

ハイパフォーマンスWordPress

スライドURL <http://goo.gl/rRwrvj>



プライム・ストラテジー株式会社

代表取締役 中村 けん牛

1. 今日お話しすること

「WordPressとサーバをチューニングして
どこまで高速化できるのか？」

をテーマに高速化の技術と考え方をお話しします。

「まずは証拠」

をお見せします。

お手数ですが、

「KUSANAGI WordPress」と検索して

<http://kusanagi.tokyo/>

にアクセスして見ていただけませんか？

Run time 0.005 s.



超高速 WordPress 仮想マシン KUSANAGI

Powered by  Prime Strategy



Run time 0.005 s.

KUSANAGI

超高速WORDPRESS仮想マシン「KUSANAGI」について

KUSANAGIがご利用いただけるパブリッククラウド

超高速WordPress仮想
マシン

KUSANAGI FOR MICROSOFT AZURE がご利用いただけるよう になりました

© 2015年7月2日

KUSANAGI for Microsoft Azureがご利用いただけるようになりました。

ご利用方法は[こちら](#)でご案内しています。

2. 自己紹介

中村 けん牛 自己紹介



WordPressのフルマネージドサービスを提供するプライム・ストラテジー株式会社の代表をしています。

おもに東京とジャカルタで働いています。



@kengyu_n



Kengyu.Nakamura

月間1億PV超のメディアサイトなどの構築、サーバ運用

マイナビ様
「マイナビウーマン」

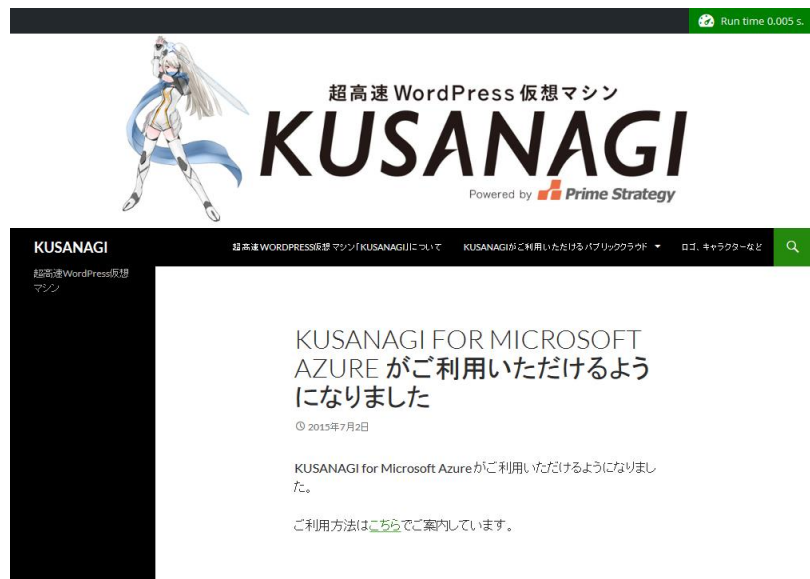
テレビ朝日様
番組ブログポータル



Adobe Systems 様
事例サイト



超高速WordPress仮想マシン「KUSANAGI」の開発

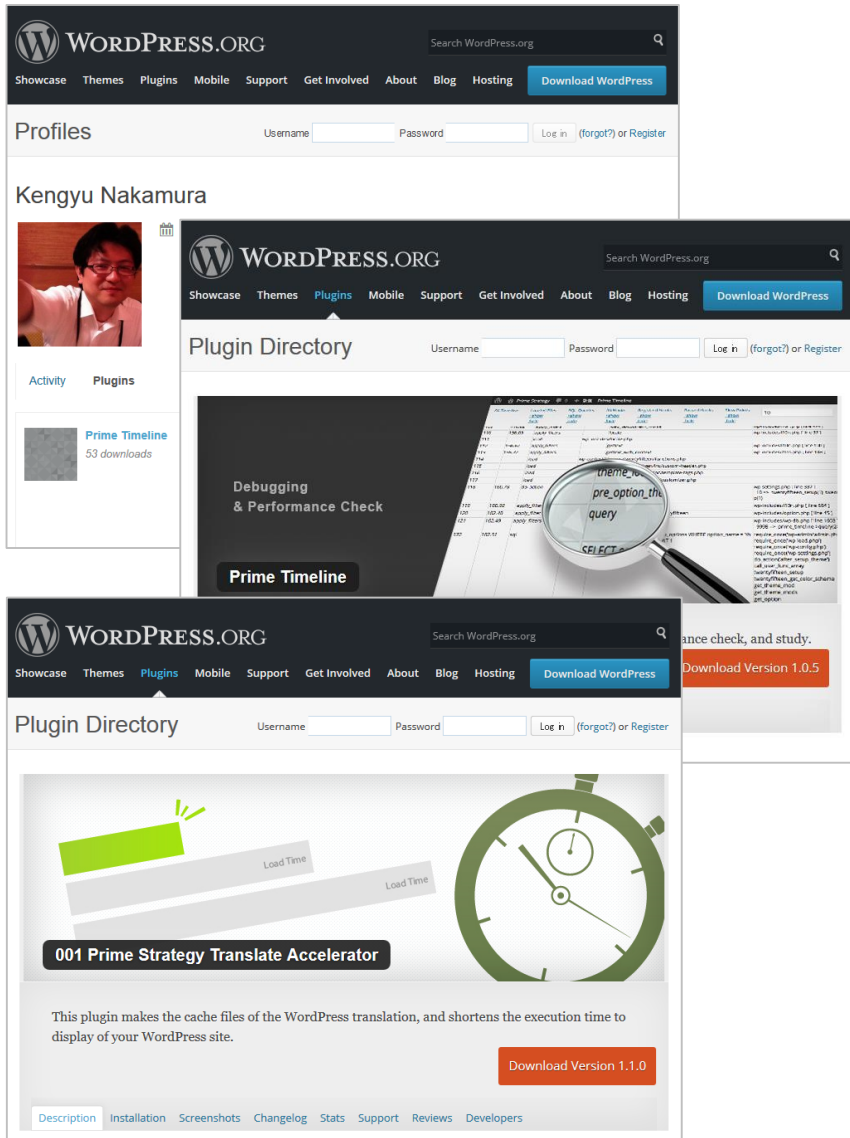


・WordPress 実行時間3 ミリ秒
台

・1000 リクエスト／秒

をページキャッシュ非使用で実
現する仮想マシン
(4vCPU、最大性能時)

WordPressプラグインの開発



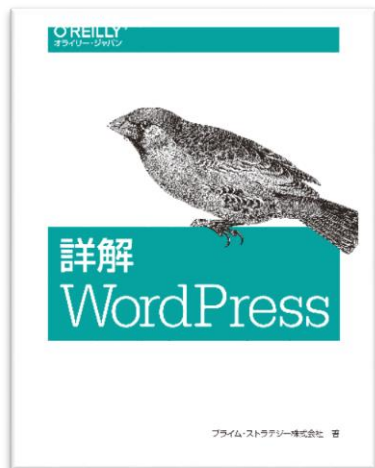
▪ Prime Timeline

=> ランタイムプロファイラ

▪ 001 Prime Strategy Translate Accelerator

=> 翻訳アクセラレータ

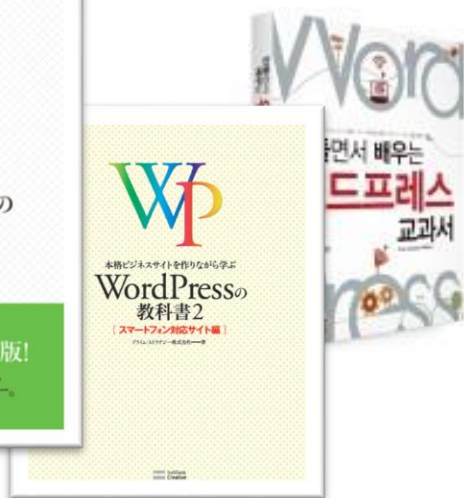
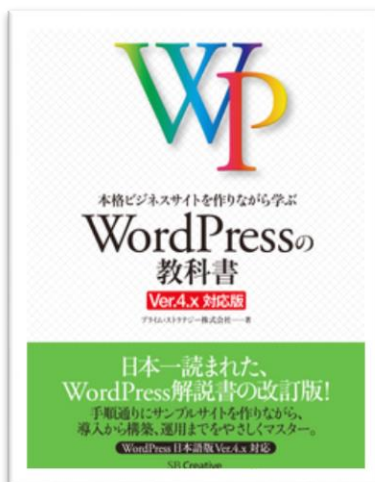
WordPress関連書籍の執筆など



