



SYMFONY DAY

TORINO, 5 OTTOBRE 2012

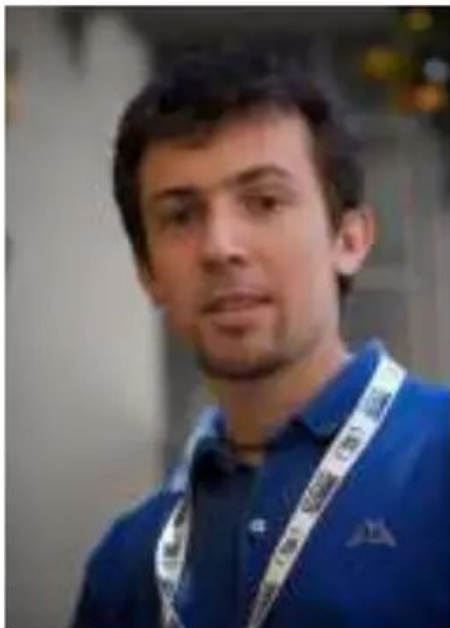
Use **Symfony2** components inside **WordPress**

(...and live happily)



Walter Dal Mut

- Cofounder at Corley S.r.l.
- First level degree Electronic Engineering
- Ungraduated student of Computer Science



Maurizio Pelizzone

- Cofounder at Mavida S.n.c
- ***WordPress Lover***



Technologies

[Content Management](#)
[Server-side Languages](#)
[Client-side Languages](#)
[JavaScript Libraries](#)
[Markup Languages](#)
[Character Encodings](#)
[Image File Formats](#)
[Site Elements](#)
[SSL Certificate Authorities](#)
[Social Widgets](#)
[Web Servers](#)
[Operating Systems](#)
[Content Delivery](#)
[Traffic Analysis Tools](#)
[Advertising Networks](#)
[Top Level Domains](#)
[Content Languages](#)

Trends

[Usage Trend](#)
[Market Share Trend](#)

Market

[Top Site Usage](#)
[Market Position](#)

Breakdown

[Ranking](#)
[Multiple Technology Usage](#)
[Server-side Languages](#)
[Client-side Languages](#)
[JavaScript Libraries](#)
[Markup Languages](#)
[Character Encodings](#)
[Image File Formats](#)
[Site Elements](#)
[SSL Certificate Authorities](#)
[Social Widgets](#)
[Web Servers](#)
[Operating Systems](#)
[Content Delivery](#)
[Traffic Analysis Tools](#)
[Advertising Networks](#)
[Top Level Domains](#)

[Technologies](#) > [Content Management](#)

Usage of content management systems for websites

This diagram shows the percentages of websites using various content management systems. See [technologies overview](#) for explanations on the methodologies used in the surveys. Our reports are updated daily.

