redmineorg-copy202205 - Vote #65238

Update CodeRay 0.9

2022/05/09 14:03 - Admin Redmine

ステータス:	Closed	**開始日:**	2009/05/13
優先度:	通常	**期日:**	
担当者:		**進捗率:**	0%
カテゴリ:	Text formatting_26	**予定工数:**	0.00時間
対象バージョン:	1.0.0 (RC)_14	**作業時間:**	0.00時間
Redmineorg_URL:	https://www.redmine.org/issues/3359	**status_id:**	5
category_id:	26	**tracker_id:**	2
version_id:	14	**plus1:**	2
issue_org_id:	3359	**affected_version:**	
author_id:	2882	**closed_on:**	
assigned_to_id:	0	**affected_version_id:**	
comments:	28		

説明

Could you please update the CodeRay plugin to the svn HEAD version? We are currently developing some projects in Grails and the svn version of CodeRay supports some Groovy syntax...

journals

+1

+1

I have done this on my local installation. The only thing you have to do is to modify the scm css file (attachment:scm.css.patch) and to apply the changes from r710 (attachment:coderay-r710.patch) .

Actually, I just updated the plugin at the time. It works perfectly, but it would be great if it was already under Redmine repository. Actually it would be even greater if Redmine moved itself to Git instead of subversion. Installing the plugins as submodules would be great indeed.

Rodrigo Rosenfeld Rosas wrote:

> Actually, I just updated the plugin at the time. It works perfectly, but it would be great if it was already under Redmine repository. Actually it would be even greater if Redmine moved itself to Git instead of subversion. Installing the plugins as submodules would be great indeed.

We've discussed this in the forums a bit and decided "not right now". Until then, you can use my git mirror at http://github.com/edavis10/redmine

As far as upgrading CodeRay, I'd like to say on the released versions. Maybe we can upgrade to "0.8.3":http://rubyforge.org/frs/?group_id=704 ?

Eric Davis wrote:

> As far as upgrading CodeRay, I'd like to say on the released versions. Maybe we can upgrade to "0.8.3": http://rubyforge.org/frs/?group_id=704 ?

There is a 0.8.4 release launched on 2009-10-01 23:00 which is a lot better from the version that we download with redmine

dated 2006-10-20.
Is there any posibility you can include this version with the official redmine release? In case you can, when do you think that would be possible?
To make it work myself, if I replace the old folder with the new one, should work? or should I do something else?

Thanks,

Is there any progress on this feature?

Jean-Philippe has tried to update the currently used CodeRay-library from 0.7.6.227 to 0.9 RC2 in r3014. This led to some errors on redmine.org which made him decide to revert the update in r3079.
Though, to me it seems that the issue was caused by a forgotten change in the CodeRay-library which was initially made in r710.

I've reverted my Redmine dev-environment to trunk@r3078 (which includes CodeRay 0.9) and re-applied the patches to source:trunk/vendor/plugins/coderay-0.7.6.227/lib/coderay/encoders/html.rb from r710. This seems to work without any errors (like the issue #4253).

I've wrapped-up the patches against r3078 of the trunk and have attached them. Please test it to be able to integrate CodeRay 0.9 in Redmine version#6.

I also want to CodeRay has been upgraded to 0.9, because it supports PHP.

But I do not think that the modification of third-party libraries is a good idea. Maybe it's better to create a special HTML encoder, which enables to avoid modifying in the CodeRay library.

In the attached file a possible solution. Maybe not very good (I know Ruby is very bad), but on "my site": http://lastdragon.ru/projects/ipb3/wiki/Redmine%D0%9DighlightTest_CodeRay_09 it works. I will be glad if it will be useful.

I prefer the Aleksej's approach if the results are the same.

I would add some modifications to make the patch more Ruby-like.

The separator '\' is not necessary in the line below, although I think that line doesn't even need to be split:

```
DEFAULT_OPTIONS = CodeRay::Encoders::HTML::DEFAULT_OPTIONS.merge :line_numbers => :inline
```

Method are usually defined using parenthesis in Ruby coding style. Here are some other suggestions:

```
def token(text, type = :plain)
    return super unless text.is_a? String
    @out << (text != "\n" && style = @css_style[@opened]) ?
        (style << text << "") : text
end
```

I hope we are progressing on this :)

Aleksej Lebedev wrote:

> But I do not think that the modification of third-party libraries is a good idea. Maybe it's better to create a special HTML encoder, which enables to avoid modifying in the CodeRay library.