『詳解 WordPress』

『WordPressによるWebアプリケーション開発』

(出版社:株式会社オライリー・ジャパン)



『WordPressの教科書』 シリーズ

(出版社:SBクリエイティブ株式会社他)

3. WordPressの高速化

なぜWordPressの高速化が必要とされるのか？

1. WordPressはPHP+MySQLの動的なシステム

=>

静的なHTMLページに比べて動作速度の点で不利

なぜWordPressの高速化が必要とされるのか？

2. CPUの開発ロードマップは動作クロック(周波数)よりもコア数重視の流れ

=>

ハードの進化による処理速度向上を期待しづらい

なぜWordPressの高速化が必要とされるのか？

3. このような背景の中でオウンドメディアやサービスサイトでは

- (1) PV獲得の機会を失わないという観点
- (2) ユーザーエクスペリエンス向上の観点
- (3) 検索エンジン最適化の観点
- (4) Webサイトの信頼性、安定性の観点

WordPressの高速化とは？

1. サーバサイドでの高速化 <= 本日のテーマ
(サーバ、WordPress)

2. フロントエンドの高速化
(HTML、CSS、JavaScript)

サーバサイドの高速化とは？

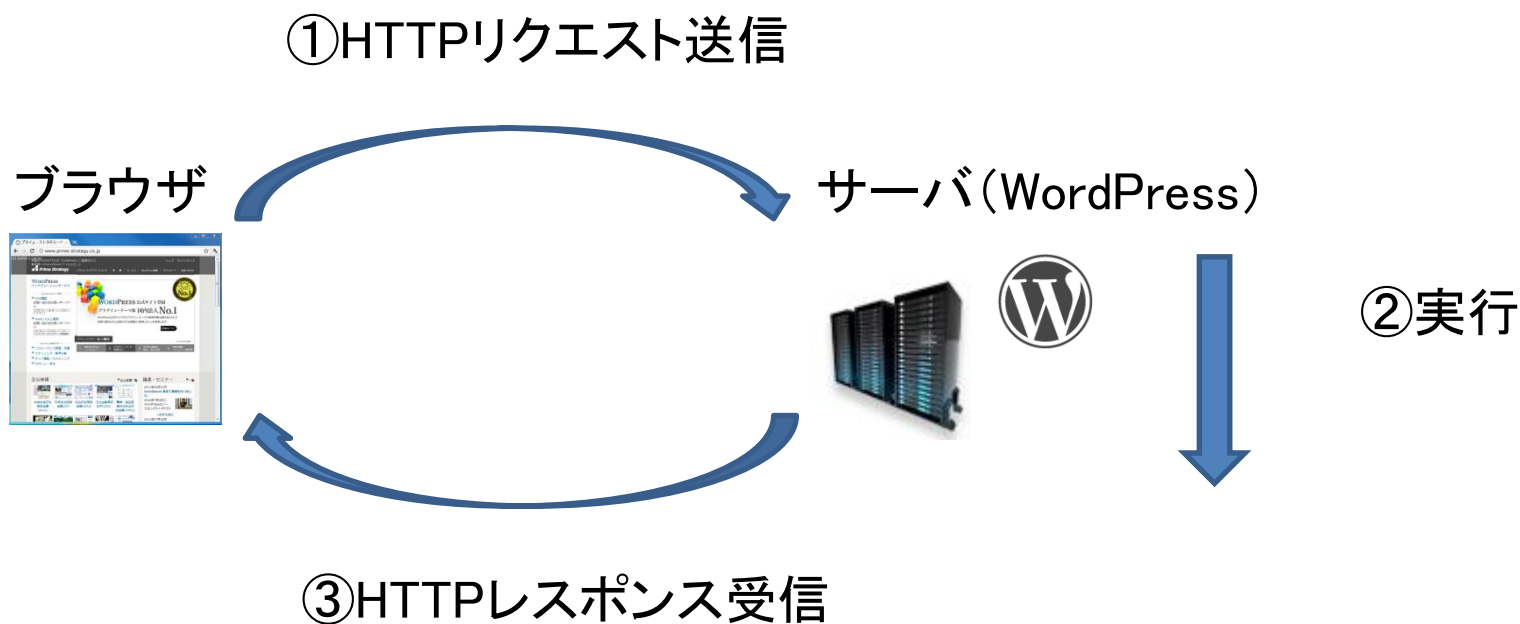
HTMLページのロード時間を短くして

サーバサイドの高速化とは？

HTMLページのロード時間を短くして
1秒あたりのリクエスト数を増やすこと

WordPressの高速化

HTMLページのロード時間を短くする = ① + ② + ③を短縮する



WordPressの高速化

HTMLページのロード時間をFirebugのネットタブで確認する

The screenshot shows a web browser displaying the KUSANAGI website. The page title is "KUSANAGI | 超高速WordPr...". The URL is "kusanagi.tokyo". The page content includes a character illustration and the text "超高速 WordPress 仮想マシン KUSANAGI Powered by Prime Strategy".

The Firebug network tab is open, showing a table of network requests. The first request is a GET request to "kusanagi.tc" with a status of "200 OK", a size of "4.3 KB", and a response time of "31ms". This response time is circled in orange. A blue arrow points from the text "ページのロード時間" to this circled value.

URL	ステータス	ドメイン	サイズ	リモート IP	タイムライン
GET kusanagi.tc	200 OK	kusanagi.tokyo	4.3 KB	59 106.68.240:80	31ms
1 件のリクエスト			4.3 KB		21ms (onload: 610ms)

ページのロード時間

WordPressの高速化

1秒あたりのアクセス数を増やすとは？

ココ

The screenshot shows the Yahoo! JAPAN homepage. A large blue arrow points from the top right towards the news section. The news section is circled in orange and contains the following items:

- 12時21分更新
- 台風12号沖縄直撃へ大雨警戒
- 中2自殺 担任「予期できず」
- シャープ 再建の柱もう廃止
- 車検離機・飛燕 部品相次ぎ発見
- 車速高操作 140万台リコール **NEW!**
- イチロー フェンス際の美技 **NEW!**
- 運球きブレイク まれ子役魅力
- ひな壇ジェンマ?紫吹淳のズレ **NEW!**

Below the news section is a weather forecast for Osaka (大阪) for July 25th (土):

2015年7月25日(土)	
今日の天気 (大阪)	明日の天気 (大阪)
☀️ 10% 34°C / 25°C	☀️ 10% 34°C / 25°C
熱中症指数	熱中症指数
雨雲レーダー	箕面市

WordPressの高速化

ページのロード時間と1秒あたりのリクエスト数の関係

クイズ1 (注: 通信時間など無視しておおざっぱに考えます)

1vCPU(1コア)でページのロード時間が500ms

だった場合、

1秒あたりのリクエスト数はいくつになりますか？

WordPressの高速化

ページのロード時間と1秒あたりのリクエスト数の関係

クイズ2 (注: 通信時間など無視しておおざっぱに考えます)

1vCPU(1コア)でページのロード時間を100msにチューニングできた場合、

1秒あたりのリクエスト数はいくつになりますか？

WordPressの高速化

ページのロード時間と1秒あたりのリクエスト数の関係

クイズ3 (注:通信時間など無視しておおざっぱに考えます)

