Redmine - Feature #24681

Syntax highlighter: replace CodeRay with Rouge

2016-12-26 02:15 - Go MAEDA

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Jean-Philippe Lang	**% Done:**	0%
Category:	Text formatting	**Estimated time:**	0.00 hour
Target version:	4.0.0		
Resolution:	Fixed		

Description

I propose replacing CodeRay with other syntax highlighter, Rouge . It supports 100+ languages.

Current syntax highlighter CodeRay does not support popular languages such as C#, Visual Basic, Objective C, Swift, ... and so on.
Unfortunately the development of CodeRay is not so active now, it is difficult to expect that CodeRay will support those languages.

Citation from Rouge's README.md :

> **Advantages to CodeRay**
>
> - The HTML output from Rouge is fully compatible with stylesheets designed for pygments.
> - The lexers are implemented with a dedicated DSL, rather than being hand-coded.
> - Rouge supports every language CodeRay does and more.

Related issues:

Related to Redmine - Feature #2623: C# syntax highlighting	**Closed**	**2009-01-30**
Related to Redmine - Feature #3032: Use google Prettify for syntax highlighti...	**Closed**	**2009-03-23**
Related to Redmine - Feature #1313: Optionally use ultraviolet for syntax hig...	**Closed**	**2008-05-27**
Related to Redmine - Patch #1651: Hack to make redmine use pygmentize instead...	**Closed**	**2008-07-15**
Related to Redmine - Feature #28094: Kotlin code highlight support	**Closed**	
Related to Redmine - Defect #29259: Attachment preview does not work for some...	**Closed**	
Related to Redmine - Defect #29681: "x%x%" is rendered as "&" in Textile form...	**New**	
Related to Redmine - Defect #26708: Diff formatting results empty lines if th...	**Closed**	
Related to Redmine - Defect #20758: Ampersand+keyword in code highlighting	**Closed**	
Related to Redmine - Defect #30434: Line height is too large when previewing ...	**Closed**	
Related to Redmine - Feature #35676: Optimize performance of syntax highlight...	**Needs feedback**	

Associated revisions

Revision 17532 - 2018-09-29 08:57 - Jean-Philippe Lang

Syntax highlighter: replace CodeRay with Rouge (#24681).

Patch by Go MAEDA.

Revision 17533 - 2018-09-29 11:06 - Jean-Philippe Lang

Update the list with popular languages (#24681).

Revision 17536 - 2018-09-30 04:45 - Go MAEDA

Updated the "Code highlighting" section in wiki help as the syntax highlighter was replaced with Rouge (#24681).

Revision 17539 - 2018-09-30 11:32 - Jean-Philippe Lang

Update the style of JQuery dropdown menu (#24681).

Revision 17545 - 2018-09-30 17:51 - Go MAEDA

Added the list of supported languages on redmine.org (#24681).

Revision 17626 - 2018-11-10 07:50 - Go MAEDA

Upgrade to Rouge 3.3.0 (#24681).

History

#1 - 2016-12-26 02:16 - Go MAEDA

- Related to Feature #2623: C# syntax highlighting added

#2 - 2016-12-26 02:17 - Go MAEDA

- Related to Feature #3032: Use google Prettify for syntax highlighting instead of CodeRay added

#3 - 2016-12-26 02:17 - Go MAEDA

- Related to Feature #1313: Optionally use ultraviolet for syntax highlighting added

#4 - 2016-12-26 02:18 - Go MAEDA

- Related to Patch #1651: Hack to make redmine use pygmentize instead of CodeRay added

#5 - 2016-12-26 04:38 - Mischa The Evil

Just for the information: see also the redmine_rouge (https://github.com/ngyuki/redmine_rouge) plugin.

#6 - 2016-12-26 05:31 - Go MAEDA

- File redmine_rouge_plugin_csharp.png added

Mischa The Evil wrote:

> Just for the information: see also the redmine_rouge (https://github.com/ngyuki/redmine_rouge) plugin.

Thanks for the information. The plugin works fine on the current trunk (r16111).
It seems that we can implement this feature in a small amount of code.

Updated by Redmine Admin less than a minute ago

```
//Hello World in C#
class HelloWorld
{
    static void Main()
    {
        System.Console.WriteLine("Hello, World!");
    }
}
```

#7 - 2016-12-28 07:48 - Go MAEDA

- File 0001-Replace-syntax-highlighter-CodeRay-with-Rouge.patch added

- File highlight-sample.png added

- Subject changed from Syntax highlighter: replace CodeRay with rouge to Syntax highlighter: replace CodeRay with Rouge

- Description updated

This is a patch to replace CodeRay with Rouge.

With this patch applied, we can highlight 100+ languages including C# (csharp), Visual Basic (vb), Objective-C (objective_c), Swift (swift) and Perl (perl).

Users can use all language classes (code class="XXX") currently supported by CodeRay except for taskpaper.

Since the supported language dramatically increases from 24 to 100+, I think that the merit of this patch for users is very large.

Visual Basic:

```
' Allow easy reference to the System namespace classes.
Imports System

' This module houses the application's entry point.
Public Module modmain
   ' Main is the application's entry point.
   Sub Main()
     ' Write text to the console.
     Console.WriteLine ("Hello World using Visual Basic!")
   End Sub
End Module
```

Java: (already supported by CodeRay)

```
public class Hello {
  public static void main(String[] args) {
    System.out.println("Hello, world!");
  }
}
```

#8 - 2016-12-28 10:41 - Marius BĂLTEANU

- File with_open_tag.png added

- File without_opening_tag.png added

I tested the patch and I've the following observations:

1. We should rename the "codeRay" variable in source:trunk/public/javascripts/jstoolbar/jstoolbar.js#L378
2. Rename the "CodeRay" from source:trunk/public/stylesheets/rtl.css#L375
3. Update help documentation: source:trunk/public/help/en/wiki_syntax_detailed_markdown.html#L300

Also, the Rogue library seems to have an issue with the PHP language which is highlighted only when the opening tag is present.