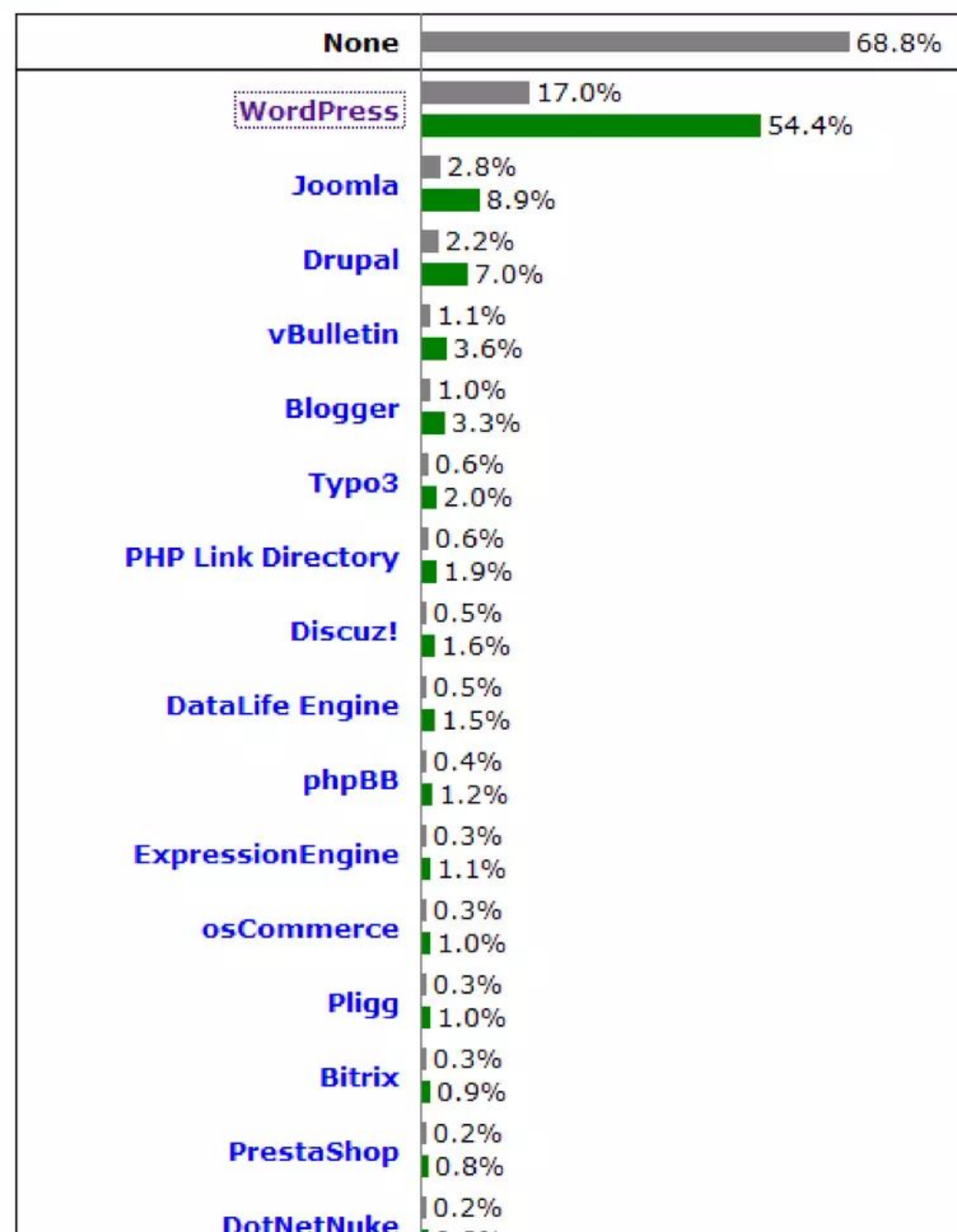
How to read the diagram:

68.8% of the websites use none of the content management systems that we monitor.

WordPress is used by 17.0% of all the websites, that is a content management system market share of 54.4%.

Request an extensive
market report of
specific content
management systems.

[Learn more](#)



Technology Brief

Content Management Systems

Content management systems are web-based applications for creating and managing the content of a website. We include all such systems in this category, also systems that are often classified as wikis, blog engines or discussion boards.

Latest related posting [read all](#)

Web technology fact of the day 24 September 2012

30.1% of all Linux-based websites run one of the established content management systems, but only 9.8% of Windows-based sites.
[» more](#)

Latest related forum entry [read all](#)

New tools in our surveys 29 September 2012 [» more](#)

Our Book Recommendation

*Symfony2 is a reusable set of standalone,
decoupled, and cohesive PHP
components that solve common web
development problems*

Fabien Potencier

<http://fabien.potencier.org/article/49/what-is-symfony2>

Use Composer autoload in your theme

```
1  <?php
2
3  /*
4   * functions.php
5   * include autoload generated generated by Composer
6   */
7  require_once TEMPLATEPATH . '/vendor/autoload.php';
8
9
10
11
```

WordPress & Monolog

Monolog is a logging library for PHP 5.3 used by Symfony2. It is inspired by the Python LogBook library.

Get Monolog

Website: <https://github.com/Seldaek/monolog>

Using Monolog:

Create a new composer.json file

```
{  
    "require": {  
        "monolog/monolog": "1.1.*"  
    }  
}
```

```
curl -s http://getcomposer.org/installer | php  
php composer.phar install
```