1vCPU(1コア)でページのロード時間が100msの場合、

2vCPU(2コア)にスケールアップすると

1秒あたりのリクエスト数はいくつになりますか？

WordPressの高速化

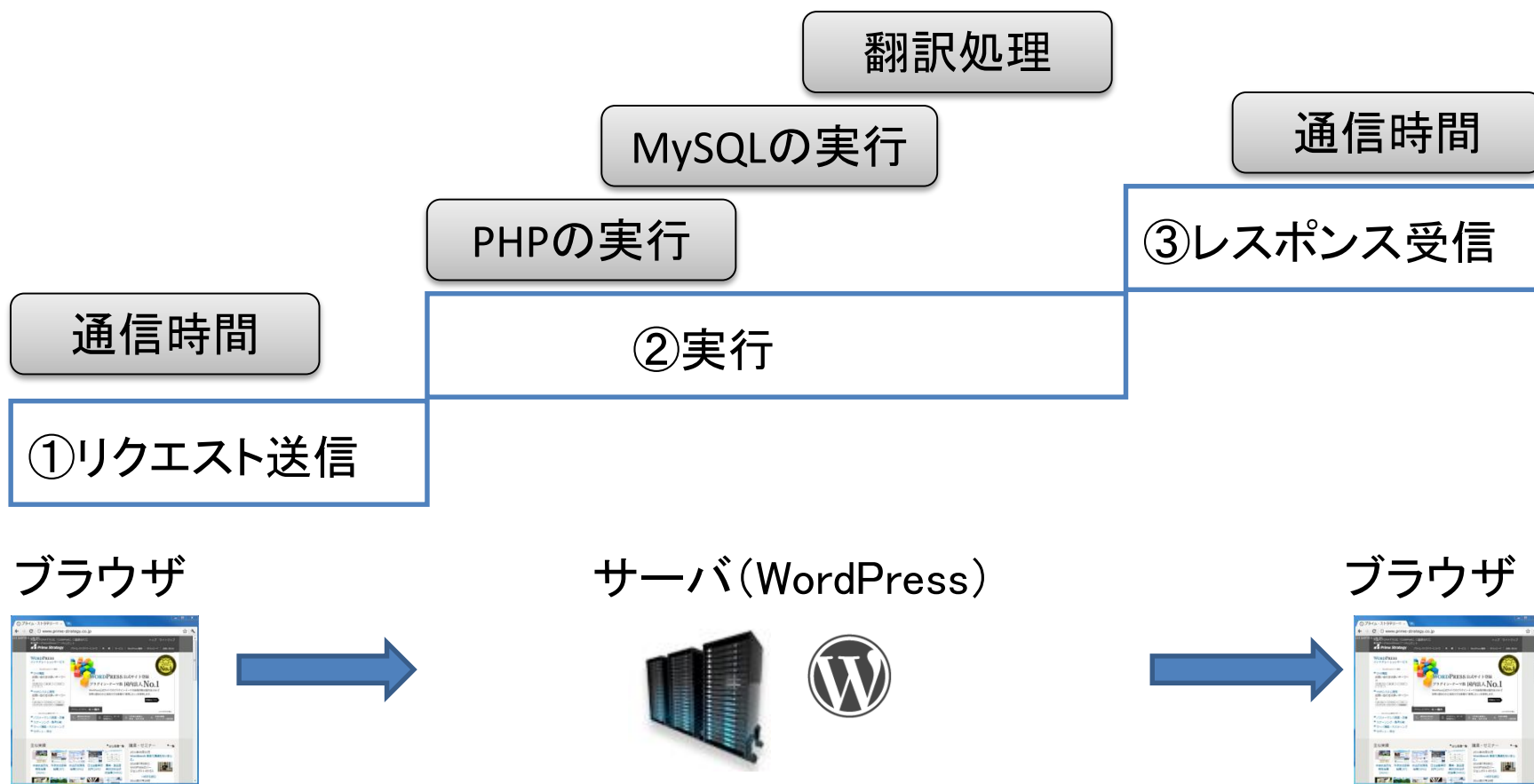
ページのロード時間と1秒あたりのリクエスト数の関係

クイズ4 (注:通信時間など無視しておおざっぱに考えます)

2vCPU(2コア)でページのロード時間が100msの場合、
4vCPU(4コア)にスケールアップするとページのロード時間はいくつになりますか？

WordPressの高速化

HTMLページのロード時間を分解すると



4. ページキャッシュを使わずに WordPressを高速化する

WordPressを高速化する

あるサーバを利用してデフォルトの状態だと



1. ロード時間	246ms
2. リクエスト数	4.9リクエスト/秒

WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



APC (PHPアクセラレータ) 導入で約1.85倍
246ms→133ms

WordPressを高速化する

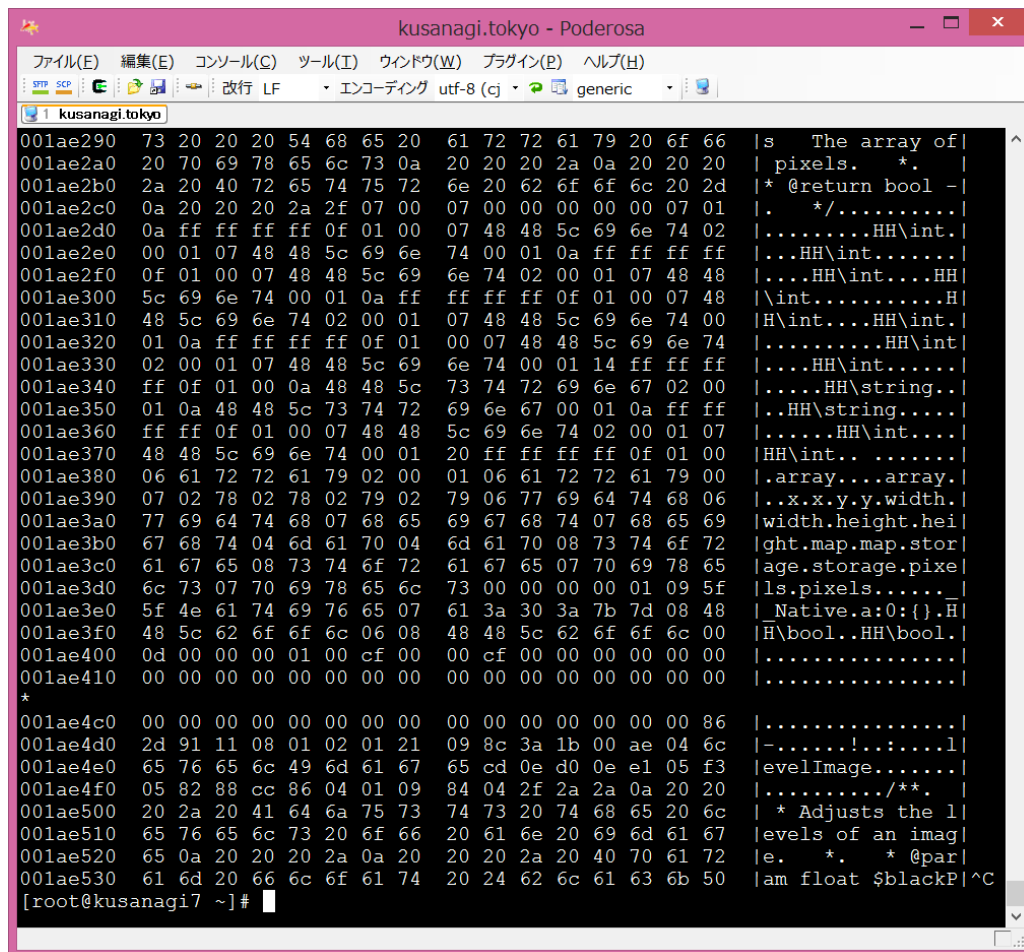
PHP実行の仕組み

```
kusanagi.tokyo - Poderosa
ファイル(E) 編集(E) コンソール(C) ツール(I) ウィンドウ(W) プラグイン(P) ヘルプ(H)
SFTP SCP 改行 LF エンコーディング utf-8 (cj) generic
1 kusanagi.tokyo
2 /**
3  * The main template file
4  *
5  * This is the most generic template file in a WordPress theme
6  * and one of the two required files for a theme (the other being sty
7  * le.css).
8  * It is used to display a page when nothing more specific matches a
9  * query.
10 * e.g., it puts together the home page when no home.php file exists.
11 *
12 * Learn more: {@link https://codex.wordpress.org/Template_Hierarchy}
13 *
14 * @package WordPress
15 * @subpackage Twenty_Fifteen
16 * @since Twenty Fifteen 1.0
17 */
18
19 get_header(); ?>
20
21 <div id="primary" class="content-area">
22   <main id="main" class="site-main" role="main">
23
24     <?php if ( have_posts() ) : ?>
25
26       <?php if ( is_home() && ! is_front_page() ) : ?>
27         <header>
28           <h1 class="page-title screen-reader-text"><?php s
29             ingle_post_title(); ?></h1>
30         </header>
31       <?php endif; ?>
32
33       <?php
34         // Start the loop.
35         while ( have_posts() ) : the_post();
36
37           10,1
38           3%
```

