Please separate PHP and Javascript | Tiki Wiki CMS Groupware :: Development
Please separate PHP and Javascript

Status
● Open

Subject
Please separate PHP and Javascript

Version
18.x

Category
- Feature request
- Consistency

Feature
i18n (Multilingual, l10n, Babelfish)

Resolution status
New

Submitted by
hman

Lastmod by
hman

Rating
★★★★★ (0)

Related-to
- ● Please update German localization
- ● Create a new language for Tiki, "generic"

Description
While searching for a string that I did not find in Tiki's source code, I came across /lib/wiki-plugins/wikiplugin_registermemberpayment.php. It does contain Javascript! And quite a lot. Almost half of the file is after the statement

```
$headerlib->add_jq_onready(<<<JS
```

My request for the devs would be: If you have full-fledged Javascript programs, please separate them off into their own .js files. The reason is that PHP's tokenizer will not parse Javascript behind a <<< statement, which is token T_START_HEREDOC (which is eventually ended in a T_END_HEREDOC), and a call to tr() will be inside a huge block of T_ENCAPSED_AND_WHITESPACE, and not in one T_STRING (with the string to be translated in a T_CONSTANT_ENCAPSED_STRING)...

Thanks from your fellow translator
hman

Importance

5

Easy to solve?

5

Priority

25

Demonstrate Bug (Tiki 19+)

Please demonstrate your bug on show2.tiki.org

Version: trunk ▼

Demonstrate Bug (older Tiki versions)

Please demonstrate your bug on show.tikiwiki.org

Version: 18.x ▼

Ticket ID

7731

Created

Wednesday 21 April, 2021 21:08:40 GMT-0000
by hman

LastModif

Monday 26 April, 2021 19:11:42 GMT-0000

Comments

hman 22 Apr 21 19:36 GMT-0000

Not only that sometimes this kind of "hidden" Javascript contains text to be translated in tr(), sometime you find things that look strange to me like this:

/lib/toolbars/toolbarslib.php line 1531:

```
if (data.data.added.length === 1 && confirm(tr(\'Do you want to use this file in your page?\'))) {
```

A PHP function call to tr() with escaped single quote delimiters? When you open that in gedit, the auto-coloring shows that this syntax is even beyond the scope of gedit's PHP syntax highlighting profile... Everything pink (data).

This is because all this is supposed to be part of the return statement in line 1519, which yields a loooong statement starting with a single quote, so technically this is correctly escaped (the entire return argument runs from line 1519 all the way to 1536), but I doubt that it works, and I am puzzled how this string could be automatically extracted.

At least my tokenizer-based approach can't, and I don't think Tikis own mechanism does catch it, as not one single language.php contains that string...

hman 23 Apr 21 10:26 GMT-0000

I wrote a small tool that will crawl throught the entirety of a Tiki directory tree (or a copy thereof), and analyze the code (with the use of PHP's built-in tokenizer) to find heredoc, and extract that into a new file with added 'NUM.js' to it's filename, with NUM being the number of the extract, starting from 1.

Because this can produce large numbers of new files, a script for easy deletion is also created (Linux only). Also, all content of the heredoc is output to stdout, making it easy to grep through the entirety of heredoc with one single grep (note: in order to find tr(), first grep for "tr(", then grep -v for "attr(" to reduce noise. All output is stripslashed, because heredocs are natively double quoted strings.

This extraction is useful for many purposes. For me, it enables me to use my tra() extraction tool, that can deal with PHP and Javascript, but only if Javascript is inside a file with .js extension, but could also be useful for easier syntax checking of the Javasript code (note: Variables will get expanded only in runtime!)

hman 23 Apr 21 10:30 GMT-0000

I see now that Javascript in heredoc is vastly used in Tiki, so my wish for separation of Tiki PHP and Javascript code can only be a wish for future development, but I'll let this stand anyhow, as a feature request.

hman 23 Apr 21 11:27 GMT-0000

This analysis leads to a LARGE number of new strings that need to be translated. With just the nine test .PHP files from Tiki 18.8 I picked, I gathered two dozen (!) Javascripts in heredocs, containing almost five dozen new strings that need translation (!), all of them have to be added to language.js... Which means I have to adjust the language.js for German (de.de) as well as my "generic" language.js.

hman 23 Apr 21 12:27 GMT-0000

Uploaded the tra() extract from the heredoc extract of my nine test files, to illustrate to you how many "hidden" translation strings reside inside heredocs...

