Rocky Enterprise Linux 9.2 Manual Pages on command 'Eval::TypeTiny.3pm'

C:\>man Eval::TypeTiny.3pm

NAME

Eval::TypeTiny - utility to evaluate a string of Perl code in a clean environment

STATUS

This module is covered by the Type-Tiny stability policy.

DESCRIPTION

This module is used by Type::Tiny to compile coderefs from strings of Perl code,

and hashrefs of variables to close over.

Functions

This module exports one function, which works much like the similarly named

function from Eval::Closure:

"eval_closure(source => $source, environment => \%env, %opt)"

Constants

The following constants may be exported, but are not by default.

"HAS_LEXICAL_SUBS"

Boolean indicating whether Eval::TypeTiny has support for lexical subs. (This feature requires Perl 5.18.)

"ALIAS_IMPLEMENTATION"

Returns a string indicating what implementation of "alias => 1" is being used. Eval::TypeTiny will automatically choose the best implementation. This constant can be matched against the "IMPLEMENTAION_*" constants.

"IMPLEMENTATION_NATIVE"

If "ALIAS_IMPLEMENTATION eq IMPLEMENTATION_NATIVE" then Eval::TypeTiny is currently using Perl 5.22's native alias feature. This requires Perl 5.22.

"IMPLEMENTATION_DEVEL_LEXALIAS"

If "ALIAS_IMPLEMENTATION eq IMPLEMENTATION_DEVEL_LEXALIAS" then Eval::TypeTiny is currently using Devel::LexAlias to provide aliases.

"IMPLEMENTATION_PADWALKER"

If "ALIAS_IMPLEMENTATION eq IMPLEMENTATION_PADWALKER" then Eval::TypeTiny is currently using PadWalker to provide aliases.

"IMPLEMENTATION_TIE"

If "ALIAS_IMPLEMENTATION eq IMPLEMENTATION_TIE" then Eval::TypeTiny is using the fallback implementation of aliases using "tie". This is the slowest implementation, and may cause problems in certain edge cases, like trying to alias already-tied variables, but it's the only way to implement "alias => 1" without a recent version of Perl or one of the two optional modules mentioned above.

EVALUATION ENVIRONMENT

The evaluation is performed in the presence of strict, but the absence of warnings. (This is different to Eval::Closure which enables warnings for compiled closures.)

The feature pragma is not active in the evaluation environment, so the following

will not work:

```
use feature qw(say);
use Eval::TypeTiny qw(eval_closure);

my $say_all = eval_closure(
  source => 'sub { say for @_ }',
);
$say_all->("Hello", "World");
```

The feature pragma does not "carry over" into the stringy eval. It is of course possible to import pragmas into the evaluated string as part of the string itself:

```
use Eval::TypeTiny qw(eval_closure);

my $say_all = eval_closure(
  source => 'sub { use feature qw(say); say for @_ }',
);
$say_all->("Hello", "World");
```

BUGS

Please report any bugs to <http://rt.cpan.org/Dist/Display.html?Queue=Type-Tiny>.

SEE ALSO

Eval::Closure, Error::TypeTiny::Compilation.

AUTHOR

Toby Inkster <tobyink@cpan.org>.

COPYRIGHT AND LICENCE

terms as the Perl 5 programming language system itself.

DISCLAIMER OF WARRANTIES

THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.