PHPのソースコード

WordPressを高速化する

PHP実行の仕組み



```
kusanagi.tokyo - Poderosa
ファイル(E) 編集(E) コンソール(C) ツール(I) ウィンドウ(W) プラグイン(P) ヘルプ(H)
SFTP SCP 改行 LF エンコーディング utf-8 (cj) generic
1 kusanagi.tokyo
001ae290 73 20 20 20 54 68 65 20 61 72 72 61 79 20 6f 66 |s The array of
001ae2a0 20 70 69 78 65 6c 73 0a 20 20 20 2a 0a 20 20 20 | pixels. *.
001ae2b0 2a 20 40 72 65 74 75 72 6e 20 62 6f 6f 6c 20 2d |* @return bool -|
001ae2c0 0a 20 20 20 2a 2f 07 00 07 00 00 00 00 00 07 01 |. */.....|
001ae2d0 0a ff ff ff ff 0f 01 00 07 48 48 5c 69 6e 74 02 |.....HH\int.|
001ae2e0 00 01 07 48 48 5c 69 6e 74 00 01 0a ff ff ff ff |...HH\int.....|
001ae2f0 0f 01 00 07 48 48 5c 69 6e 74 02 00 01 07 48 48 |...HH\int....HH|
001ae300 5c 69 6e 74 00 01 0a ff ff ff ff 0f 01 00 07 48 |\int.....H|
001ae310 48 5c 69 6e 74 02 00 01 07 48 48 5c 69 6e 74 00 |H\int....HH\int.|
001ae320 01 0a ff ff ff ff 0f 01 00 07 48 48 5c 69 6e 74 |.....HH\int|
001ae330 02 00 01 07 48 48 5c 69 6e 74 00 01 14 ff ff ff |...HH\int.....|
001ae340 ff 0f 01 00 0a 48 48 5c 73 74 72 69 6e 67 02 00 |....HH|string..|
001ae350 01 0a 48 48 5c 73 74 72 69 6e 67 00 01 0a ff ff |.HH|string....|
001ae360 ff ff 0f 01 00 07 48 48 5c 69 6e 74 02 00 01 07 |....HH\int....|
001ae370 48 48 5c 69 6e 74 00 01 20 ff ff ff ff 0f 01 00 |HH\int.. ..|
001ae380 06 61 72 72 61 79 02 00 01 06 61 72 72 61 79 00 |.array...array.|
001ae390 07 02 78 02 78 02 79 02 79 06 77 69 64 74 68 06 |..x.x.y.y.width.|
001ae3a0 77 69 64 74 68 07 68 65 69 67 68 74 07 68 65 69 |width.height.hei|
001ae3b0 67 68 74 04 6d 61 70 04 6d 61 70 08 73 74 6f 72 |ght.map.map.stor|
001ae3c0 61 67 65 08 73 74 6f 72 61 67 65 07 70 69 78 65 |age.storage.pixe|
001ae3d0 6c 73 07 70 69 78 65 6c 73 00 00 00 00 01 09 5f |ls.pixels.....|
001ae3e0 5f 4e 61 74 69 76 65 07 61 3a 30 3a 7b 7d 08 48 | Native.a:0:{}.H|
001ae3f0 48 5c 62 6f 6f 6c 06 08 48 48 5c 62 6f 6f 6c 00 |H\bool..HH\bool.|
001ae400 0d 00 00 00 01 00 cf 00 00 cf 00 00 00 00 00 00 |.....|
001ae410 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
*
001ae4c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 86 |.....|
001ae4d0 2d 91 11 08 01 02 01 21 09 8c 3a 1b 00 ae 04 6c |-.....!.:...l|
001ae4e0 65 76 65 6c 49 6d 61 67 65 cd 0e d0 0e e1 05 f3 |levelImage.....|
001ae4f0 05 82 88 cc 86 04 01 09 84 04 2f 2a 2a 0a 20 20 |...../**. |
001ae500 20 2a 20 41 64 6a 75 73 74 73 20 74 68 65 20 6c | * Adjusts the l|
001ae510 65 76 65 6c 73 20 6f 66 20 61 6e 20 69 6d 61 67 |levels of an imag|
001ae520 65 0a 20 20 20 2a 0a 20 20 20 2a 20 40 70 61 72 |e. *. * @par|
001ae530 61 6d 20 66 6c 6f 61 74 20 24 62 6c 61 63 6b 50 |am float $blackP| ^C
[root@kusanagi1 ~]#
```

中間コード
←これを
Zend Engine
(VM) が実行

WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



APC (PHPアクセラレータ) 導入で約1.85倍
246ms→133ms

けっこう簡単に導入できます。

たとえば、
Centos 6の場合、最短rootで
次のコマンドを打つだけ

```
yum install -y php-pecl-apc;  
service httpd restart;
```

WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



APC (PHPアクセラレータ) 導入で約1.85倍
246ms→133ms

PHP5.4まではAPC
APC=PHPアクセラレータ+
ユーザーキャッシュ

PHP5.5からはOPcache (+20%)
ユーザーキャッシュはAPCu拡張

PHP5.3、5.4はOPcacheとAPCuをPHP拡張として利用可能

WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



APC (PHPアクセラレータ) 導入で約1.85倍
246ms→133ms

MySQLのクエリキャッシュ導入で約1.15倍
133ms→115ms

WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



APC (PHPアクセラレータ) 導入で約1.85倍
246ms→133ms

MySQLのクエリキャッシュ導入で約1.15倍
133ms→115ms

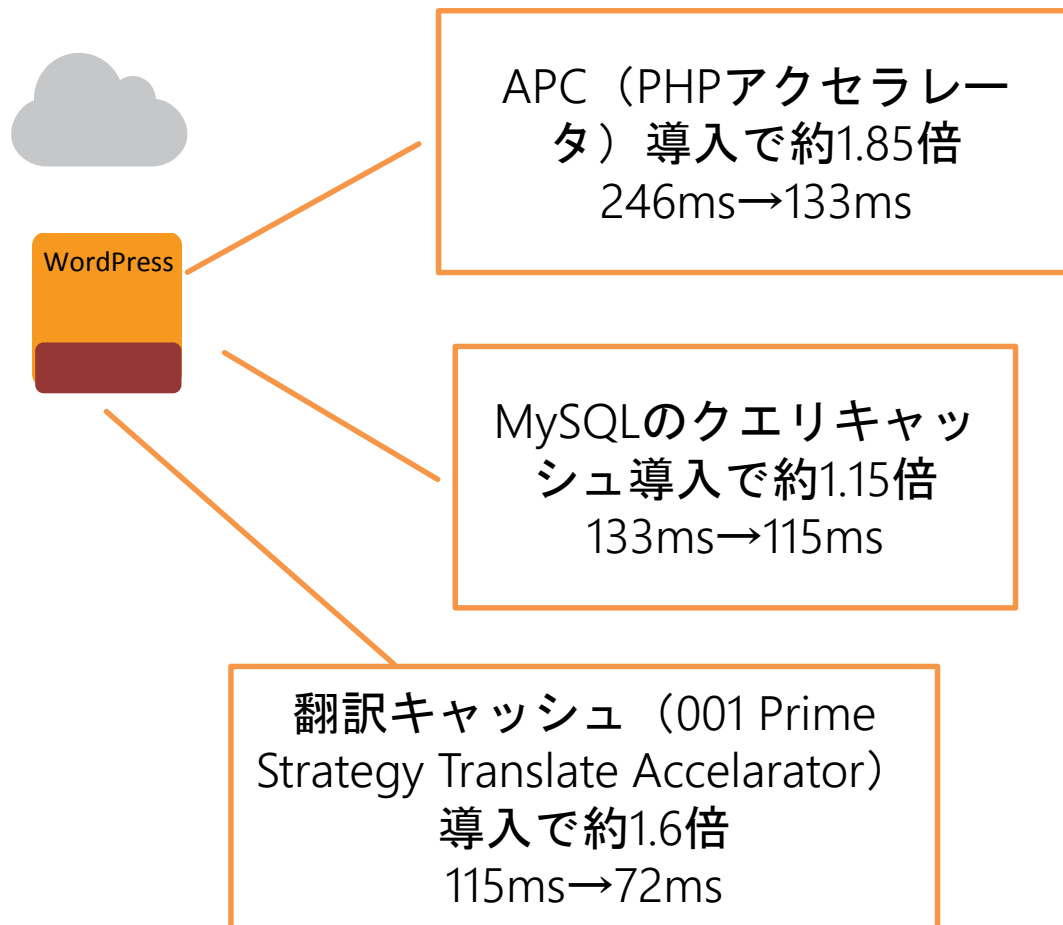
簡単に導入できます。

たとえば、my.cnfのmysqldセクションに次の1行を追加してMySQLサーバをrestartすればOK

```
query_cache_size = 64M
```

WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



APC (PHPアクセラレータ) 導入で約1.85倍
246ms→133ms

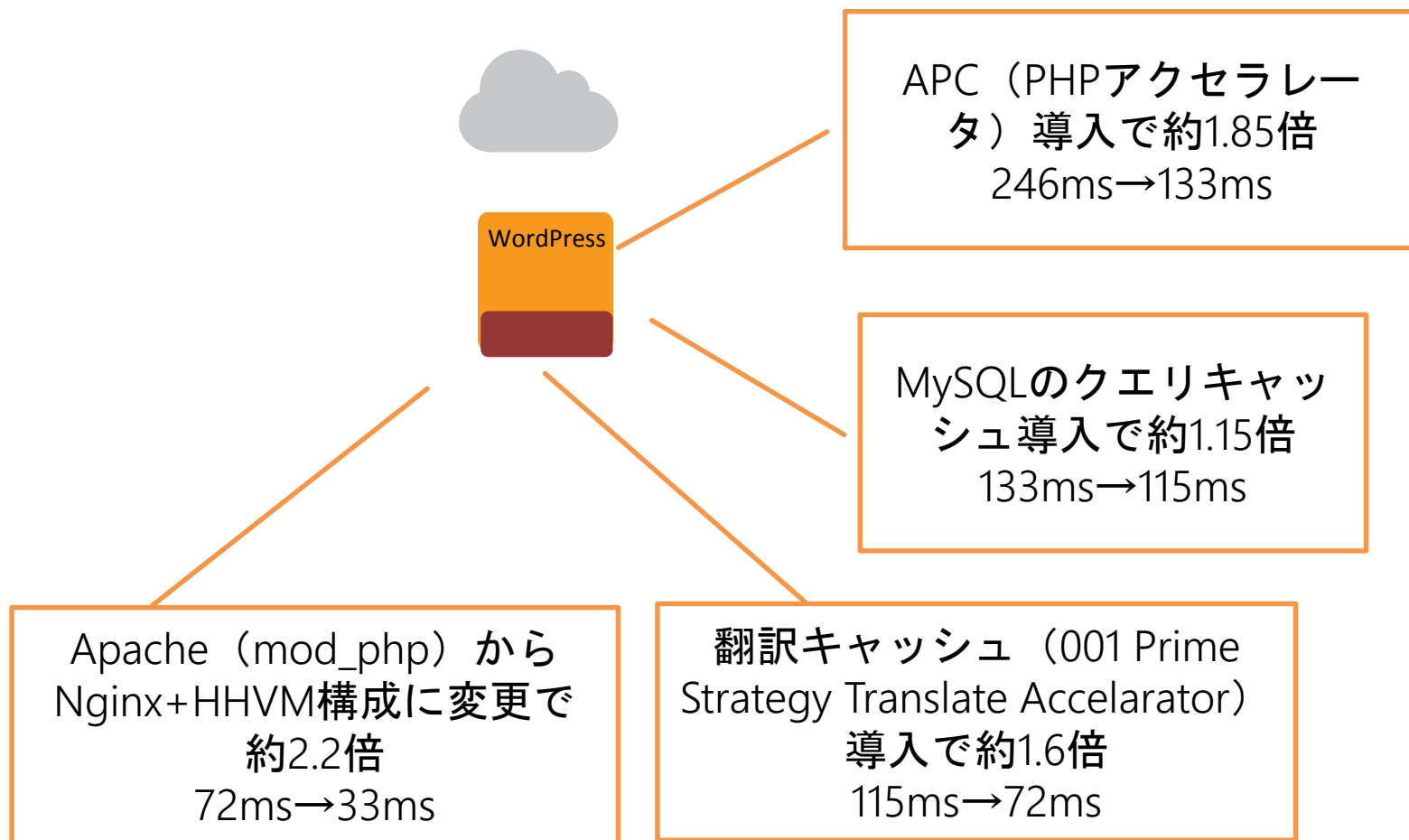
MySQLのクエリキャッシュ導入で約1.15倍
133ms→115ms

翻訳キャッシュ (001 Prime Strategy Translate Accelerator)
導入で約1.6倍
115ms→72ms

WordPressのプラグインなので簡単に導入できます。

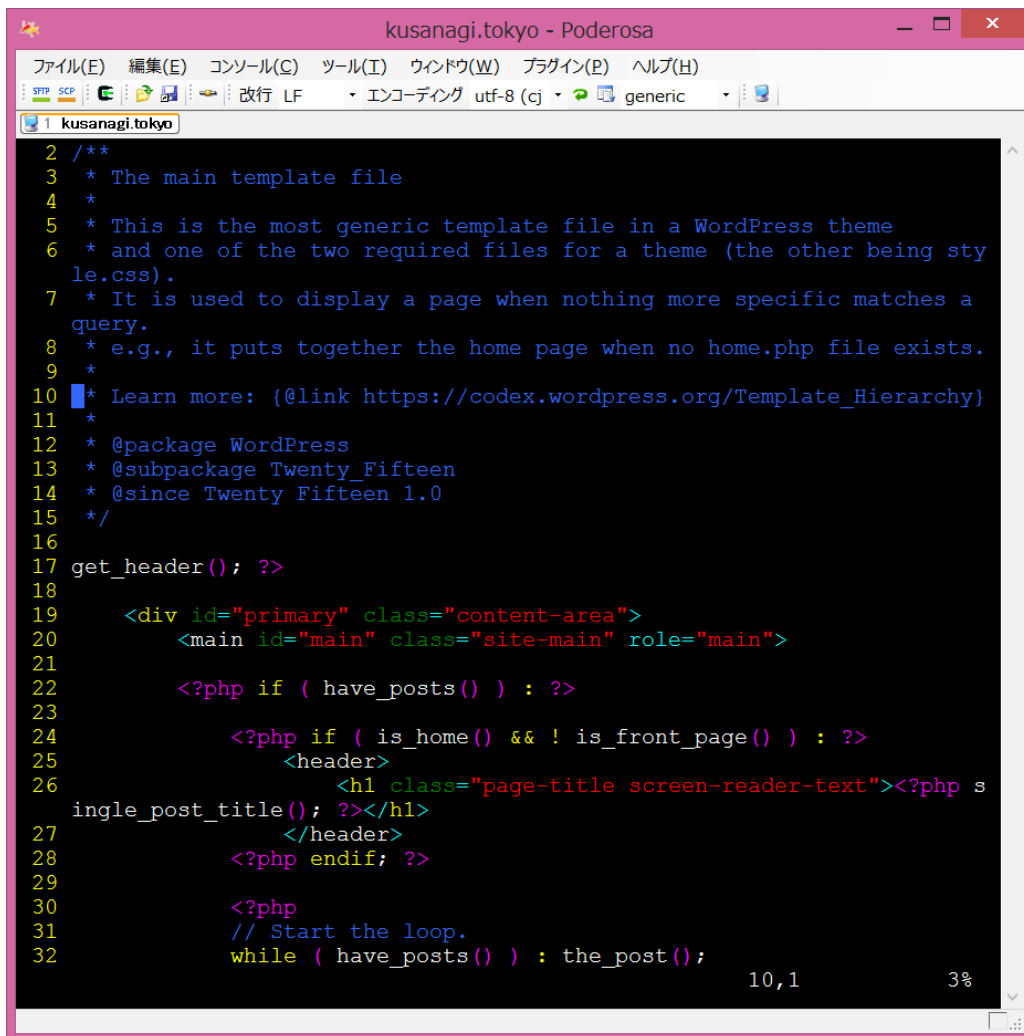
WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



