Better way to do Regexp.new(Regexp.escape("some string"))

It is relatively common to call Regexp.new(Regexp.escape("some string")) or Regexp.new(Regexp.quote("some string")) in Ruby (a quick search turns up over half a million matches in Github). However, these seems rather cumbersome. It would be nice to have a class instance method on Regexp and/or an instance method on String to go directly to an escaped regular expression.

Personally I'd rather see #to_regex added to String. This would not overlap with the functionality that Rexexp.new("string") provides since that doesn't quote the String.

It would also be good to see #9846 addressed along with this request so one could more easily duck type a method to accept both Regexp and String arguments.

Thoughts?

History

#1 - 12/18/2016 02:23 AM - nobu (Nobuyoshi Nakada)
A known hack, Regexp.union("some string").

#2 - 12/21/2016 07:17 PM - justcolin (Colin Fulton)
It would be nice to have a real/more-idiomatic way instead of a hack, and that hack doesn't support duck typing, while adding #to_regex to String does.

#3 - 01/20/2017 04:01 AM - shyouhei (Shyouhei Urabe)
We looked at this issue at yesterday's developer meeting.

While looking at "Regexp.new Regexp.escape" search result of GitHub, we felt in common that some (if not most) of those usage are questionable. For instance multiple copies of String#to_regex implementations specify Regexp::IGNORECASE along with the escaped text; which does not immediately make sense to me. There are other use cases in tests. Because they are written in their own DSLs, why not have custom matcher that escape strings appropriately.

Nobody was against adding this functionality but I think more closer look at use cases would give us better insight of the needs.