wrap inside WordPress...

```

1  <?php
2
3  /**
4   * WordPress Monolog Wrapper
5   *
6   * Usage:
7   * $logger = new Logger( );
8   * $logger->info('We just got the logger');
9   */
10 Class Logger {
11
12     private $logger;
13
14     function __construct( $args = array() ) {
15
16         $defaults = array(
17             'name' => 'WordPress_Logger',
18             'path' => TEMPLATEPATH . '/log.txt',
19             'level' => Monolog\Logger::DEBUG,
20         );
21
22         $args = wp_parse_args( $args, $defaults );
23         extract( $args, EXTR_SKIP );
24
25         $this->logger = new Monolog\Logger($name);
26         $this->logger->pushHandler(new Monolog\Handler\StreamHandler( $path, $level ));
27
28     }
29
30     // wrap function to all 8 Monolog logging levels
31     function notice    ( $message, array $context = array() ) { $this->logger->notice( $message , $context ); }
32     function emerg     ( $message, array $context = array() ) { $this->logger->emerg( $message , $context ); }
33     function debug     ( $message, array $context = array() ) { $this->logger->debug( $message , $context ); }
34     function info      ( $message, array $context = array() ) { $this->logger->info( $message , $context ); }
35     function warn      ( $message, array $context = array() ) { $this->logger->warn( $message , $context ); }
36     function err        ( $message, array $context = array() ) { $this->logger->err( $message , $context ); }
37     function alert     ( $message, array $context = array() ) { $this->logger->alert( $message , $context ); }
38     function critical   ( $message, array $context = array() ) { $this->logger->critical( $message , $context ); }
39
40 }
41
42
43

```

... and use in your theme

1 <?php

2
3 /*
4 * functions.php
5 * include autoload generated generated by Composer
6 */

7 require_once TEMPLATEPATH . '/vendor/autoload.php';
8 require("class/logger.php");
9

10 /*
11 * by default log all message in log.txt
12 */

13 \$log = new Logger();

14
15 \$log->info('We just got the logger');
16 \$log->warn('Warging!!!');
17 \$log->err('An error occurred');
18
19

20 /*
21 * customize name, path and log level
22 */

23
24 \$args = array(
25 'name' => 'Template_Log',
26 'path' => TEMPLATEPATH . '/logs/template.log',
27 'level' => 400, // ERROR
28);

29
30 \$log2 = new Logger(\$args);
31

32 \$log2->err('This one is template error');
33

```
<?php
2
3 /**
4  * Smf Day Theme Class
5  */
6 Class SmfDayTheme {
7
8     private $log;
9
10    function __construct( ) {
11
12        // add hook
13        add_filter("query", array( $this , "QueryLog" ) );
14    }
15
16    /**
17     * callback function to log query
18     */
19    function QueryLog( $query ) {
20
21        $args = array(
22            'name' => 'Query_Log',
23            'path' => TEMPLATEPATH . '/logs/query.log',
24            'level' => 200, // info
25        );
26
27        // check instance
28        if (is_null($this->log)) {
29            $this->log = new Logger( $args );
30        }
31
32        // log all "select" query run by frontend
33        if ( !is_admin() && preg_match( '/^\s*(select) /i', $query ) ) {
34
35            $this->log->info( $query );
36        }
37    }
38
39 } // end class
```

WordPress & Assetic

Assetic is an asset management framework for PHP.

Assetic is based on the Python webassets library

Get Assetic

Website: <https://github.com/kriswallsmith/assetic>

Using Assetic:

Create a new composer.json file

```
{  
    "require": {  
        "kriswallsmith/assetic": "v1.0.4"  
    }  
}
```

```
curl -s http://getcomposer.org/installer | php  
php composer.phar install
```

*GOAL: Merge, Minimize and Compile all
javascript loaded through WordPress*



```
<?php
```

2

3



```
/*
```

4

```
 * functions.php
```

5

```
 * include autoload generated generated by Composer
```

6

```
 */
```

7

```
require_once TEMPLATEPATH . '/vendor/autoload.php';
```

8

9



```
if ( !is_admin() ) {
```

10

11

```
    // include lib
```

12

```
    include("libs/assetic.php");
```

13

```
    include("libs/load-script.php");
```

14

15

```
    // load js file
```

16

```
    add_action( 'wp_enqueue_scripts', 'load_script' );
```

17

18

```
    // run merge + minification
```

19

```
    add_action( 'wp_footer', 'run_assetic' , 1);
```

20

21



```
}
```

22

23

24

25

26

27



```
<?php
```

2

3



```
/**
```

4

```
 * default function to load script in wordpress way
```

5

```
 */
```

6



```
function load_script() {
```

7

8

```
    wp_enqueue_script('jquery');
```

9

10

```
    wp_enqueue_script("fancy",
```

11

```
        get_template_directory_uri() . '/js/fancy.js', '', '', true );
```

12

13

```
    wp_enqueue_script("wp-seo-admin",
```

14

```
        get_template_directory_uri() . '/js/wp-seo-admin.js', '', '', true );
```

