Extending Press This in WordPress to support indieweb reply, like and repost

---

I've been able to reply/like/repost with my site for a long while, but is wasn't elegant, fast or slick at all; it was time to fix it.

The lazier I want to get the more code I need to write, but in the end, it always worth it. Pfefferle had pre-baked a solution already[^1] but my theme is special[^2] so I had to come up with my own solution.

Me vs sanity

For a long while I manually copy pasted URLs into an additional post meta field in my WordPress, which I called `webmention_url`. When I wanted to reply I:

1. copied the URL
2. went over to my admin area
3. opened a new post
4. copied the URL to a post meta field
5. selected webmention type from the radiobutton
6. posted the reply

This was tedious, smelled redundant, and introduced a lot more work.

To show the URLs I had to add it in my theme - it wasn't part of `the_content`, so it wasn't showing in the RSS or in the mail I'm sending out to some.

I had to hook that post meta field into the webmention plugin and do other magic on it, which made all the things even more complicated.

Bookmarklets

To solve the copy-paste part I turned to Press This[^3] - which I used a long, long time ago, and when I wanted to use it now, it didn't work. Because Firefox[^4].

To solve it:

```
about:config
security.csp.enable => false
```

This sacrificed some security, which I'm not proud of, but I'm tired of all the fresh things that break things we're actively using.

After this I realized, I need to pass the type of the webmention - so for reply, I added `&type=reply` at the end of the bookmarklet url, which now looks like this:

```javascript
javascript: (function(a, b, c, d) {
    function e(a, c) {
        if ("undefined" != typeof c) {
            var d = b.createElement("input");
            d.name = a, d.value = c, d.type = "hidden",
p.appendChild(d)
        }
    }
    var f, g, h, i, j, k, l, m, n, o = a.encodeURIComponent,
        p = b.createElement("form"),
        q = b.getElementsByTagName("head")[0],
        r = "_press_this_app",
        s = !0;
    if (d) {
        if (!c.match(/^https?:/)) return
void(top.location.href = d);
        if (d += "&u=" + o(c), c.match(/^https:/) && d.match(/
^http:/) && (s = !1), a.getSelection ? h = a.getSelection() +
"" : b.getSelection ? h = b.getSelection() + "" : b.selection
&& (h = b.selection.createRange().text || ""), d += "&buster="
+ (new Date).getTime(), s || (b.title && (d += "&t=" +
o(b.title.substr(0, 256))), h && (d += "&s=" + o(h.substr(0,
512)))), f = a.outerWidth || b.documentElement.clientWidth ||
```

```
600, g = a.outerHeight || b.documentElement.clientHeight ||
700, f = 800 > f || f > 5e3 ? 600 : .7 * f, g = 800 > g || g >
3e3 ? 700 : .9 * g, !s) return void a.open(d, r,
"location,resizable,scrollbars,width=" + f + ",height=" + g);
        i = q.getElementsByTagName("meta") || [];
        for (var t = 0; t < i.length && !(t > 200); t++) {
            var u = i[t],
                v = u.getAttribute("name"),
                w = u.getAttribute("property"),
                x = u.getAttribute("content");
            x && (v ? e("_meta[" + v + "]", x) : w &&
e("_meta[" + w + "]", x))
        }
        j = q.getElementsByTagName("link") || [];
        for (var y = 0; y < j.length && !(y >= 50); y++) {
            var z = j[y],
                A = z.getAttribute("rel");
            ("canonical" === A || "icon" === A || "shortlink"
=== A) && e("_links[" + A + "]", z.getAttribute("href"))
        }
        b.body.getElementsByClassName && (k =
b.body.getElementsByClassName("hfeed")[0]), k =
b.getElementById("content") || k || b.body, l =
k.getElementsByTagName("img") || [];
        for (var B = 0; B < l.length && !(B >= 100); B++) n =
l[B], n.src.indexOf("avatar") > -1 ||
n.className.indexOf("avatar") > -1 || n.width && n.width < 256
|| n.height && n.height < 128 || e("_images[]", n.src);
        m = b.body.getElementsByTagName("iframe") || [];
        for (var C = 0; C < m.length && !(C >= 50); C++)
e("_embeds[]", m[C].src);
        b.title && e("t", b.title), h && e("s", h),
p.setAttribute("method", "POST"), p.setAttribute("action", d),
p.setAttribute("target", r), p.setAttribute("style", "display:
none;"), a.open("about:blank", r,
"location,resizable,scrollbars,width=" + f + ",height=" + g),
b.body.appendChild(p), p.submit()
    }
})(window, document, top.location.href, "https:\/\/
example.com\/wp-admin\/press-this.php?v=8&type=reply");
```

The harder part was to catch this in PHP. The initial approach of filling the post meta required the post to be saved first, which made me trying to introduce 3 additional filters to Press This[^5] - 2 of them turned out to be redundant.