This is in the syntax of my tra_extract2, with a comment giving relative file path, then strings found and a generic "translation" that is constructed from the file name and the line number. The file names here are, of course, referring to the created extract files. The filename of an extract file is identical to the base PHP file it was generated from, plus the counter of the heredoc and extension .js (So my tra_extractor can deal with it like a "regular" Javascript file).

hman 26 Apr 21 11:24 GMT-0000

I ran my heredoc_extractor on the entirety of Tiki 18.8. Tiki contains 111 heredocs, as you can see from the removal script that heredoc extractor creates for easy cleanup. Not all heredocs are really Javascript, but very many. Some just contain HTML. heredoc_extractor names all extracted heredoc .js, so that tra_extractor will treat them as Javascript. Should they contain HTML, they will not contain tra() or tr() - a corner case that can be safely ignored.

hman 26 Apr 21 11:34 GMT-0000

And running tra_extractor2 shows that with the Javascipt tra out of the heredocs, language.js will grow considerably to almost 300 lines (before deduplication)... 213 after deduplication.

hman 26 Apr 21 19:07 GMT-0000

Merging the deduplicated output with my RC 6 of the German (de.de) language.js shows that solely from the heredocs (!), 35 new strings appeared, that as of today are untranslated. So I'll make a RC 7 of that, which I will publish on the "Please update German localization" feature request.

hman 03 May 21 18:55 GMT-0000

Oh my goodness. As if heredocs wouldn't be enough to drive a translator crazy, I found yet another source of javascript:

$headerlib->add_jq_onready() with a string as an argument. Technically, that is something completely different than a heredoc, although it looks like it does the same.

Technically it is different, because PHPs tokenizer deals with the whole string as one token (T_STRING), and not a heredoc between T_START_HEREDOC and T_END_HEREDOC, which leads to my heredoc extractor not catching this. Not a big deal, but I have to add yet another subroutine for a corner case.

Found in /tiki-slideshow.php and /tiki-timesheet.php.

Although, a quick grep through Tiki 18.8 reveals there are dozens of Javascripts "hidden" this way, and this means that there are more tr() calls inside. I already found 4 new strings that add to language.js, so I have to move my RC up to 7.2. The 4 are easily added manually, but I need the new subroutine to automatically create comprehensive Javascript extracts...

hman 04 May 21 15:47 GMT-0000

To be more precise, it's a T_CONSTANT_ENCAPSED_STRING, this is how PHP's tokenizer sees the start of the Javascript in /tiki-slideshow.php:

Line 161: T_VARIABLE ('$headerlib') Line 161: T_OBJECT_OPERATOR ('->') Line 161: T_STRING ('add_jq_onready') Line 161: T_WHITESPACE (' ') Line 162: T_CONSTANT_ENCAPSED_STRING (''$("#toc").remove(); window.s5Settings = (window.s5Settings ? window.s5Settings : {}); [and then all the rest, all part of the one token encapsed string]

So this should be detectable, und then extractable.

hman 07 May 21 16:52 GMT-0000

Actually, I wonder whether the Javascript injection in tiki-slideshow really works... Because when I looked closer at the output of the tokenizer, I found that PHP's tokenizer really cannot determine the proper end of the injected Javascript. A heredoc ends with a clear T_END_HEREDOC, but there is no such thing with add_jq_onready(). The syntax highlighter of Linux' gedit seems to have no problem, but a look at the tokenizer from PHP shows that the last line, the one the contains the closing parenthesis of the function call to add_jq_onready() is just a T_WHITESPACE, one of many. And since the tokenizer strips the parenthesis and the semicolon, there is no way to actually determine the end of the Javascript.