WordPressを高速化する

HHVM実行の仕組み

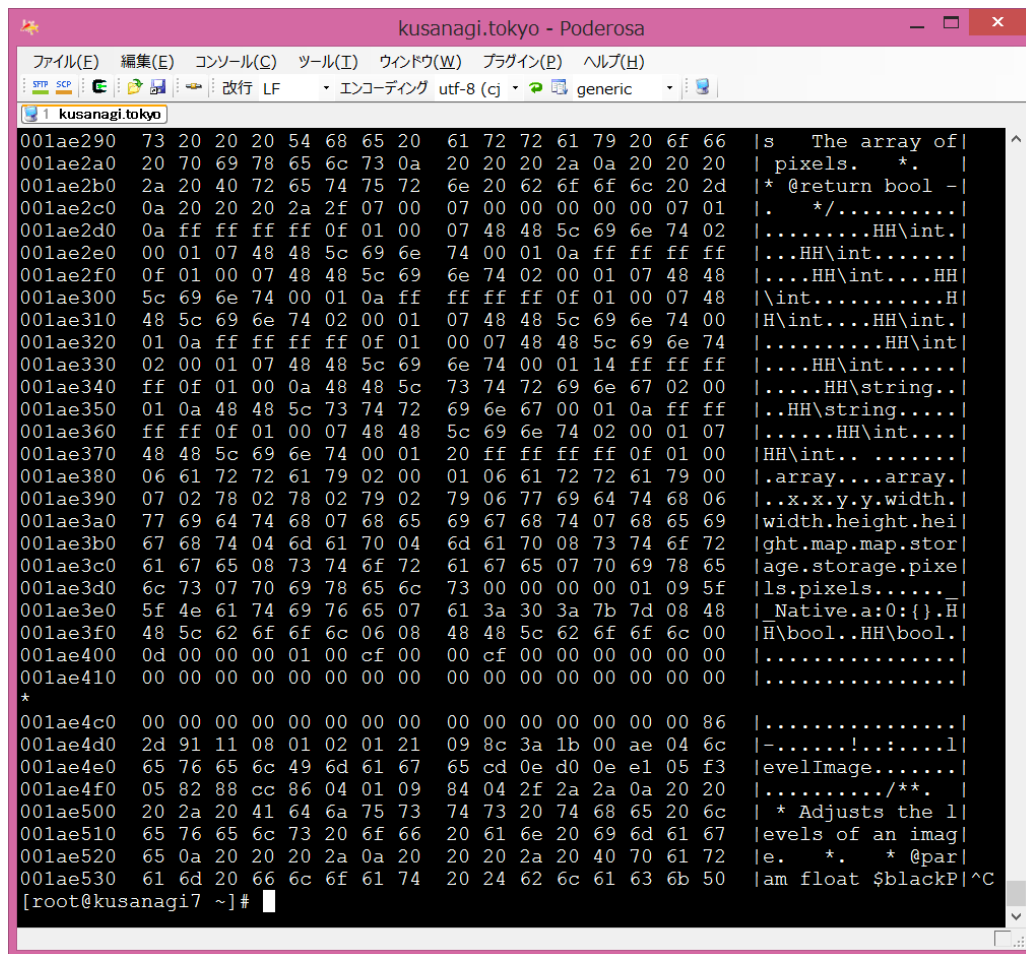


```
kusanagi.tokyo - Poderosa
ファイル(E) 編集(E) コンソール(C) ツール(I) ウィンドウ(W) プラグイン(E) ヘルプ(H)
SFTP SCP 改行 LF エンコーディング utf-8 (cj) generic
1 kusanagi.tokyo
2 /**
3  * The main template file
4  *
5  * This is the most generic template file in a WordPress theme
6  * and one of the two required files for a theme (the other being sty
7  * le.css).
8  * It is used to display a page when nothing more specific matches a
9  * query.
10 * e.g., it puts together the home page when no home.php file exists.
11 *
12 * Learn more: {@link https://codex.wordpress.org/Template_Hierarchy}
13 *
14 * @package WordPress
15 * @subpackage Twenty_Fifteen
16 * @since Twenty_Fifteen 1.0
17 */
18
19 get_header(); ?>
20
21 <div id="primary" class="content-area">
22   <main id="main" class="site-main" role="main">
23
24     <?php if ( have_posts() ) : ?>
25
26       <?php if ( is_home() && ! is_front_page() ) : ?>
27         <header>
28           <h1 class="page-title screen-reader-text"><?php s
29             ingle_post_title(); ?></h1>
30         </header>
31       <?php endif; ?>
32
33       <?php
34         // Start the loop.
35         while ( have_posts() ) : the_post();
36
37           10,1 3%
```

