Linux Ubuntu 22.4.5 Manual Pages on command 're::engine::RE2.3pm'

$ man re::engine::RE2.3pm

re::engine::RE2(3pm) User Contributed Perl Documentation re::engine::RE2(3pm)

NAME

re::engine::RE2 - RE2 regex engine

SYNOPSIS

```
use re::engine::RE2;
if ("Hello, world" =~ /Hello, (world)/) {
    print "Greetings, $1!";
}
```

DESCRIPTION

This module replaces perl's regex engine in a given lexical scope with RE2.

RE2 is a primarily DFA based regexp engine from Google that is very fast at matching large amounts of text. However it does not support look behind and some other Perl regular expression features. See RE2's website <http://code.google.com/p/re2> for more information.

Fallback to normal Perl regexp is implemented by this module. If RE2 is unable to compile a regexp it will use Perl instead, therefore features not implemented by RE2 don't suddenly stop working, they will just use Perl's regexp implementation.

METHODS

To access extra functionality of RE2 methods can be called on a compiled regular expression (i.e. a "qr//").

? "possible_match_range([length = 10])"

Returns an array of two strings: where the expression will start matching and

just after where it will finish matching. See RE2's documentation on

PossibleMatchRange for further details.

Example:

```
my($min, $max) = qr/^(a|b)/->possible_match_range;
is $min, 'a';
is $max, 'c';'
```

PRAGMA OPTIONS

Various options can be set by providing options to the "use" line. These will be pragma scoped.

? "-max_mem => 1<<24"

Configure RE2's memory limit.

? "-strict => 1"

Be strict, i.e. don't allow regexps that are not supported by RE2.

? "-longest_match => 1"

Match on the longest match in alternations. For example with this option set matching "abc" against "(a|abc)" will match "abc", without depending on order.

? "-never_nl => 1"

Never match a newline ("\n") even if the provided regexp contains it.

PERFORMANCE

Performance is really the primary reason for using RE2, so here's some benchmarks.

Like any benchmark take them with a pinch of salt.

Simple matching

```
my $foo = "foo bar baz";
$foo =~ /foo/;
$foo =~ /foox/;
```

On this very simple match RE2 is actually slower:

```
        Rate  re2   re
re2  674634/s   -- -76%
re  2765739/s 310%   --
```

URL matching

Matching "m{([a-zA-Z][a-zA-Z0-9]*)://([^ /]+)(/[^]*)?|([^ @]+)@([^ @]+)}" against a several KB file:

```
      Rate    re   re2
```

```
re  35.2/s    --  -99%
re2 2511/s 7037%    --
```

Many alternatives

Matching a string against a regexp with 17,576 alternatives ("aaa .. zzz").

This uses trie matching on Perl (obviously RE2 does similar by default).

```
$ perl misc/altern.pl
        Rate   re  re2
re   52631/s   -- -91%
re2 554938/s 954%   --
```

NOTES

? No support for "m//x"

The "/x" modifier is not supported. (There's no particular reason for this, just RE2 itself doesn't support it). Fallback to Perl regexp will happen automatically if "//x" is used.

? "re2/dfa.cc:447: DFA out of memory: prog size xxx mem yyy"

If you attempt to compile a really large regular expression you may get this error. RE2 has an internal limit on memory consumption for the DFA state tables. By default this is 8 MiB.

If you need to increase this size then use the max_mem parameter:

```
use re::engine::RE2 -max_mem => 8<<23; # 64MiB
```

? How do I tell if RE2 will be used?

See if your regexp is matching quickly or slowly ;).

Alternatively normal OO concepts apply and you may examine the object returned by "qr//":

```
use re::engine::RE2;
ok qr/foo/->isa("re::engine::RE2");
# Perl Regexp used instead
ok not qr/(?<=foo)bar/->isa("re::engine::RE2");
```

If you wish to force RE2, use the "-strict" option.

BUGS

Known issues:

? Unicode handling

Currently the Unicode handling of re::engine::RE2 does not fully match Perl's

behaviour.

The UTF-8 flag of the regexp currently determines how the string is matched.

This is obviously broken, so will be fixed at some point.

? Final newline matching differs to Perl

"\n" =~ /$/

The above is true in Perl, false in RE2. To work around the issue you can write "\n?\z" when you mean Perl's "$".

Please report bugs via RT in the normal way. (Or a patch at <https://github.com/dgl/re-engine-RE2> would be most welcome.)

AUTHORS

David Leadbeater <dgl[at]dgl[dot]cx>

COPYRIGHT