Without open tag:

```
$txt = "Hello world!";
$x = 5;
$y = 10.5;

echo $txt;
```

With open tag:

```
<?php

$txt = "Hello world!";
$x = 5;
$y = 10.5;

echo $txt;
```

I think is related to this issue. I tried the workaround from there and it doesn't work.

#9 - 2016-12-29 06:56 - Go MAEDA

- File highlight-php.png added

- File 0002-Update-help-for-Rouge-syntax-highlighter.patch added

- File 0003-Update-jstoolbar-for-Rouge-syntax-highlighter.patch added

- File 0004-s-CodeRay-Rouge.patch added

- File 0005-Support-PHP-snippets-without-open-tag.patch added

Marius BALTEANU wrote:

> I tested the patch and I've the following observations:
>
> 1. We should rename the "codeRay" variable in source:trunk/public/javascripts/jstoolbar/jstoolbar.js#L378
> 2. Rename the "CodeRay" from source:trunk/public/stylesheets/rtl.css#L375

3. Update help documentation: source:trunk/public/help/en/wiki_syntax_detailed_markdown.html#L300

Thanks for your feedback, I have fixed all of the above.

> Also, the Rogue library seems to have an issue with the PHP language which is highlighted only when the opening tag is present.

Also fixed.

With open tag:

```php
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
echo $txt;
```

Without open tag:

```php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
echo $txt;
```

#10 - 2016-12-29 16:50 - Go MAEDA

- File 0006-Fixed-multiline-comments-highlighting-issue-in-file-.patch added

- File multiline-comment-before.png added

- File multiline-comment-after.png added

Fixed multiline comments highlighting issue in file view, the same problem that was reported as #7495 for CodeRay.

Before fix:

After fix:

#11 - 2016-12-30 02:13 - Go MAEDA

- Description updated

Rouge 2.0.7 supports 113 languages.
The following is the list of supported languages.

Show the list of supported languagesHide

Languages written in **bold** are supported by CodeRay.

abap	SAP - Advanced Business Application Programming
actionscript	ActionScript [aliases: as,as3]
apache	configuration files for Apache web server
apiblueprint	Markdown based API description language. [aliases: apiblueprint,apib]
applescript	The AppleScript scripting language by Apple Inc. (http://developer.apple.com/applescript/) [aliases: applescript]
biml	BIML, Business Intelligence Markup Language
bsl	The 1C:Enterprise programming language
c	**The C programming language**
ceylon	Say more, more clearly.
cfscript	CFScript, the CFML scripting language [aliases: cfc]
clojure	**The Clojure programming language** (clojure.org) [aliases: clj,cljs]
cmake	The cross-platform, open-source build system
coffeescript	The Coffeescript programming language (coffeescript.org) [aliases: coffee,coffee-script]
common_lisp	The Common Lisp variant of Lisp (common-lisp.net) [aliases: cl,common-lisp,elisp,emacs-lisp]
conf	A generic lexer for configuration files [aliases: config,configuration]
coq	Coq (coq.inria.fr)
cpp	**The C++ programming language** [aliases: c++]
csharp	a multi-paradigm language targeting .NET [aliases: c#,cs]
css	**Cascading Style Sheets, used to style web pages**
d	The D programming language(dlang.org) [aliases: dlang]
dart	The Dart programming language (dartlang.com)
diff	**Lexes unified diffs or patches** [aliases: patch,udiff]
docker	Dockerfile syntax [aliases: dockerfile]
eiffel	Eiffel programming language
elixir	Elixir language (elixir-lang.org) [aliases: elixir,exs]
erb	**Embedded ruby template files** [aliases: eruby,rhtml]
erlang	The Erlang programming language (erlang.org) [aliases: erl]
factor	Factor, the practical stack language (factorcode.org)
fortran	Fortran 95 Programming Language
fsharp	F# (fsharp.net)
gherkin	A business-readable spec DSL (github.com/cucumber/cucumber/wiki/Gherkin) [aliases: cucumber,behat]
glsl	The GLSL shader language
go	**The Go programming language** (http://golang.org) [aliases: go,golang]
gradle	A powerful build system for the JVM
groovy	**The Groovy programming language** (http://www.groovy-lang.org/)
haml	**The Haml templating system for Ruby** (haml.info) [aliases: HAML]
handlebars	the Handlebars and Mustache templating languages [aliases: hbs,mustache]
haskell	The Haskell programming language (haskell.org) [aliases: hs]
html	**HTML, the markup language of the web**
http	http requests and responses
idlang	Interactive Data Language
ini	the INI configuration format

io	The IO programming language (http://iolanguage.com)
java	**The Java programming language** (java.com)
javascript	**JavaScript, the browser scripting language** [aliases: js]
jinja	Django/Jinja template engine (jinja.pocoo.org) [aliases: django]
json	**JavaScript Object Notation** (json.org)
json-doc	JavaScript Object Notation with extenstions for documentation
jsonnet	An elegant, formally-specified config language for JSON
jsx	jsx [aliases: jsx,react]
julia	The Julia programming language [aliases: jl]
kotlin	Kotlin <http://kotlinlang.org>
liquid	Liquid is a templating engine for Ruby (liquidmarkup.org)
literate_coffeescript	Literate coffeescript [aliases: litcoffee]
literate_haskell	Literate haskell [aliases: lithaskell,lhaskell,lhs]
llvm	The LLVM Compiler Infrastructure (http://llvm.org/)
lua	**Lua** (http://www.lua.org)
make	Makefile syntax [aliases: makefile,mf,gnumake,bsdmake]
markdown	Markdown, a light-weight markup language for authors [aliases: md,mkd]
matlab	Matlab [aliases: m]
moonscript	Moonscript (http://www.moonscript.org) [aliases: moon]
mxml	MXML
nasm	Netwide Assembler
nginx	configuration files for the nginx web server (nginx.org)
nim	The Nim programming language (http://nim-lang.org/) [aliases: nimrod]
objective_c	an extension of C commonly used to write Apple software [aliases: objc]
ocaml	Objective CAML (ocaml.org)
pascal	**a procedural programming language commonly used as a teaching language.**
perl	The Perl scripting language (perl.org) [aliases: pl]
php	**The PHP scripting language** (php.net) [aliases: php,php3,php4,php5]
plaintext	**A boring lexer that doesn't highlight anything** [aliases: text]
powershell	powershell [aliases: posh]
praat	The Praat scripting language (praat.org)
prolog	The Prolog programming language (http://en.wikipedia.org/wiki/Prolog) [aliases: prolog]
prometheus	prometheus [aliases: prometheus]
properties	.properties config files for Java
protobuf	Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data [aliases: proto]
puppet	The Puppet configuration management language (puppetlabs.org) [aliases: pp]
python	**The Python programming language (python.org)** [aliases: py]
qml	QML, a UI markup language [aliases: qml]
r	The R statistics language (r-project.org) [aliases: r,R,s,S]
racket	Racket is a Lisp descended from Scheme (racket-lang.org)
ruby	**The Ruby programming language** (ruby-lang.org) [aliases: rb]
rust	The Rust programming language (rust-lang.org) [aliases: rs]
sass	**The Sass stylesheet language language** (sass-lang.com)
scala	The Scala programming language (scala-lang.org) [aliases: scala]
scheme	The Scheme variant of Lisp

