



# Improving Performance of Code I have seen in the wild

Pascal Knecht

E-Mail: [pascal.knecht99@gmail.com](mailto:pascal.knecht99@gmail.com)

Twitter: [@revenwo](https://twitter.com/revenwo)

GitHub: <https://github.com/pascalknecht>

# About me

- Software Engineer based in Thun
- Currently doing my bachelor's degree in Computer Science at the FHNW
- Working as a full-stack developer with WordPress for over 3 years

# Web Performance Impacts Conversion Rates



**You have 5 seconds**  
to engage a customer  
before they leave your  
web site.

**46%**

of users will  
NOT return to a  
poorly performing  
website.

Just a one second delay in load time  
means a loss in conversions:

**7%**

Loss in  
Conversions

**11%**

Fewer Page  
Views

**16%**

Decrease in  
Customer  
Satisfaction



**1 in 4 People**  
will abandon a website  
if it takes longer than  
4 seconds to load

**74%**

of users will abandon  
after waiting 5 seconds  
for a mobile site to load

Peak Load Time  
for Conversions

**2  
Seconds**

Waiting 500 ms causes  
stress and frustration:

**26%**

Increase  
in Frustration

**8%**

Decrease  
in Engagement

For every 1 second  
speed improvement to  
the Amazon website  
conversions increased

**+ 2%**

Mozilla made pages  
2.2 seconds faster. . .

**60**

Million More  
Firefox Downloads  
per Year

A \$250 million  
presidential fundraising site  
increased speed 60% . . .

**14%**

Increase In Donation  
Conversions

# What should we do to improve Performance?

- Optimize MYSQL-Queries that are generated by WordPress
- Optimize PHP-Code
- Optimize frontend assets loading (GZIP Compression, HTTP 2 (Server Push), Browser Caching)
- Remove unnecessary code

# Code

```
1  <?php
2  function getCantons() {
3      global $term;
4      $args = array(
5          'parent' => 0,
6          'hide_empty' => false,
7          'taxonomy' => 'dl_canton_cat'
8      );
9      $taxChildren = get_terms('dl_canton_cat', $args);
10     $cantons = array();
11     foreach ($taxChildren as $childTerm) {
12         $canton = array();
13         $canton['classes'] = array();
14         $postArgs = array(
15             'post_type' => 'dl',
16             'post_status' => 'publish',
17             'posts_per_page' => -1,
18             'tax_query' => array(
19                 array(
20                     'taxonomy' => 'dl_canton_cat',
21                     'field' => 'slug',
22                     'terms' => array($childTerm->slug)
23                 )
24             )
25         );
26         if ($term) {
27             $postArgs['tax_query']['relation'] = 'AND';
28             array_push($postArgs['tax_query'], array(
29                 'taxonomy' => 'dl_cat',
30                 'field' => 'slug',
31                 'terms' => array($term->slug),
32                 'include_children' => true
33             ));
34         }
35         $posts = new WP_Query($postArgs);
36         if ($posts->post_count > 0) {
37             array_push($canton['classes'], 'has-items');
38         } else {
39             array_push($canton['classes'], 'no-items');
40         }
41         array_push($cantons, $canton);
42     }
43     return $cantons;
44 }
45
```

# Solution: Option/Transient API

- Data – stored in wp\_options
- Transients
  - Transients expire after a given time
  - Non-deterministic caching – have fallback code in place to generate the necessary data
  - Will use object cache if possible, otherwise MySQL
  - Transients may expire before the expiration date (due to external object caches, or database upgrades)

# Improved Code

```
11 $cached_option = get_option('canton_counter');
12 foreach ($taxChildren as $childTerm) {
13     $canton = array();
14     $canton['classes'] = array();
15     $count = $childTerm->post_count;
16     if ($term) {
17         $option_key = $term->slug;
18
19         if(isset($cached_option[$option_key]) && $cached_option[$option_key] !== false) {
20             $count = (int)$cached_option[$option_key];
21         } else {
22             $postArgs = array(
23                 'post_type' => 'dl',
24                 'post_status' => 'publish',
25                 'posts_per_page' => -1,
26                 'tax_query' => array(
27                     'relation' => 'AND',
28                     array(
29                         'taxonomy' => 'dl_canton_cat',
30                         'field' => 'slug',
31                         'terms' => array($childTerm->slug)
32                     ),
33                     array(
34                         'taxonomy' => 'dl_cat',
35                         'field' => 'slug',
36                         'terms' => array($term->slug),
37                         'include_children' => true
38                     )
39                 )
40             );
41             $posts = get_posts($postArgs);
42             $count = count($posts);
43
44             $cached_option[$option_key] = $count;
45
46             update_option('canton_counter', $cached_option);
47         }
48     }
```

# Use For

- Options:
  - Data which does not expire and you want to have until you clear it
- Transients:
  - Caching data
  - Data which is heavy or slow to compute every time
  - Cache data which was fetched from an API



# Code

```
1  <?php
2  /*
3  archive.php
4  */
5  $temp_query = $wp_query;
6  $wp_query = null;
7  $wp_query = new WP_Query(array(
8      'post_type' => 'forum',
9      'posts_per_page' => -1
10 ));
11 while(have_posts()) {
12     the_post();
13     // Do something with post
14 }
15
16 $wp_query = null;
17 $wp_query = $temp_query;
18 wp_reset_postdata();
19
```