PHPのソースコード

WordPressを高速化する

HHVM実行の仕組み



```
kusanagi.tokyo - Poderosa
ファイル(E) 編集(E) コンソール(C) ツール(I) ウィンドウ(W) プラグイン(P) ヘルプ(H)
SFTP SCP 改行 LF エンコーディング utf-8 (cj) generic
1 kusanagi.tokyo
001ae290 73 20 20 20 54 68 65 20 61 72 72 61 79 20 6f 66 |s The array of
001ae2a0 20 70 69 78 65 6c 73 0a 20 20 20 2a 0a 20 20 20 | pixels. *. |
001ae2b0 2a 20 40 72 65 74 75 72 6e 20 62 6f 6f 6c 20 2d |* @return bool -|
001ae2c0 0a 20 20 20 2a 2f 07 00 07 00 00 00 00 00 07 01 |. */.....|
001ae2d0 0a ff ff ff ff 0f 01 00 07 48 48 5c 69 6e 74 02 |.....HH\int.|
001ae2e0 00 01 07 48 48 5c 69 6e 74 00 01 0a ff ff ff ff |...HH\int.....|
001ae2f0 0f 01 00 07 48 48 5c 69 6e 74 02 00 01 07 48 48 |...HH\int...HH|
001ae300 5c 69 6e 74 00 01 0a ff ff ff ff 0f 01 00 07 48 | \int.....H|
001ae310 48 5c 69 6e 74 02 00 01 07 48 48 5c 69 6e 74 00 |H\int...HH\int.|
001ae320 01 0a ff ff ff ff 0f 01 00 07 48 48 5c 69 6e 74 |.....HH\int|
001ae330 02 00 01 07 48 48 5c 69 6e 74 00 01 14 ff ff ff |...HH\int.....|
001ae340 ff 0f 01 00 0a 48 48 5c 73 74 72 69 6e 67 02 00 |...HH\string..|
001ae350 01 0a 48 48 5c 73 74 72 69 6e 67 00 01 0a ff ff |.HH\string....|
001ae360 ff ff 0f 01 00 07 48 48 5c 69 6e 74 02 00 01 07 |...HH\int....|
001ae370 48 48 5c 69 6e 74 00 01 20 ff ff ff ff 0f 01 00 |HH\int.. ..|
001ae380 06 61 72 72 61 79 02 00 01 06 61 72 72 61 79 00 |.array...array.|
001ae390 07 02 78 02 78 02 79 02 79 06 77 69 64 74 68 06 |..x.x.y.y.width.|
001ae3a0 77 69 64 74 68 07 68 65 69 67 68 74 07 68 65 69 |width.height.hei|
001ae3b0 67 68 74 04 6d 61 70 04 6d 61 70 08 73 74 6f 72 |ght.map.map.stor|
001ae3c0 61 67 65 08 73 74 6f 72 61 67 65 07 70 69 78 65 |age.storage.pixe|
001ae3d0 6c 73 07 70 69 78 65 6c 73 00 00 00 00 01 09 5f |ls.pixels.....|
001ae3e0 5f 4e 61 74 69 76 65 07 61 3a 30 3a 7b 7d 08 48 | Native.a:0:{}.H|
001ae3f0 48 5c 62 6f 6f 6c 06 08 48 48 5c 62 6f 6f 6c 00 |H\bool..HH\bool.|
001ae400 0d 00 00 00 01 00 cf 00 00 cf 00 00 00 00 00 00 | |
001ae410 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
*
001ae4c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 86 | |
001ae4d0 2d 91 11 08 01 02 01 21 09 8c 3a 1b 00 ae 04 6c |-.....!.:.....|
001ae4e0 65 76 65 6c 49 6d 61 67 65 cd 0e d0 0e e1 05 f3 |levelImage.....|
001ae4f0 05 82 88 cc 86 04 01 09 84 04 2f 2a 2a 0a 20 20 |...../**. |
001ae500 20 2a 20 41 64 6a 75 73 74 73 20 74 68 65 20 6c | * Adjusts the l|
001ae510 65 76 65 6c 73 20 6f 66 20 61 6e 20 69 6d 61 67 |levels of an imag|
001ae520 65 0a 20 20 20 2a 0a 20 20 20 2a 20 40 70 61 72 |e. *. * @par|
001ae530 61 6d 20 66 6c 6f 61 74 20 24 62 6c 61 63 6b 50 |am float $blackP|^C
[root@kusanagi1 ~]#
```

中間コード

←これを
HHVMが実行

WordPressを高速化する

HHVM実行の仕組み

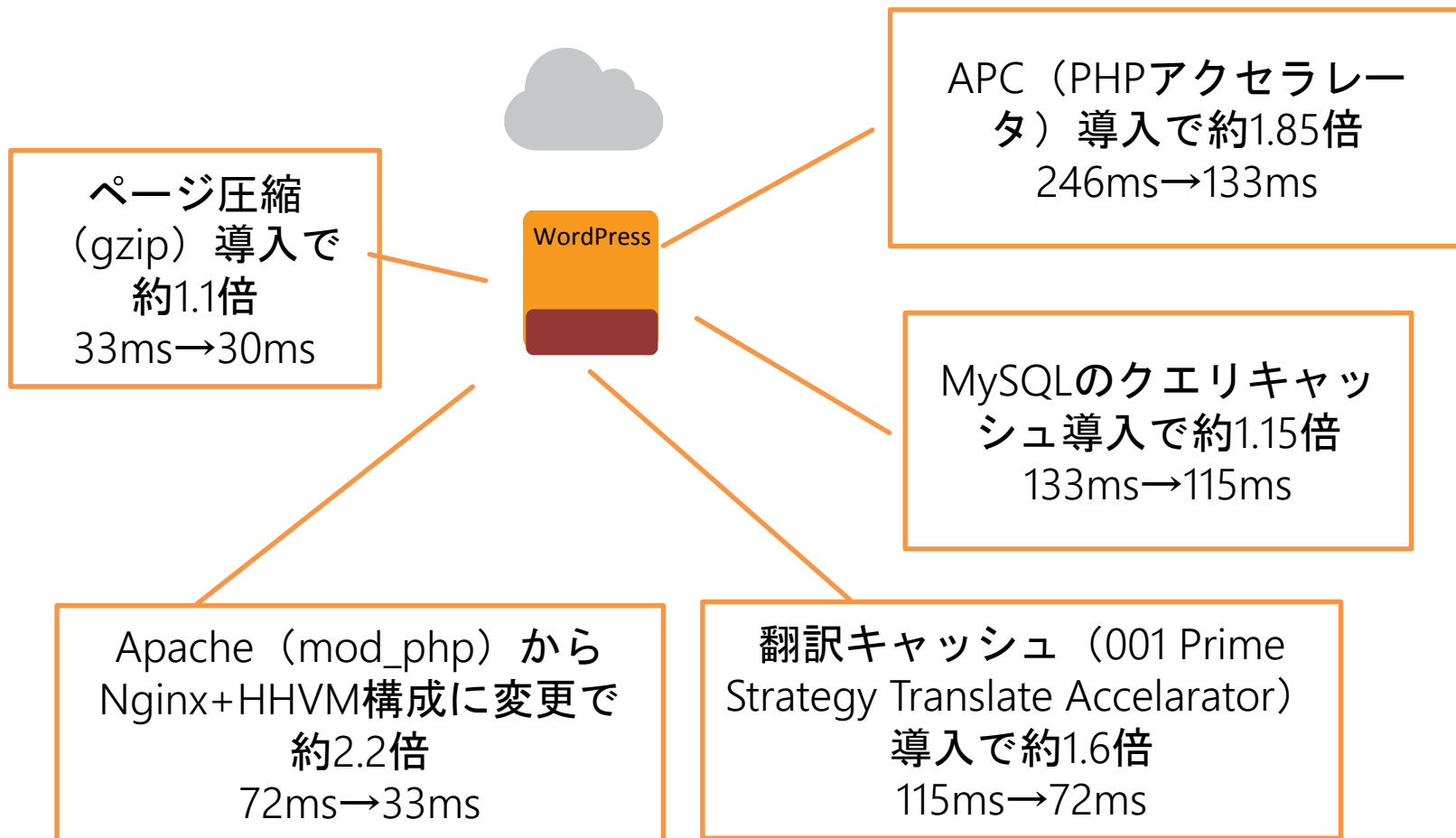
```
kusanagi.tokyo - Poderosa
ファイル(E) 編集(E) コンソール(C) ツール(I) ウィンドウ(W) プラグイン(P) ヘルプ(H)
SFTP SCP 改行 LF エンコーディング utf-8 (cj) generic
1 kusanagi.tokyo
001ae290 73 20 20 20 54 68 65 20 61 72 72 61 79 20 6f 66 |s The array of
001ae2a0 20 70 69 78 65 6c 73 0a 20 20 20 2a 0a 20 20 20 | pixels. *.
001ae2b0 2a 20 40 72 65 74 75 72 6e 20 62 6f 6f 6c 20 2d |* @return bool -
001ae2c0 0a 20 20 20 2a 2f 07 00 07 00 00 00 00 07 01 |. */.....
001ae2d0 0a ff ff ff ff 0f 01 00 07 48 48 5c 69 6e 74 02 |.....HH\int.
001ae2e0 00 01 07 48 48 5c 69 6e 74 00 01 0a ff ff ff ff |...HH\int.....
001ae2f0 0f 01 00 07 48 48 5c 69 6e 74 02 00 01 07 48 48 |...HH\int....HH
001ae300 5c 69 6e 74 00 01 0a ff ff ff ff 0f 01 00 07 48 | \int.....H
001ae310 48 5c 69 6e 74 02 00 01 07 48 48 5c 69 6e 74 00 |H\int....HH\int.
001ae320 01 0a ff ff ff ff 0f 01 00 07 48 48 5c 69 6e 74 |.....HH\int
001ae330 02 00 01 07 48 48 5c 69 6e 74 00 01 14 ff ff ff |...HH\int.....
001ae340 ff 0f 01 00 0a 48 48 5c 73 74 72 69 6e 67 02 00 |...HH\string..
001ae350 01 0a 48 48 5c 73 74 72 69 6e 67 00 01 0a ff ff |..HH\string....
001ae360 ff ff 0f 01 00 07 48 48 5c 69 6e 74 02 00 01 07 |.....HH\int....
001ae370 48 48 5c 69 6e 74 00 01 20 ff ff ff ff 0f 01 00 |HH\int.. .....
001ae380 06 61 72 72 61 79 02 00 01 06 61 72 72 61 79 00 |.array...array.
001ae390 07 02 78 02 78 02 79 02 79 06 77 69 64 74 68 06 |..x.x.y.y.width.
001ae3a0 77 69 64 74 68 07 68 65 69 67 68 74 07 68 65 69 |width.height.hei
001ae3b0 57 68 74 04 6d 61 70 04 6d 61 70 08 73 74 6f 72 |ght.map.map.stor
001ae3c0 51 67 65 08 73 74 6f 72 61 67 65 07 70 69 78 65 |age.storage.pixe
001ae3d0 5c 73 07 70 69 78 65 6c 73 00 00 00 01 09 5f |ls.pixels.....
001ae3e0 5f 4e 61 74 69 76 65 07 61 3a 30 3a 7b 7d 08 48 | Native.a:0:{}.H
001ae3f0 48 5c 62 6f 6f 6c 06 08 48 48 5c 62 6f 6f 6c 00 |H\bool..HH\bool.
001ae400 0d 00 00 00 01 00 cf 00 00 cf 00 00 00 00 00 |.....
001ae410 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....
*
001ae4c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 86 |.....
001ae4d0 2d 91 11 08 01 02 01 21 09 8c 3a 1b 00 ae 04 6c |-.....!.:.....l
001ae4e0 55 76 65 6c 49 6d 61 67 65 cd 0e d0 0e e1 05 f3 |levelImage.....
001ae4f0 05 82 88 cc 86 04 01 09 84 04 2f 2a 2a 0a 20 20 |...../**.
001ae500 20 2a 20 41 64 6a 75 73 74 73 20 74 68 65 20 6c | * Adjusts the l
001ae510 65 76 65 6c 73 20 6f 66 20 61 6e 20 69 6d 61 67 |levels of an imag
001ae520 65 0a 20 20 20 2a 0a 20 20 2a 20 40 70 61 72 |e. *. * @par
001ae530 61 6d 20 66 6c 6f 61 74 20 24 62 6c 61 63 6b 50 |am float $blackP| ^C
[root@kusanagi17 ~]#
```