scss	SCSS stylesheets (sass-lang.com)
sed	sed, the ultimate stream editor
shell	Various shell languages, including sh and bash [aliases: bash,zsh,ksh,sh]
shell_session	A generic lexer for shell session and command line [aliases: terminal,console]
slim	The Slim template language
smalltalk	The Smalltalk programming language [aliases: st,squeak]
smarty	Smarty Template Engine [aliases: smarty]
sml	Standard ML [aliases: ml]
sql	**Structured Query Language, for relational databases**
swift	Multi paradigm, compiled programming language developed by Apple for iOS and OS X development. (developer.apple.com/swift)
tap	Test Anything Protocol [aliases: tap]
tcl	The Tool Command Language (tcl.tk)
tex	The TeX typesetting system [aliases: TeX,LaTeX,latex]
toml	the TOML configuration format (https://github.com/mojombo/toml)
tulip	the tulip programming language (twitter.com/tuliplang) [aliases: tulip]
turtle	Terse RDF Triple Language, TriG
twig	Twig template engine (twig.sensiolabs.org)
typescript	TypeScript, a superset of JavaScript [aliases: ts]
vala	A programming language similar to csharp.
vb	Visual Basic [aliases: visualbasic]
verilog	The System Verilog hardware description language
vhdl	Very High Speed Integrated Circuit Hardware Description Language
viml	VimL, the scripting language for the Vim editor (vim.org) [aliases: vim,vimscript,ex]
vue	Vue.js single-file components [aliases: vuejs]
xml	**<desc for="this-lexer">XML</desc>**
yaml	**Yaml Ain't Markup Language** (yaml.org) [aliases: yml]

#12 - 2016-12-30 16:50 - Go MAEDA

- Target version set to Candidate for next major release

#13 - 2016-12-31 10:26 - Mischa The Evil

- Assignee set to Mischa The Evil

I'm currently wrapping-up a review of the proposed change and the corresponding patches provided by Go.

#14 - 2017-01-01 08:07 - Mischa The Evil

- Status changed from New to Needs feedback

- Assignee deleted (Mischa The Evil)

Done! Here it is...

I have spent some time testing the Rouge syntax highlighter using the patches provided by Go in contrast to the implementation provided by the redmine_rouge plugin (which seems to have an issue with the CSS-styles — thus the highlighting — missing during printing).
I specifically looked at the Ruby code syntax highlighting, but I also included two others: Diff and HTML (with inline CSS and Javascript).

Procedure:

I started with collecting some — let's say — more challenging code examples. I found some interesting ones (specifically for comparison of CodeRay vs. Rouge) in the CodeRay Scanner Test repository (https://github.com/rubychan/coderay-scanner-tests). I also took at random some long, more complicated methods from the Redmine core to compare.

With these examples in place, I added all the code to a single wikipage and dropped the files into the projects repository for testing of the repo view file function (my findings for the single wikipage are not different from the projects repository view file functionality, so I won't go into this specifically). I then printed the whole wikipage to a PDF (using Chrome's internal print function — this is actually what's not working using the redmine_rouge plugin) for easy and consistent comparison.

I did these tests (by means of anything better) on a fresh deployment of a Bitnami Redmine 3.3.1 stack. First I printed the wikipage as said above using the default CodeRay highlighter. Then I applied the patches (cherry-picked patches 0001, 0005 and 0006; omitting 0002, 0003 and 0004 patches) manually, restarted the whole stack and printed the same wikipage using the Rouge highlighter. Good to note is that I thus have tested Rouge's capabilities using the currently patched-in colorful style theme.

I think that the results I got (two, 9+ MB, 25-page, A3-landscape pdf's) are actually pretty clear albeit large — sneakpeak: I'm not very excited about them. I don't talk about performance here (I haven't tested that), but rather about the quality of the provided code highlighting of, particularly, the Ruby and the Diff code by the Rouge library. Along with this, I also see some issues with the current CodeRay highlighter (whether these are issues in the CodeRay library or the Redmine implementation, I don't know at the moment). I'll below elaborate on my observations during the review, using numbered items and referring to the filenames of the testfiles I have used. I have written it such that one can read/scan along the test code in the pdf's (preferably splitscreen) in roughly the same order.