# Solution: pre\_get\_posts Hook

- The pre\_get\_posts Hook can be used to alter the main query

# Improved Code

```
1  <?php
2  /*
3  functions.php
4  */
5  function filter_forum_query($query) {
6      if($query->is_main_query() && is_tax('forum_tax') && !is_admin()) {
7          $query->set('post_type', 'forum');
8          $query->set('posts_per_page', -1);
9      }
10 }
11 add_action('pre_get_posts', 'filter_forum_query');
12
```

# Use For

- Taxonomy and archive pages to not create a second loop to prevent getting the same data as in the main query
- Custom redirect rules to completely change the query

# Code

```
1  <?php
2  add_action( 'wp_enqueue_scripts', 'dst_enqueue_scripts', 0 );
3  function dst_enqueue_scripts() {
4
5      wp_enqueue_script('jquery');
6
7      wp_enqueue_script(
8          'dstembed',
9          '/dist/app.js',
10         array('jquery'),
11         false,
12         false
13     );
14 }
15
16 function dst_shortcode( $args ) {
17     extract( shortcode_atts( array(
18         'name'      => 'error',
19     ), $args ) );
20
21     $feed = new DST_Feed();
22
23     return $feed->render( $args );
24 }
25
26 add_shortcode( 'dst', 'dst_shortcode' );
27
```

# Solution: Conditional Enqueueing of Assets/Data

- Enqueue the script only when the shortcode is actually rendered
- Enqueue it in the footer since rendering of the header has already passed

# Improved Code

```
1  <?php
2  add_action( 'wp_enqueue_scripts', 'jd_scripts' );
3  function jd_scripts() {
4
5      wp_register_script(
6          'shortcodescript',
7          '/dist/main.js',
8          array('jquery'),
9          false,
10         true
11     );
12 }
13
14 function jd_shortcode( $args ) {
15     wp_enqueue_script('shortcodescript');
16     extract( shortcode_atts( array(
17         'name'      => 'error',
18     ), $args ) );
19
20     $feed = new JD_Feed();
21
22     return $feed->render( $args );
23 }
24
25 add_shortcode( 'jd_s', 'jd_shortcode' );
26
```

# Use For

- Assets/Data you only need in certain circumstances
- Javascript Files you use in shortcodes if you have HTTP2 available
- Other data like big JSON data you will need in Javascript



# Code

```
1  <?php
2  function post_to_pdf_action() {
3      global $wpdb;
4      $exported = 0;
5      $post_ids = $_REQUEST['post'];
6      foreach( $post_ids as $post_id ) {
7          $posttitle = $wpdb->get_results("select * from $wpdb->posts where id='".$post_id.''",ARRAY_A);
8          require_once("dompdf/dompdf_config.inc.php");
9          $dompdf = new DOMPDF();
10         $html = "<h2>".$posttitle[0]['post_title']."</h2>".$posttitle[0]['post_content'];
11         $dompdf->load_html($html);
12         $dompdf->render();
13         $dompdf->stream($posttitle[0]['post_title'].".pdf");
14     }
15     exit();
16 }
17 add_action('save_post', 'post_to_pdf_action');
18
```

# Solution: **WP Asynchronous Tasks**

- WP Asynchronous Tasks is a simple PHP-Class that is developed by TechCrunch
- This Class uses a clever technique to trigger your actions via wp\_remote\_post and hence will not interfere with user facing views
- <https://github.com/techcrunch/wp-async-task>

# Improved Code

```
1  <?php
2  // Rest stays as before
3  add_action('wp_async_save_post', 'post_to_pdf_action');
4
5  class Post_PDF_Async_Task extends WP_Async_Task {
6
7      protected $action = 'save_post';
8
9      /**
10       *
11       * @param array $data An array of data sent to the hook
12       *
13       * @return array
14       */
15     protected function prepare_data( $data ) {
16         $post_id = $data[0];
17         return array( 'post_id' => $post_id );
18     }
19
20     /**
21      * Run the async task action
22      */
23     protected function run_action() {
24         $post_id = $_POST['post_id'];
25         $post = get_post( $post_id );
26         do_action( "wp_async_$this->action", $post->ID, $post );
27     }
28 }
29
```

# Use For

- Resource Heavy Tasks
- Downloading images
- Making API Calls
- Storing data in Transients for Caching Purpose

# Debugging the queries

- Query Monitor is a free plugin that shows you all queries that are made with a request
- You can profile individual plugins and their impact on performance
- <https://de.wordpress.org/plugins/query-monitor/>

```
0,93s 35.562KB 0,2518s 76Q
```

# Debugging the queries

Querys der Funktionen		
Abrufffunktion	SELECT	Zeit
		
WP_Term_Query->get_terms	25	0,2345
update_meta_cache	19	0,0068
WP_Post::get_instance	14	0,0034
WP_Query->get_posts	4	0,0029
get_option	6	0,0014
get_objects_in_term	1	0,0010
WP_Term::get_instance	1	0,0006
WC_Session_Handler->get_session	1	0,0003
WP_User::get_data_by	1	0,0003
WP_Query->set_found_posts	2	0,0002
WC_Data_Store_WP->read_meta	1	0,0002
_prime_post_caches	1	0,0002
Total	76	0,2518



# Questions?

# Slides



Scan the QR-Code

<https://github.com/pascalknecht/wordcampbern>