何度も利用される部分をJITでコンパイルしてネイティブコードへ

ネイティブコードをCPUが実行

WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか



WordPressを高速化する

ページキャッシュを使わないでどこまでいけるか

1. ロード時間	246ms
2. リクエスト数	4.9リクエスト/秒



ロード時間 約8倍
リクエスト数 約11.6倍に向上

1. ロード時間	30ms
2. リクエスト数	56.8リクエスト/秒

WordPressを高速化する

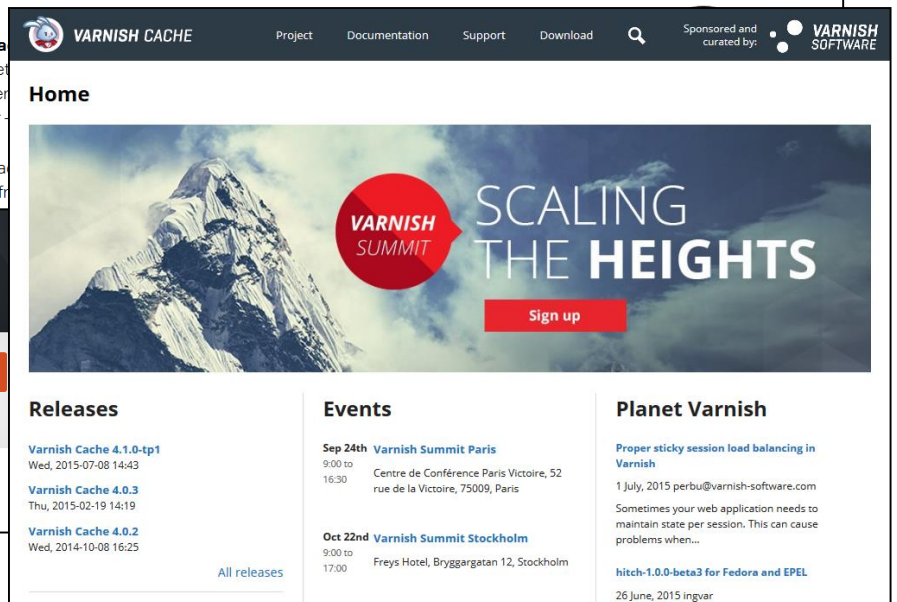
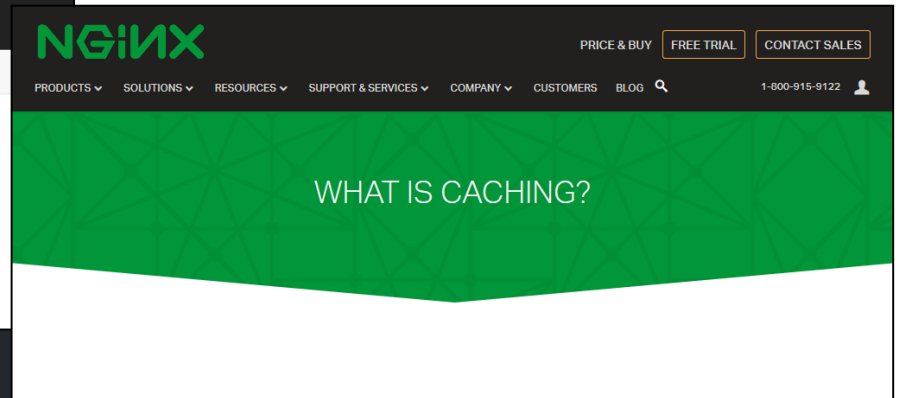
ページキャッシュを使わないでどこまでいけるか

1. ロード時間	30ms
2. リクエスト数	56.8リクエスト/秒

このサーバのCPUの周波数とコア数を変更すると
どのような影響があるのか？

5. (おまけ) ページキャッシュとトランジェント

ページキャッシュを導入する (WP SiteManager、WP Super Cache、Nginx、Varnishなど)



ページキャッシュを導入する WP SiteManagerの場合

1. ロード時間	246ms
2. リクエスト数	4.9リクエスト/秒



ロード時間 約16.4倍
リクエスト数 約53.1倍に向上

1. ロード時間	15ms
2. リクエスト数	260リクエスト/秒

トランジェントを導入する トランジェントとは？



WordPress内部の値を一時的にDB (wp_optionsテーブル)に保存して、異なるプロセスで再利用するためのWordPressの機能
=>
部分キャッシュとして利用可能

トランジェントを導入する

トランジェントの具体例 (footer.php)

```
<?php
if ( ! $footer_cache = get_transient(
'footer_cache' ) ) {
    ob_start();
?>
<footer id="colophon" class="site-footer"
role="contentinfo">
    <?php get_sidebar( 'footer' ); ?>
    <div class="site-info">
        <?php do_action(
'twentyfourteen_credits' ); ?>
```

```
<a href="<?php echo esc_url( __(
'http://wordpress.org/', 'twentyfourteen' ) );
?>"><?php printf( __( 'Proudly powered by
%s', 'twentyfourteen' ), 'WordPress' ); ?></a>
</div><!-- .site-info -->
</footer><!-- #colophon -->
<?php
    $footer_cache = ob_get_clean();
    set_transient( 'footer_cache',
$footer_cache, 60 * 5 );
}
echo $footer_cache;
?>
```

ということで

ぜひ実践して

快適なWordPressライフをお過ごしください！



ご清聴ありがとうございました。