Observations:

No.:	Filename:	Comments:
Ruby code from CodeRay Scanner Test repository		
o01.	[def.in.rb]	Rouge breaks on the ampersands in front of the blocks.
o02.	[diffed.in.rb]	With Rouge no clear difference between regex and string highlighting, and operator keywords aren't highlighted.
o03.	[operators.in.rb]	Rouge doesn't recognize the aliases, gives the at signs the error class and doesn't highlight method definitions where it should.
o04.	[quotes.in.rb]	Rouge breaks on the complex quoted literals and doesn't differ between different parts of regexes. CodeRay seems to break on the ampersand within the regex.
o05.	[regexp.in.rb]	Rouge doesn't highlight code inside regexes, doesn't highlight escape sequences within regexes.
o06.	[ruby19.in.rb]	Rouge seems to have problems taking in the Ruby 1.9 hash syntaxes.
o07.	[ruby2.in.rb]	Rouge breaks on the ampersands, messes up the keyword argument symbol highlighting, highlights extend as pseudo-keyword, highlights self incorrect, doesn't differ (clearly) between the %i{} and %w{} literals (maybe even some other %* literals too), highlights Ruby 2.1 syntax wrong, doesn't highlight Ruby 2.2 hash literal symbol keys with colons/quotes correctly and has issues highlighting the Ruby 2.3 squiggly heredoc. CodeRay doesn't differ class names from module names in definitions, which Rouge does.
o08.	[strange.in.rb]	Rouge does not seem to differ floats from constants correctly, does not recognize/differ backticked shell code (using both ` and %x), does not differ between inline instance-/class-/global variables, doesn't differ modifiers within regex constructs, misses several distinctions (char vs content, delimiter vs constant), has troubles with nested code which causes the rendering of the highlighting of the rest of the code completely broken.
o09.	[undef.in.rb]	Rouge breaks on the / method. Both Rouge and CodeRay break on the ampersands.
o10.	[unicode.in.rb]	Rouge does not seem to differ between integers and floats, and it renders unicode chars with class error — which seems wrong.
Diff 'code' from CodeRay Scanner Test repository		
o11.	[diff.in.diff]	Rouge differs in what it highlights in comparison with CodeRay, where Rouge

		seems to put the attention more on the meta-data of the patch (diff command, indexes) and CodeRay more to the changes itself (filename, linenumbers). Rouge doesn't support inline change highlighting at all.
o12.	[github.in.diff]	See above. No inline change highlight.
o13.	[heredoc.in.diff]	CodeRay seems to do some inline highlighting of heredocs which Rouge doesn't (this seems to be a curious thing as far as I understand it fully).
HTML code from CodeRay Scanner Test repository		
o14.	[cdata.in.html]	Rouge does not mark the inside of the cdata blocks in the inline javascript and css comments and renders the < as a unicode char with class error.
o15.	[redmine.in.html]	Rouge has the same cdata block issue as above. Rouge doesn't highlight inline css and javascript, and does not highlight html entities. Redmine breaks both highlighters due to incorrect handling of code tags.
Ruby code from redmine source and issue		
o16.	[application_helper.rb]	Rouge incorrectly highlights attr as pseudo keyword. CodeRay has more distinct highlighting of escape sequences within regexes in contrast to Rouge. Both CodeRay and Rouge break on the ampersands (Rouge multiple times, CodeRay once). CodeRay provides more distinct string highlighting. CodeRay provides better distinction of regexes vs strings than Rouge. CodeRay does not highlight buildin methods (name, const_defined?, lambda, class_eval, etc.) specially as like Rouge does (as they are just methods which CodeRay doesn't highlight in any way).
o17.	[multiline_comment.rb]	Rouge omits highlighting of the she-bang line.
o18.	[project_nested_set.rb]	Here it becomes clear IMO that Rouge's method highlighting can be a real good thing (see eg. the calls to the *_changed? methods within the lambda and the method chains of self.class.where...where...maximum and self.class.where...pluck...first, etc).

Two of the biggest differences that seem to be recurring along a multitude of the Ruby testfiles are due to differences in design perspective:

x1: Rouge highlights 'pseudo-keywords', CodeRay doesn't — for good reasons AFAICR.
x2: Rouge highlights methods (class 'nf'), CodeRay doesn't — again, for good reasons AFAICR.

Some conclusions:

A1: The tested patch series succesfully implements the replacement of CodeRay with Rouge (excluding conclusions B1 and B2, in case these are due to Redmine implementation(s)).
A2: The reported PHP issue and the additionally found multiline issue are succesfully fixed by the additional patches.
A3: We might be able to tweak the js-toolbar code button in some neat way to support (a selection of) popular languages added by Rouge.

B1: CodeRay (and Rouge) highlighting (within the current Redmine implementation) can break on code tags within (at least) HTML code.
B2: CodeRay (and [significantly more] Rouge) highlighting (within the current Redmine implementation) can break on ampersands within (at least) Ruby code.

C1: Rouge has a lot (more) issues highlighting Ruby/HTML/JS/CSS/Diff code correctly/sufficiently.
C2: Rouge misses some more essential Diff highlighting features.
C3: Rouge misses support for inline CSS/JS highlighting within HTML code.

D1: Overall I think that CodeRay performs better judging by the quality and features of the provided highlighting of the tested languages.
D2: Rouge does indeed provide support for way more languages than CodeRay does (and I really like that), but if the same kind of issues are among those languages too, I think we'd just be making a bad trade-of between quantity over quality if we switch away from CodeRay (at least at the moment).

As the observant reader may have already noticed above, I am a bit disappointed by the overall quality of the highlighting of the Rouge library in

comparison to CodeRay. If you'd ask me whether or not to replace CodeRay with Rouge, I'd vote against such a switch now.
I do however think that a lot of people are willing to make this trade-of considering the amount of languages supported by Rouge. As such could it be a good idea to build upon the changes from #2985 and integrate additional support for the Rouge highlighter within the core while keeping CodeRay as the default (with a front-end UI in the form of a setting; just like done for the Redmine Text Formatting and as like was originally proposed by Jean-Baptiste in #2985). That shouldn't be all too difficult to get done and provides the users an easy way to make the choice for themselfs without having to rely on a third-party plugin.

Some additional comments:

Go MEADA wrote:

> Citation from Rouge's README.md :
>
> > Advantages to CodeRay
> >
> > - The HTML output from Rouge is fully compatible with stylesheets designed for pygments.
> > - The lexers are implemented with a dedicated DSL, rather than being hand-coded.
> > - Rouge supports every language CodeRay does and more.

Regarding the first: I don't see how this is an advantage of Rouge over CodeRay for the end user. Regarding the second: I think this may actually well be a drawback being (one of) the cause(s) of a some or more of the Ruby highlighting issues observed above in the testfiles. Regarding the third: well, I see taskpaper (which indeed isn't a language, true ;). Besides that one, I really think that with syntax highlighting it is a case of doing it good, then it's useful. Doing it wrong, then it's just pretty-coloring code and as such a waste of processor-cycles. In my opinion quality should go way over quantity on this matter.