15

16

```
    wp_enqueue_script("wp-seo-metabox",
```

17

```
        get_template_directory_uri() . '/js/wp-seo-metabox.js', '', '', true );
```

18

19

```
    wp_enqueue_script("core",
```

20

```
        get_template_directory_uri() . '/js/core.js', '', '', true );
```

21

22

```
    wp_enqueue_script("main",
```

23

```
        get_template_directory_uri() . '/js/main.js', '', '', true );
```

24

25

```
    wp_enqueue_script("file_a",
```

26

```
        get_template_directory_uri() . '/js/core.js', '', '', true );
```

27

28

29

30



```
 }
```

31

32

<?php

/**

* for each script enqueued:

* add to assetic collection and remove from wordpress queue

*/

function run_assetic() {

global \$wp_scripts; // wordpress script container

\$AssetCollection = array();

foreach (\$wp_scripts->in_footer as \$script_name) {

// wordpress admin js

if (\$script_name != "admin-bar") {

\$registered = \$wp_scripts->registered;

\$AssetCollection[] = new Assetic\Asset\FileAsset(\$registered[\$script_name]->src);

wp_dequeue_script(\$script_name); // remove from queue

}

}

\$js = new Assetic\Asset\AssetCollection(\$AssetCollection, array(

new Assetic\Filter\GoogleClosure\CompilerApiFilter(),

));

\$assetic_js = TEMPLATEPATH . '/cache/assetic.js';

\$assetic_cachetime = 3600;

\$assetic_created = ((@file_exists(\$assetic_js))) ? @filemtime(\$assetic_js) : 0;

if ((\$assetic_created - (time() - \$assetic_cachetime)) < 0) {

file_put_contents(\$assetic_js , \$js->dump());

};

if (file_exists(\$assetic_js))

wp_enqueue_script("assetic", get_template_directory_uri() . '/cache/assetic.js','',true);

} // end function

Code example:

<https://github.com/miziomon/symfonyday>

WordPress & Twig

*Twig: the flexible, fast, and secure
template engine for PHP.*

*Twig uses a syntax similar to the
Django and Jinja template
languages which inspired the Twig
runtime environment.*

Why using a Template Engine ?

PHP doesn't support many features modern template languages should have nowadays

- Concision
- Template oriented Syntax
- Reusability
- "Security" by default
 - I'll explain in the next slides
 - Read more on dedicated webpages.

Read more at: <http://fabien.potencier.org/article/34/templating-engines-in-php>

Concision

PHP

```
<?php echo $name ?>
```

```
<?php echo  
    htmlspecialchars(  
        $name,  
        ENT_QUOTES,  
        'UTF-8')  
?>
```

Short tags

```
<?=$name?>
```

Using short tags could creates problems. You have to enable the short tags that are disabled by default.

TWIG

```
{{name}}
```

```
{{name|escape}}
```

No short version it's
already the short
version.

Template oriented Syntax

PHP

```
<?php if ($items): ?>  
  <?php foreach ($items as $item): ?>  
    * <?php echo $item ?>  
  <?php endforeach; ?>  
<?php else: ?>  
  No item has been found. <?php endif;  
  ?>
```

TWIG

```
{% for item in items %}  
  * {{ item }}  
{% else %}  
  No item has been found.  
{% endfor %}
```

It's more clear and use
a new syntax.

Reusability

```
<!-- base.html -->
<html
  xmlns="http://www.w3.org/1999/xhtml"
  xml:lang="en" lang="en">
<head>
  {% block head %}
    <link rel="stylesheet" href="main.css" />
  {% endblock %}
</head>
<body>
  {% block content %}{% endblock %}
</body>
</html>
```

```
<!-- index.html -->
{% extends "base.html" %}

{% block head %}
  {{ block.super }}
  <link rel="stylesheet" href="main.css" />
{% endblock %}

{% block content %}
  Index content
{% endblock %}
```

Security

" I'm not saying PHP is not a secure language, far from it. But needless to say that escaping a variable in a template is just a nightmare.

" Sometimes, security should be enabled by default, especially for templates written by non-developers who are not necessarily aware of the common web threats like XSS or CSRF.

@fabpot

PHP

```
<?php echo  
    htmlspecialchars(  
    $var,  
    ENT_QUOTES,  
    'UTF-8') ?>
```

TWIG

Output is escaped by default.

```
{% autoescape off %}  
    {{ object.as_html }}  
{% endautoescape %}
```

I'm well aware of the automatic output escaping problems.