I used this setup for a little while, but it wasn't as good as I wanted it to. Then I came across how voxpelli[^6] is doing replies on Android:

https://www.youtube.com/watch?v=CBPmSpD2jN4

And it hit me: this is how I should be doing it! Inline, nicely integrated with my Markdown format and since I already extend the webmention/pingback URL list with the matches from the content, I wouldn't need to deal with that at all. No more special post meta which can't be accessed with the Android WordPress app either.

The technical gore

I have to give credit to http://regex101.com/ again for helping creating that monster regex.

```php
<?php
add_filter( 'press_this_data', 'cleanup_press_this_data', 9,
1 );
add_filter( 'press_this_suggested_html',
'cleanup_press_this_suggested', 2, 2 );
add_filter('enable_press_this_media_discovery',
'__return_false' );
function cleanup_press_this_data ( $data ) {
    if ( isset( $data['s'] ) && ! empty( $data['s'] ))
        // do magic here; I try to make Markdown from HTML,
for example
        $data['s'] = some_parser_function( $data['s'] );
    return $data;
}
function cleanup_press_this_suggested ( $default_html, $data )
{
    $ref = array();
    $relation = '';
    parse_str ( parse_url( $_SERVER['REQUEST_URI'],
PHP_URL_QUERY ), $ref );
    if ( is_array( $ref ) && isset ( $ref['u'] ) && !
empty( $ref['u'] ) ) {
        $url = $ref['u'];
        $t = '';
        if ( isset( $ref['type'] ) )
            $t = $ref['type'];
        switch ( $t ) {
            case 'fav':
            case 'like':
            case 'u-like-of':
                $type = 'like: ';
                break;
            case 'repost':
                $type = 'from: ';
                break;
```

```
            case 'reply':
                $type = 're: ';
                break;
            default:
                $type = '';
                break;
            }
            $relation = "---\n{$type}{$url}\n---\n\n";
        }
        $default_html = array (
            'quote' => '> %1$s',
            'link' => '',
            'embed' => $relation,
        );
        return $default_html;
    }
```

Presenting this is a little trickier; this is what I use:

```php
<?php
add_filter( 'the_content', 'convert_reaction', 1 );
function convert_reaction ( $content ) {
    $pattern = "/---[\n\r]+(?:(.*?):\s+)?+\b((?:http|https)\:\/
\/?[a-zA-Z0-9\.\/\?\:@\-_=#]+\.[a-zA-Z0-9\.\/\?\:@\-_=#&]*)(?:
[\n\r]+((?!---).*))?[\n\r]+---/mi";
    $matches = array();
    preg_match_all( $pattern, $content, $matches);
    if ( empty( $matches ) || ! isset( $matches[0] ) ||
empty( $matches[0] ) )
        return $content;
    $replace = false;
    $r = false;
    $type = false;
    $rsvp = '';
    $rsvps = array (
        'no' => __("Sorry, can't make it."),
        'yes' => __("I'll be there."),
        'maybe' => __("I'll do my best, but don't count on me
for sure."),
    );
    $replace = $matches[0][0];
```

```
    $type = trim($matches[1][0]);
    $url = trim($matches[2][0]);
    $data = trim($matches[3][0]);
    if ( $type == 're' && !empty( $data ) )
        $rsvp = '<data class="p-rsvp" value="' . $rsvp .'">'.
$rsvps[ $rsvp ] .'</data>';
    switch ($type) {
        case 'like':
        case 'fav':
            $cl = 'u-like-of';
            $prefix = '';
            break;
        case 'from':
        case 'repost':
            $cl = 'u-repost-of';
            $prefix = '*reposted from:* ';
            break;
        case 're':
            $cl = 'u-in-reply-to';
            $prefix = '**RE:** ';
            break;
        default:
            $cl = 'u-url';
            $prefix = '**URL:** ';
            break;
    }
    $title = str_replace ( parse_url( $url,
PHP_URL_SCHEME) .'://', '', $url);
    $r = "\n{$prefix}[{$title}]({$url}){.{$cl}}\n{$rsvp}";
    return str_replace ( $replace, $r, $content );
}
```

Adding Android

Thanks to the long and detailed description of Chris Aldrich[^7] I added this functionality to Android as well, with URL Forward[^8] as: `https://example.com/wp-admin/press-this.php?v=8&type=reply&u=@url`

Footnotes

Automating thins is good for you.

Links

1. https://github.com/pfefferle/wordpress-indieweb-press-this
2. http://satwcomic.com/coat-of-arms
3. http://codex.wordpress.org/Press_This
4. https://github.com/blog/1477-content-security-policy#bookmarklets
5. https://core.trac.wordpress.org/ticket/34455
6. http://voxpelli.com/
7. http://stream.boffosocko.com/2016/sharing-from-the-indieweb-on-mobile-android-with-apps-and
8. https://play.google.com/store/apps/details?id=net.daverix.urlforward

Created by Peter Molnar <mail@petermolnar.net>, published at 2016-03-10 16:00 UTC, last modified at 2021-05-11 11:49 UTC , to canonical URL https://petermolnar.net/article/press-this-indieweb/ , licensed under CC-BY-4.0 .