Go MAEDA wrote:

> Marius BALTEANU wrote:
>
> > I tested the patch and I've the following observations:
> >
> > 1. We should rename the "codeRay" variable in source:trunk/public/javascripts/jstoolbar/jstoolbar.js#L378
> > 2. Rename the "CodeRay" from source:trunk/public/stylesheets/rtl.css#L375
> >   [...]
>
> Thanks for your feedback, I have fixed all of the above.

Ragarding the first: you seem to know how the minified js file is created as you seem able to patch it. I — and I'm sure some others too — are quite interested in the used process. Do you want to elaborate on that? Regarding the second: as far as I know the linenumbers styles became unused after r10131 and were removed from regular stylesheets with r14487. As such is patch 0004 obsolete and should it be replaced by a complete removal of both the comment and the actual style definition from rtl.css.

Some additional comments about the in-/output files/code:

Total files: 4 (22 unpacked):

- One plain-text input document containing the content (of a wikipage) in textile: *codehighlight.examples.txt*;
- Two pdf output documents containing the rendered content from codehighlight.examples.txt as a wikipage for both highlighters: *codehighlight.examples.coderay.pdf* and *codehighlight.examples.rouge.pdf*;
- One zip-archive containing eighteen individual test code files: *syntaxhl-testfiles.zip*.

Because of the size limitations on redmine.org I have uploaded these four files to my old MediaFire-account (http://www.mediafire.com/evildev). I uploaded them to the subfolder 'rm24681' of the 'Redmine Miscellaneous' folder. A direct link to this shared folder is:
https://www.mediafire.com/folder/9994htt2r7fdg/rm24681

Wrap-up:

If you have made it to here, yay. Seriously, apologies for the lenghty post! I thought let's start this new year by posting a nice detailed review... Best whishes for 2017 to all participants of this issue and beyond!

I'm interested in (further) feedback and willing to answer/participate in additional questions/discussions. Other experiences (eg. with other languages) from users using this patch or the plugin are very welcome if you ask me.

Kind regards, Mischa.

#15 - 2017-01-01 13:15 - Go MAEDA

A happy new year, Mischa. I am very impressed and deeply grateful for your deep inspection for Rouge and the patches.

As you wrote, I have to admit that there are some problems for now.

> I do however think that a lot of people are willing to make this trade-of considering the amount of languages supported by Rouge. As such could it be a good idea to build upon the changes from #2985 and integrate additional support for the Rouge highlighter within the core while keeping CodeRay as the default (with a front-end UI in the form of a setting; just like done for the Redmine Text Formatting and as like was originally proposed by Jean-Baptiste in #2985).

How about using both CodeRay and Rouge? It means that syntax highlighting will be basically processed by CodeRay and fall back to Rouge if the language is not supported by CodeRay. With this method, we can increase supported languages while avoiding deteriorating the quality of syntax highlighting. In addition, it is possible to prevent increase of admin settings.

> Ragarding the first: you seem to know how the minified js file is created as you seem able to patch it. I — and I'm sure some others too — are quite interested in the used process. Do you want to elaborate on that?

I just replaced the variable name with an editor.

> Regarding the second: as far as I know the linenumbers styles became unused after r10131 and were removed from regular stylesheets with r14487. As such is patch 0004 obsolete and should it be replaced by a complete removal of both the comment and the actual style definition from rtl.css.

Thanks, I will fix it.

#16 - 2017-01-02 12:18 - Go MAEDA

- File 0001-Syntax-highlighter-fall-back-to-Rouge-if-the-languag.patch added

Go MAEDA wrote:

> How about using both CodeRay and Rouge? It means that syntax highlighting will be basically processed by CodeRay and fall back to Rouge if the language is not supported by CodeRay. With this method, we can increase supported languages while avoiding deteriorating the quality of syntax highlighting. In addition, it is possible to prevent increase of admin settings.

I have created a new patch with a new approach: attachment:0001-Syntax-highlighter-fall-back-to-Rouge-if-the-languag.patch

- Syntax highlighting is mainly done by CodeRay.
- Rouge is used only when the given language is not supported by CodeRay.

We can increase supported language of syntax highlighting by this fallback mechanism while enjoying the high quality of CodeRay.

#17 - 2017-01-12 01:01 - Go MAEDA

- Assignee set to Mischa The Evil

Mischa, could you review the new patch on #24681#note-16?

attachment:0001-Syntax-highlighter-fall-back-to-Rouge-if-the-languag.patch

I think it can make use of the good points of both CodeRay and Rouge.

#18 - 2017-01-12 12:46 - Mischa The Evil

Go MAEDA wrote:

> Mischa, could you review the new patch on #24681#note-16?

I will, and I'll also try yet another approach building upon yours.
I have a quick question in advance though (which holds true for both patch approaches): unpatched highlight_by_filename seem to render unknown content (language == false) using ERB::Util.h and this behaviour seems to change in both your patches while highlight_by_language seems to gain this behaviour by setting lexer to ::Rouge::Lexers::PlainText when the content isn't recognized. Is this correct and wanted? Can you elaborate on that?

#19 - 2017-01-13 04:54 - Go MAEDA

Mischa The Evil wrote:

> I have a quick question in advance though (which holds true for both patch approaches): unpatched highlight_by_filename seem to render unknown content (language == false) using ERB::Util.h and this behaviour seems to change in both your patches while highlight_by_language seems to gain this behaviour by setting lexer to ::Rouge::Lexers::PlainText when the content isn't recognized. Is this correct and wanted? Can you elaborate on that?

I wrote the code to avoid "RuntimeError: unknown lexer" exception. But I looked the original syntax_highlight.rb again and I realized that the exception will be catched in Redmine::SyntaxHighlighting.highlight_by_language and then ERB::Util.h(text) will be performed. So, I don't have to set lexer to ::Rouge::Lexers::PlainText.

Thanks for pointing it out.

