WordPress: integrating Bitly in our themes

Bitly provides a complete set of APIs which allows us to shorten our long URLS so that they can be easily shared across the web. Using Bitly in WordPress is quite easy: all you need is a Bitly account and an API token. Once you get this data, you can create the code required to shorten the permalinks of your WordPress posts.

The Bitly wrapper class

We need to keep our routines separated so we start with creating a simple wrapper class that makes a request to the Bitly APIs and return a response in JSON format.

Our class needs only the URL to be shortened as the parameter passed to its constructor. This URL will be in turn sent to Bitly with a GET request and the shortened URL will be extracted from the JSON output returned by the remote server.

```
class BitlyShortener {

        private $_token = 'your API token';
        private $_apiURL = 'https://api-
ssl.bitly.com/v3/shorten?';

        public $url;

        public function __construct( $uri ) {
                $this->url = $uri;
```

```
        }

        private function _buildURL() {
                $requestURL = $this->_apiURL .
'access_token=' . $this->_token . '&longUrl=' .
urlencode( $this->url ) . '&format=json';
                return $requestURL;
        }

        public function shorten() {
                $requestURI = $this->_buildURL();
                $ci = curl_init();

                curl_setopt( $ci, CURLOPT_USERAGENT,
'Bit.ly Client' );
                curl_setopt( $ci,
CURLOPT_CONNECTTIMEOUT, 30 );
                curl_setopt( $ci, CURLOPT_TIMEOUT, 30
);
                curl_setopt( $ci,
CURLOPT_RETURNTRANSFER, true );
                curl_setopt( $ci,
CURLOPT_SSL_VERIFYPEER, false );
                curl_setopt( $ci, CURLOPT_HEADER,
false );
                curl_setopt( $ci, CURLOPT_URL,
$requestURI );
                $response = curl_exec( $ci );



                curl_close( $ci );

                $data = json_decode( $response, true
```

```
);

                return $data['data']['url'];
        }

}
```

The theme function

Our wrapper class implements the necessary steps required to get the shortened URL from Bitly. What we need now is an utility function to be used in our theme that displays the shortened URL with the aid of the wrapper class.

The major problem with this solution is that we have to perform a remote request for every post in the Loop if we want to display an HTML link outside the `single.php` template.

To solve this issue, we can use custom fields to store the shortened URL and associate it with the current post. Our specialized function goes into the `functions.php` file.

You'll notice that also the full URL will be saved as a custom field: this allows us to address the problem of changes to the permalink structure.

```
<?php
require_once( TEMPLATEPATH .
'/inc/BitlyShortener.php' );

function my_shortener( $url ) {
        global $post;
        $post_id = $post->ID;
```

```
        $short = '';
        $link = get_post_meta( $post_id, 'bitly-
link', true );
        $original_link = get_post_meta( $post_id,
'original-link', true );
        if( !empty( $link ) ) {
                if( $url == $original_link ) {
                        $short = $link;
                } else {
                        $shortener = new
BitlyShortener( $url );
                        $shortened_link = $bitly-
>shorten();
                        update_post_meta( $post_id,
'bitly-link', $shortened_link );
                        update_post_meta( $post_id,
'original-link', $url );

                        $short = $shortened_link;
                }
        } else {
                $bitly = new BitlyShortener( $url );
                $short_link = $bitly->shorten();
                update_post_meta( $post_id, 'bitly-
link', $short_link );
                update_post_meta( $post_id,
'original-link', $url );
                $short = $short_link;
        }
        echo $short;
}
?>
```

Then we can use our function in the Loop:

```
<?php $permalink = get_permalink(); ?>
<a href="<?php my_shortener( $permalink ); ?>">Short
link</a>
```

This solution is pretty simple but effective.