Redmine - Feature #35676

Optimize performance of syntax highlighting implementation

2021-08-03 12:11 - Mischa The Evil

Status:	Needs feedback	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Performance	**Estimated time:**	0.00 hour
Target version:			
Resolution:			

Description

- re-implement performance optimizations of #26055
- don't send input through syntax highlighting if it:
  - does not need to be highlighted, or
  - can not be highlighted

Related issues:

Related to Redmine - Defect #26055: Three issues with Redmine::SyntaxHighligh...	**Closed**
Related to Redmine - Feature #24681: Syntax highlighter: replace CodeRay with...	**Closed**

History

#1 - 2021-08-04 17:38 - Mischa The Evil

- Related to Defect #26055: Three issues with Redmine::SyntaxHighlighting::CodeRay.language_supported? added

#2 - 2021-08-04 17:38 - Mischa The Evil

- Related to Feature #24681: Syntax highlighter: replace CodeRay with Rouge added

#3 - 2022-07-30 09:11 - Go MAEDA

Mischa The Evil wrote:

> - re-implement performance optimizations of #26055

I wrote another version of Redmine::SyntaxHighlighting::Rouge.language_supported? that implements the optimization of #26055, but I found that the current implementation is twice faster.

The code that implements the optimizations of #26055:

```
diff --git a/lib/redmine/syntax_highlighting.rb b/lib/redmine/syntax_highlighting.rb
index 731c4eaec..9424eec03 100644
--- a/lib/redmine/syntax_highlighting.rb
+++ b/lib/redmine/syntax_highlighting.rb
@@ -82,6 +82,13 @@ module Redmine
       end

       class << self
+        def self.retrieve_supported_languages
+          (::Rouge::Lexer.all.map(&:tag) + ::Rouge::Lexer.all.map(&:aliases).flatten).uniq.map(&:downcase)
+        end
+        private_class_method :retrieve_supported_languages
+
+        SUPPORTED_LANGUAGES = retrieve_supported_languages.sort
+
         # Highlights +text+ as the content of +filename+
         # Should not return line numbers nor outer pre tag
         def highlight_by_filename(text, filename)
@@ -109,6 +116,10 @@ module Redmine
           find_lexer(language.to_s.downcase) ? true : false
         end

+        def language_supported_const?(language)
```

```
+           SUPPORTED_LANGUAGES.bsearch {|e| language <=> e}
+         end
+
          def filename_supported?(filename)
            !::Rouge::Lexer.guesses(:filename => filename).empty?
          end
```

Benchmark script:

```
require 'benchmark/ips'

languages = %w(abap lua zsh)

Benchmark.ips do |bench|
  bench.report('language_supported?') do
    Redmine::SyntaxHighlighting::Rouge.language_supported?(languages.sample)
  end

  bench.report('language_supported_const?') do
    Redmine::SyntaxHighlighting::Rouge.language_supported_const?(languages.sample)
  end

  bench.compare!
end
```

Result:

```
Warming up --------------------------------------
 language_supported?   244.405k i/100ms
language_supported_const?
                       122.629k i/100ms
Calculating -------------------------------------
 language_supported?      2.445M (± 2.3%) i/s -     12.465M in   5.100127s
language_supported_const?
                         1.215M (± 2.5%) i/s -      6.131M in   5.050739s

Comparison:
 language_supported?:  2445257.6 i/s
language_supported_const?:  1214719.1 i/s - 2.01x  (± 0.00) slower
```

#4 - 2024-03-21 18:05 - Holger Just

- Status changed from New to Needs feedback

Rather than a bsearch over a sorted array, you can also use a Set (which uses a hash underneath). Adapting your patch from #note-3, it could be implemented as such:

```
diff --git a/lib/redmine/syntax_highlighting.rb b/lib/redmine/syntax_highlighting.rb
index 731c4eaec..9424eec03 100644
--- a/lib/redmine/syntax_highlighting.rb
+++ b/lib/redmine/syntax_highlighting.rb
@@ -82,6 +82,13 @@ module Redmine
       end

       class << self
+        def self.retrieve_supported_languages
+          (::Rouge::Lexer.all.map(&:tag) + ::Rouge::Lexer.all.map(&:aliases).flatten).uniq.map(&:downcase)
+        end
+        private_class_method :retrieve_supported_languages
+
+        SUPPORTED_LANGUAGES = Set.new(retrieve_supported_languages).freeze
+
         # Highlights +text+ as the content of +filename+
         # Should not return line numbers nor outer pre tag
         def highlight_by_filename(text, filename)
@@ -109,6 +116,10 @@ module Redmine
           find_lexer(language.to_s.downcase) ? true : false
         end

+        def language_supported_const?(language)
+          SUPPORTED_LANGUAGES.include?(language)
+        end
+
```

```
        def filename_supported?(filename)
          !::Rouge::Lexer.guesses(:filename => filename).empty?
        end
```

This version is much faster than your Array#bsearch version (about 4.5 times in fact) and about 1.4 times faster than the current code. The performance is roughly expected, as rouge also performs just a single Hash access to find a language.

Still, as the current implementation with Rouge is already much faster than what we had with Coderay, I'm not sure if this change is worth it at all.

As for the other issues mentioned by Mischa, namely:

- don't send input through syntax highlighting if it:
  - does not need to be highlighted, or
  - can not be highlighted

I believe this is solved already in the current code.

All call-sites check for Redmine::SyntaxHighlighting.language_supported? before calling Redmine::SyntaxHighlighting.highlight_by_language.

The filename based highlighting can't be easily improved here as it uses Rouge's builtin features to guess a suitable lexer based on the filename (or falls back to a no-op plaintext lexer).

As such, I believe there is nothing remaining that we would need to do here. What do you think?