#20 - 2017-01-16 14:38 - Go MAEDA

Mischa The Evil wrote:

> I have a quick question in advance though (which holds true for both patch approaches): unpatched highlight_by_filename seem to render unknown content (language == false) using ERB::Util.h and this behaviour seems to change in both your patches while highlight_by_language seems to gain this behaviour by setting lexer to ::Rouge::Lexers::PlainText when the content isn't recognized. Is this correct and wanted? Can you elaborate on that?

I think it is no problem to use ::Rouge::Lexers::PlainText as a lexer because it does nothing and finally Rouge.highlight(text, ::Rouge::Lexers::PlainText, ::Rouge::Formatters::HTML) returns almost the same HTML with ERB::Util.h.

And patched version of highlight_by_filename also uses ::Rouge::Lexers::PlainText for unknown languages because ::Rouge::Lexer.guess_by_filename returns ::Rouge::Lexers::PlainText if Rouge cannot guess language.

#21 - 2017-01-25 16:06 - Go MAEDA

- Status changed from Needs feedback to New

- Target version changed from Candidate for next major release to 3.4.0

The patch attachment:0001-Syntax-highlighter-fall-back-to-Rouge-if-the-languag.patch is ready to merge. I am sure that supporting over 100 languages brings great benefits to users. Let's include this feature in 3.4.0.

But I cannot write good English, so it is hard for me to update public/help/*/wiki_syntax_detailed_*.html. Could someone add explanation about Rouge to help files?

#22 - 2017-01-28 09:56 - Mischa The Evil

Go MAEDA wrote:

> The patch attachment:0001-Syntax-highlighter-fall-back-to-Rouge-if-the-languag.patch is ready to merge. [...] Let's include this feature in 3.4.0.

As I've said in note-18, I have been working on extensively reviewing this. My preliminary conclusion is that the here mentioned patch should not yet be committed as its approach has some (major) drawback(s) and the implementation as-is comes with a big performance drawback. I'll wrap-up my review (including some clear data supporting my previous performance statement) and create/wrap-up an alternative patch this weekend.

#23 - 2017-01-29 11:58 - Jean-Philippe Lang

- Target version changed from 3.4.0 to Candidate for next major release

Thanks Mischa for reviewing this.

I didn't try the patches but it looks like CodeRay would still be used in some cases. Am I wrong? Because I'm not really in favor of keeping 2 code highlighters in the core.

#24 - 2017-03-03 08:50 - Hontvári Levente

Jean-Philippe, FYI: the conclusion of the discussion was that CodeRay's syntax highlighting seems to be significantly better quality for all 3 tested languages, but its list of supported languages is minuscule. That is why the combined approach was implemented in the final patch.

#25 - 2017-06-27 16:07 - Adrien Crivelli

This might be inappropriate, but have you considered re-using what GitHub uses ?

I'd say they should have a rather mature solution to support lots of language while keeping a good quality. It seems to live there: https://github.com/github/linguist

#26 - 2017-09-03 13:07 - Kornelius Kalnbach

For what it's worth, I think the idea of using CodeRay with a Rouge fallback is awesome. If you can make it work (the HTML output and formatting is pretty much incompatible), it would be the best of both worlds. I would also propose that for some languages, Rouge support is better (eg. Java and PHP).

As the maintainer of CodeRay, I can assure you that it will never support 100+ languages - at least not in its current form (version 2.0 is vaporware so far). Rouge is actively being developed, and support for JavaScript ES6, Swift, or other nice modern languages will probably never come to CodeRay.

Mischa The Evil: Wow, awesome analysis! It may be a bit unfair to use CodeRay's test suite to compare Rouge, but the quality vs. quantity aspect is the whole reason CodeRay exists. I think the ampersand issues are related to the Redmine integration, though.

Adrien Besson: Linguist is great, but it requires Python as far as I'm aware…which would make it a bad fit for Redmine, right?

@Jean-Philippe: If there would be a "coderouge" meta-gem, combining coderay and rouge, would you accept it as "1 code highlighter"? I can reach out to jneen, the maintainer of Rouge, and see if we can release such a thing. Might even be useful for other projects.

#27 - 2017-09-06 14:58 - Frank Church

I am reading this with interest as I use a number of languages Coderay doesn't serve.

Is there a separate ready-made plugin available for rouge or does it require the patch in question?

#28 - 2017-11-10 15:05 - Michael Kussmaul

Yes, redmine needs better syntax highlighting - my Swift and Objective-C code does not show any highlighting... So I like the approach of having the existing Coderay but also a fallback to Rouge. It is better than having nothing at all...

#29 - 2017-11-11 05:36 - Go MAEDA

GitLab CE uses Rouge as syntax highlighter (we can find gem 'rouge' in https://github.com/gitlabhq/gitlabhq/blob/master/Gemfile). It works well and most people are satisfied with it, I think.

Although Rouge may be imperfect as Mischa pointed out in #24681#note-14, I think supporting 100+ language is valuable for most Redmine users.

#30 - 2017-11-11 12:27 - Adrien Crivelli

I agree that supporting more languages is valuable. We have to keep in mind that this is only used to help reading code, not actual coding. So even if there are imperfection I believe most of them would be overlooked anyway.

I am favor of implementing only rouge, as it is most likely simpler to do and thus could be available sooner.

#31 - 2017-11-11 16:01 - Marius BĂLTEANU

+1 for moving to Rouge only.

#32 - 2018-01-11 02:52 - Alex Bevilacqua

For anyone that wants to take advantage of the great work Go has done here but doesn't want to patch their installation I've packaged this up as a plugin:

https://github.com/alexbevi/redmine_rouge_highlighter

#33 - 2018-01-11 05:25 - Adrien Crivelli

Alex, thank you for the plugin, I just installed it, and it seems to work great for browsing repository content. However it doesn't seems to do anything for user typed content (issues, wiki, etc.). Is this the expected behavior ?

I suppose Go Maeda could also answer that question, since he wrote the patch in the first place.