Get Twig

Website: <http://twig.sensiolabs.org>

Twig C Extensions: <https://github.com/derickr/twig-ext.git>

Using composer:

Create a new composer.json file

```
{  
    "require": {  
        "twig/twig": "1.*"  
    }  
}
```

```
curl -s http://getcomposer.org/installer | php  
php composer.phar install
```

Twig & WordPress

Using Twig into a wordpress plugin/theme could simplify and improve the development.

- Plugin development (views)
- Theme development

Integrating Twig into a Plugin

Just initialize Twig and use it. Using a static reference to share the Twig instance (simple way).

```
function get_twig()  
{  
    static $twig;  
    if (!$twig) {  
        $loader = new Twig_Loader_Filesystem(__DIR__ . '/templates');  
        $twig = new Twig_Environment($loader, array('cache' => false));  
    }  
  
    return $twig;  
}
```

Render a template

During the Twig initialization we specify the view location ("templates" folder). Now you can create a Twig view and save it. Into your plugin just link to it:

```
$template = get_twig()->loadTemplate('tweets.twig');  
echo $template->render(array('elements' => $unpack));
```

In this example we load the "tweets.twig" view and pass the elements variable.

WP Plugin Example

You can checkout an example at:

- <https://github.com/wdalmut/wp-simple-tweets>

In this example we add a widget into the admin dashboard that shows tweets of @CorleyCloud account.

Make a template using Twig

WordPress is designed to use PHP as template engine. We can force to integrate Twig as the default templating system. You can find an example of integration at:

<https://github.com/wdalmut/wp-twig-theme>

This example realize a simple blank template with some WP functionalities (sidebars, posts, pages, archives).

Making a Twig Proxy

WordPress uses a lot of functions to generate pieces of html. To speedup the Twig integration we can create a simple proxy that enable WP functions into a Twig template.

```
class TwigProxy {  
    public function __call($function, $arguments) {  
        if (!function_exists($function)) {  
            throw new Exception("{ $function } not  
exists.");  
            return NULL;  
        }  
        return call_user_func_array($function,  
$arguments);  
    }  
}
```

Using the Twig Proxy

This proxy allows us to call WordPress functions into a view.

```
<div id="sidebar">
    {% if not wp.dynamic_sidebar("Sidebar Widgets") %}

    {{ wp.get_search_form }}
<!-- continue -->
</div>
```

"dynamic_sidebar" and
"get_search_form" are WP functions.

Load Twig templates instead PHPs

We can use WordPress hooking system to override the default template loading.

- `add_filter`
 - `home_template`
 - `single_template`
 - `page_template`
 - `404_template`
 - `archive_template`
 - Etc, etc...
- `add_action`
 - `template_include`
 - Engage Twig and prepare data to view.

Layout management

Instead of use `get_header` and `get_footer` that break HTML we can use template inheritance of Twig.

```
{% extends "base.twig" %}
```

```
{% block title %}
```

```
    {{ parent() }} | Page Not Found
```

```
{% endblock %}
```

```
{% block content %}
```

```
<h2>Error 404 - Page Not Found</h2>
```

```
{% endblock %}
```

Enjoy with Twig

Twig is extremely simple and powerful, you can play and enjoy with this great template engine and get more power during your wp development.

Have a nice day

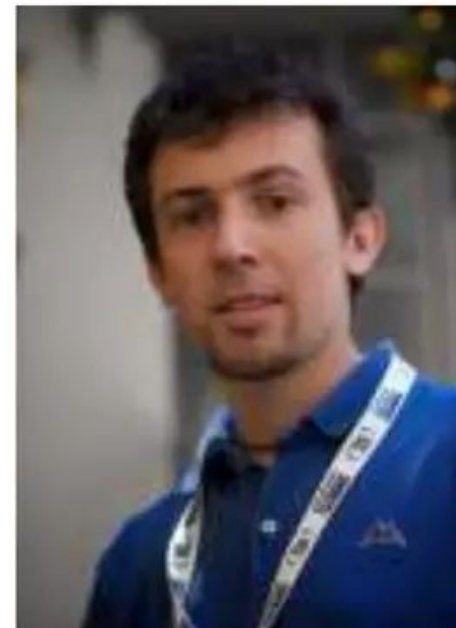


Thank you!



Walter Dal Mut

walter.dalmut@gmail.com
@wdalmut



Maurizio Pelizzone

maurizio@mavida.com
@miziomon