
68 }
69
70 $thread->join();
71
72 // Terminamos o "contador" e exibimos
73 list($usec2, $sec2) = explode(' ', microtime());
74 $script_end2 = (float) $sec2 + (float) $usec2;
75 $elapsed_time2 = round($script_end2 - $script_start, 2);
76 echo 'Multithreads: ', $elapsed_time2, ' segundo(s)<br>';
77
78 }
79
80 // Terminamos o "contador" e exibimos
81 list($usec3, $sec3) = explode(' ', microtime());
82 $script_end3 = (float) $sec3 + (float) $usec3;
83 $elapsed_time3 = round($script_end3 - $script_start, 2);
84 echo 'Tempo Total: ', $elapsed_time3, ' segundo(s)<br>';
85
86 ?>

```

Abaixo imagem do software em execução:



(<https://dicasdedevelop.files.wordpress.com/2015/06/tm.jpg>).

Resultado:

Utilizando pthread no PHP

Processador i7 (QuadCore - 8 nucleos)

Tamanho: (1000x)

Exemplo:

Resultado

Processo Simples:	41.89233 segundo(s)	- 100 unidades
Multithreads:	10.12006 segundo(s)	- 100(8 X 12.5) unidades (-75.8%)
Tempo Total:	52.0125 segundo(s)	

(<https://dicasdedevelop.files.wordpress.com/2015/06/pthread.jpg>).

Conclusão

A utilização da extensão pthreads no PHP aumenta consideravelmente o desempenho do software, no exemplo mostrado houve redução do tempo de execução em cerca de 75%.

Antônio Puças Jr

Rafael Ferreira

☐ JUNHO 11, 2015 ☐ RASFER ☐ TUTORIAL

CRIE UM WEBSITE OU BLOG GRATUITO NO WORDPRESS.COM.