In case it might be useful, here's my env:

```
Environment:
  Redmine version                3.3.3.stable
  Ruby version                   2.2.6-p396 (2016-11-15) [x86_64-linux-gnu]
  Rails version                  4.2.7.1
  Environment                    production
  Database adapter               Mysql2
SCM:
  Subversion                     1.9.3
  Git                            2.7.4
  Filesystem
  Xitolite                       2.7.4
Redmine plugins:
  redmine_bootstrap_kit          0.2.5
  redmine_git_hosting            1.2.2
  redmine_github_hook            2.2.0
  redmine_image_clipboard_paste  3.3.0
  redmine_rouge_highlighter      0.0.1
  redmine_tags                   3.2.0
```

#34 - 2018-01-11 19:36 - Alex Bevilacqua

Adrien, I didn't actually test with anything other than the repository browser. I can have a look in a bit though as I'm not sure how they're different off

the top of my head

#35 - 2018-01-11 22:03 - Alex Bevilacqua

Adrien, I've addressed this issue in the plugin by changing the check for supported lexers from using CodeRay's list to Rouge's list. This still attempts to use CodeRay first then properly falls back to Rouge (see https://github.com/alexbevi/redmine_rouge_highlighter/commit/f9ae06978e48c15c2e13ae726cf311c1af20fe1c)

#36 - 2018-01-12 05:55 - Adrien Crivelli

This is great, it works well with Markdown instead of Textile too. I finally got sh syntax highlighting among many other things. Thank you !

#37 - 2018-01-29 04:09 - Go MAEDA

- Related to Feature #28094: Kotlin code highlight support added

#38 - 2018-02-14 14:55 - Roman Yagodin

+1

#39 - 2018-02-21 14:19 - beko akabeko

+1

#40 - 2018-08-24 11:10 - Lin Yanting

+1

#41 - 2018-08-24 13:41 - Adrien Crivelli

A lot of users (12 in this thread) expressed they would rather have many ok-quality languages, rather than a few high-quality languages. Even Jean-Philippe seems to agree to move to Rouge. So far only Mischa spoke against it. The last time he spoke was over 1 year ago saying he would come up with more data. But it seems he never got to it.

Can't we move on and replace Coderay with Rouge once and for all ?

Go MAEDA, would you be able to merge this for 4.0.0 ?

#42 - 2018-08-24 15:00 - Benoit Blais

+1

#43 - 2018-08-24 15:19 - Anonymous

+1

#44 - 2018-08-24 15:20 - luigifab !

+1 :)

#45 - 2018-08-24 16:46 - Bernhard Rohloff

+1

#46 - 2018-08-26 09:39 - Go MAEDA

- File 0001-Replace-syntax-highlighter-CodeRay-with-Rouge.patch added

I have updated the patch. Now it is compatible with the latest trunk (3.4.6.devel.17471).

I Still think that switching to Rouge is beneficial for most users. I understand Mischa's opinion that CodeRay deals with complex Ruby code very well. But I think they are edge cases. Rouge works fine against most codes. Rouge is already used by major projects such as Jekyll and Gitlab.

The biggest problem for me is that CodeRay does not support popular languages such as C#, Swift, and Kotlin. The following table shows that top 25 popular languages on GitHub (according to Ranking Programming Languages by GitHub Users) and support status by CodeRay and Rouge. Rouge supports 24 of 25 languages but CodeRay supports only 10 languages. To make matters worse, CodeRay does not support emerging languages such as TypeScript, Swift, and Kotlin.

Rank	Language	CodeRay	Rouge
1	JavaScript	x	x
2	Python	x	x
3	Java	x	x

4	C++	x	x
5	C	x	x
6	PHP	x	x
7	C#		x
8	Shell		x
9	Go	x	x
10	TypeScript		x
11	Ruby	x	x
12	Jupyter Notebook		
13	Objective-C		x
14	Swift		x
15	Kotlin		x
16	R		x
17	Scala		x
18	Rust		x
19	Lua	x	x
20	Matlab		x
21	PowerShell		x
22	CoffeeScript		x
23	Perl		x
24	Groovy	x	x
25	Haskell		x

#47 - 2018-08-26 09:41 - Go MAEDA

Jean-Philippe Lang wrote:

> I didn't try the patches but it looks like CodeRay would still be used in some cases. Am I wrong? Because I'm not really in favor of keeping 2 code highlighters in the core.

I have removed CodeRay in the latest patch.

#48 - 2018-09-16 00:41 - Go MAEDA

- Assignee changed from Mischa The Evil to Jean-Philippe Lang

- Target version changed from Candidate for next major release to 4.1.0

#49 - 2018-09-20 01:31 - Go MAEDA

- Related to Defect #29259: Attachment preview does not work for some source files such as JavaScript and Go added

#50 - 2018-09-25 04:32 - Go MAEDA

- Target version changed from 4.1.0 to 4.0.0

#51 - 2018-09-25 04:34 - Go MAEDA

- File deleted (0001-Syntax-highlighter-fall-back-to-Rouge-if-the-languag.patch)

#52 - 2018-09-25 04:34 - Go MAEDA

- File deleted (0001-Replace-syntax-highlighter-CodeRay-with-Rouge.patch)

#53 - 2018-09-25 04:34 - Go MAEDA

- File deleted (0003-Update-jstoolbar-for-Rouge-syntax-highlighter.patch)

#54 - 2018-09-25 04:35 - Go MAEDA

- File deleted (0004-s-CodeRay-Rouge.patch)

#55 - 2018-09-25 04:35 - Go MAEDA

- File deleted (0005-Support-PHP-snippets-without-open-tag.patch)