Therefore I wrote something of a hack, that tries to find something typical PHP that you won't find in Javascript, like the arrow operator ->. That is readily detected by the tokenizer ("T_OBJECT_OPERATOR"). Abusing this as a stop signal to stop Javascript extraction has the unclean side effect that the variable stated right before the arrow operator will also be extracted. But since the goal of the extractor is not to produce 100 % pure Javascript, but produce something that the tra() extractor can feast on, this little clutter is tolerable, IMHO.

hman 07 May 21 16:55 GMT-0000

Operating like this, only on the two test PHPs mentioned above, four new strings for translation could be identified, that must be contained in every languages' language.js, if these are to be deemed complete. They are:

/* Extracted tra() calls by tra_extract2.php V2.8 20210506 by hman */ // tiki-slideshow.php1.js "Updating Theme..." : "## tiki-slideshow.php1.js:57", // tiki-timesheet.php1.js "Local Cache (Not Committed)" : "## tiki-timesheet.php1.js:5", "Committing..." : "## tiki-timesheet.php1.js:58", "Could not save" : "## tiki-timesheet.php1.js:84"

And I doubt any other automated function catches those, at the moment.

hman 07 May 21 20:46 GMT-0000

add_js() is also being used to inject Javascript, similar syntax. Added another extraction routine to the

heredoc extractor, but so far (in the two test PHPs) no new translation strings. Possibly after applying the extractor to the entirety of Tiki 18.8.

hman 08 May 21 08:08 GMT-0000

The updated heredoc extractor now finds significantly more Javascript etc. in Tiki 18.8: The count rises to 345... Now it pays off that my extractor writes a delete script to make cleaning up easy... I'll upload that to show how many (and where) Javascripts were found. Then let us see how many new strings for translation are discovered.

hman 08 May 21 08:29 GMT-0000

Okay, back to the drawing board. Future UI code has some "peculiarities" like combining add_jq_onready() with heredoc into one statement, a corner case I didn't foresee: /lib/core/FutureLink/FutureUI.php, line 74:

```
->add_jq_onready( <<<JQ function trackerForm(trackerId, itemId, tracker_fn_name, type, fn) { ...
```

hman 08 May 21 09:00 GMT-0000

Correction was easy, the heredoc inside add_qd_onready() is caught by the tokenizer, so I just had to switch extraction subroutine after detecting a T_START_HEREDOC inside add_qd_onready() to there heredoc routine. Count is now 324 scripts in Tiki 18.8 yielding 220 genuine unique translation strings in Javascript.

Attachments

	filename	created	hits	comment	version	filetype
	tra_extract_raw.js.txt	26 Apr 21 11:37 GMT-0000	0	Complete extraction of Javascript tra, including those in heredocs. Taken from Tiki 18.8.		
	tra_dedup.js.txt	26 Apr 21 12:01 GMT-0000	0	Dedup'd extraction of Javascript tra, including those in heredocs. Taken from Tiki 18.8.		

	heredoc_extract.php	23 Apr 21 10:28 GMT-0000	0	Heredoc extract v 1.0, a tool to extract here doc from files in a directory tree	
	tra_extract_raw.js.txt	23 Apr 21 12:31 GMT-0000	0	Sample tra() extract from the extract from the nine demo files of Tiki 18.8 I used for testing	
	tra_extract_raw.js.txt	23 Apr 21 18:52 GMT-0000	0	Sample tra() extract from the extract from the nine demo files of Tiki 18.8 I used for testing	
	heredoc_extract_deletescript.sh	08 May 21 09:06 GMT-0000	0	Updated Heredoc extractor delete script for easy cleanup, but can also serve as a log file...	

The original document is available at https://dev.tiki.org/item7731-Please-separate-PHP-and-Javascript