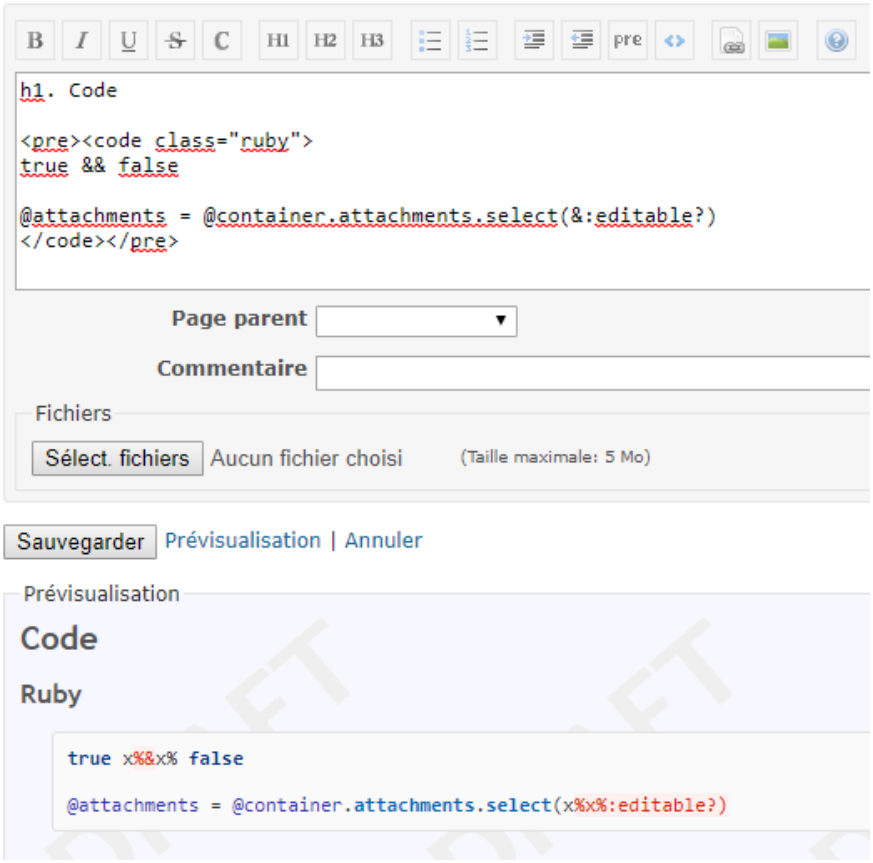
#56 - 2018-09-25 04:36 - Go MAEDA

- File deleted (0006-Fixed-multiline-comments-highlighting-issue-in-file-.patch)

#57 - 2018-09-26 09:12 - Jean-Philippe Lang

- File ruby_broken.png added

I've applied the latest patch and gave it a try with the content of attachments_controller.rb. There's a serious problem with & caracters:



#58 - 2018-09-26 09:14 - Go MAEDA

Jean-Philippe Lang wrote:

> I've applied the latest patch and gave it a try with the content of attachments_controller.rb. There's a serious problem with & caracters:

Thank you for reviewing, I will check it in a hurry.

#59 - 2018-09-27 05:43 - Go MAEDA

Jean-Philippe Lang wrote:

> There's a serious problem with & caracters:

This problem occurs when the text formatting setting is Textile. redcloth3.rb replaces all "&" to "x%x%" before calling highlight_by_language method (see #29681).

Maybe redcloth3.rb must replace "x%x%" with "&" before calling highlighter.

#60 - 2018-09-27 12:25 - Marius BĂLTEANU

- Related to Defect #29681: "x%x%" is rendered as "&" in Textile formatter added

#61 - 2018-09-27 12:33 - Go MAEDA

- File 0001-Switch-syntax-highlighter-to-Rouge-from-CodeRay.patch added

Fixed the ampersand issue. Please test 0001-Switch-syntax-highlighter-to-Rouge-from-CodeRay.patch.

The reason why ampersand is broken is that recloth3.rb replaces "&" with "x%x%" while processing Textile. if text "x = a & b" is given, it will be converted to "x = a x%x% b" temporarily and passed to syntax highlighter.

To convert "x%x%" back to "&" before processing with Rouge, I added gsub! method just before calling highlight_by_language in lib/redmine/wiki_formatting/textile/formatter.rb.

#62 - 2018-09-29 08:57 - Jean-Philippe Lang

- Status changed from New to Closed

- Resolution set to Fixed

Thanks for fixing this issue, patch is committed.

#63 - 2018-09-29 18:18 - Adrien Crivelli

Thank you Go MAEDA for not giving up this feature and everyone else who contributed to the patch being merged.

#64 - 2018-09-29 19:59 - Marius BĂLTEANU

- Status changed from Closed to Reopened

r17532 comments out the gem 'puma', please see source:trunk/Gemfile#L92 and I don't think that it is a desired change.

#65 - 2018-09-29 23:53 - Marius BĂLTEANU

- Related to Defect #26708: Diff formatting results empty lines if they contains HTML tags added

#66 - 2018-09-30 05:57 - Go MAEDA

- Related to Defect #20758: Ampersand+keyword in code highlighting added

#67 - 2018-09-30 18:32 - Marius BĂLTEANU

- Status changed from Reopened to Closed

I'm closing this because the unwanted change was reverted by r17537.

#68 - 2019-02-02 06:13 - Go MAEDA

- Related to Defect #30434: Line height is too large when previewing files with syntax highlighting if the line terminators are CRLF added

#69 - 2020-04-16 13:54 - Greg T

This broke Java syntax again. :(
https://github.com/rouge-ruby/rouge/pull/1414

#70 - 2021-08-04 17:38 - Mischa The Evil

- Related to Feature #35676: Optimize performance of syntax highlighting implementation added

Files

redmine_rouge_plugin_csharp.png	17.2 KB	2016-12-26	Go MAEDA
highlight-sample.png	88.6 KB	2016-12-28	Go MAEDA
without_opening_tag.png	2.1 KB	2016-12-28	Marius BĂLTEANU
with_open_tag.png	3.14 KB	2016-12-28	Marius BĂLTEANU
highlight-php.png	15.2 KB	2016-12-29	Go MAEDA
0002-Update-help-for-Rouge-syntax-highlighter.patch	457 KB	2016-12-29	Go MAEDA
multiline-comment-before.png	15.7 KB	2016-12-29	Go MAEDA
multiline-comment-after.png	15.7 KB	2016-12-29	Go MAEDA
0001-Replace-syntax-highlighter-CodeRay-with-Rouge.patch	40.7 KB	2018-08-26	Go MAEDA
ruby_broken.png	25.6 KB	2018-09-26	Jean-Philippe Lang
0001-Switch-syntax-highlighter-to-Rouge-from-CodeRay.patch	21.6 KB	2018-09-27	Go MAEDA