I agree! I only rebased Jean-Philippe Lang's changes from r710 to see if these fixed the errors, which was J-PL's motivation for reverting the library update (if that's true?)

Aleksej Lebedev wrote:

> In the attached file a possible solution. Maybe not very good (I know Ruby is very bad), but on "my site": http://lastdragon.ru/projects/ipb3/wiki/Redmine%D0%9DighlightTest_CodeRay_09 it works. I will be glad if it will be useful.

Works at my places too... Seems to be a pretty way to prevent library-hacking...

Rodrigo Rosenfeld Rosas wrote:

> The separator '\' is not necessary in the line below, although I think that line doesn't even need to be split:

Indeed, your example of the line works flawlessly.

Rodrigo Rosenfeld Rosas wrote:

> Method are usually defined using parenthesis in Ruby coding style. Here are some other suggestions:

```
def token(text, type = :plain)
    return super unless text.is_a? String
    @out << (text != "\n" && style = @css_style[@opened]) ?
        (style << text << ") : text
end
```

With this implementation of the @token@-method I get errors like "@ActionView::TemplateError (can't convert false into String) on line #.. of ...@" all the time with any call to @textilizable@.
To me it seems there houses a little bug in your definition of the @token@-method... :)

@ Jean-Philippe Lang: What do you think of this proposed solution?

Mischa, I didn't test the above code, and I don't really know how to test it. I was just reading and rewriting it.

At the time, I had a problem in my Linux box and my system stopped responding except for the mouse before I could submit the comment. I had to reboot and rewrite the code and I think I did not pay much attention to it then.

I guess I missed one line of code (*@opened[0] = type*):

```
...
return super unless text.is_a? String
@opened[0] = type
@out << (text != "\n" && style = @css_style[@opened]) ?
...
```

I hope this line fixes the problem...

Thanks for giving it a try :) By the way, how can I test this code?

May I join the discussion? I'm a bit surprised nobody talked to me yet ^ I'm the CodeRay maintainer.

I would be happy to provide a Redmine-compatible version that doesn't need any patches. But the main problem is not on the CodeRay side, in my view:

- You patch the HTML encoder not to escape the code.
- You have to do this because you're sending already escaped code to the scanner.
- Which is just evil and wrong. CodeRay need the source code in its plain form.

r1476 is an example of the horrible results: The C scanner doesn't recognize @#include @ any more!

Please unescape the code before sending it to CodeRay, and trust me that the HTML encoder still produces valid and secure HTML :) I've worked hard to ensure this.

Hi Kornelius, good to know you are interested on this issue. I really didn't think in talk to you due the same reason you stated in your message: I also don't believe this is a CodeRay issue.

But this is a bit complicated to apply your proposed change to Redmine from what I've read from Redmine's source code. The problem is that Redmine also patches RedCloth for achieving code highlighting. This is currently done by overriding a method that already receives an escaped string of the pre content.

Usually, I try to avoid monkey patches as much as I can. I don't think they are necessary for code highlighting in Redmine, but it would be a great effort to change Redmine textilazable approach completely for skipping the monkey patches.

Welcome aboard! :)

CodeRay 0.9 supports integration with RedCloth out of the box, but only for RedCloth 4 and up. You may be interested in the "code":http://redmine.rubychan.de/projects/coderay/repository/entry/trunk/lib/coderay/for_redcloth.rb (I'm using RedMine, too) I also need to unescape the code before highlighting (line 49f), but it works (I use it in my blog quite a lot).

So, there may be a way around this.

Well, I don't see why this change shouldn't be applied, but I'm not the one who takes this decision either.

But anyway, I think it would be even better if we didn't make use of monkey patching RedCloth.

My bet is that the correct approach should be:

1. scan the text to be textilizable, replacing the < code > tags with the html result from the code highlighter library (CodeRay, UltraViolet, whatever).
2. pass the semi-parsed content to RedCloth, BlueCloth, RDiscount, whatever.

There is no need for monkey patches in my opinion. I take this approach on my (yet in construction) own site and it is really simple.

What do you think?

I think what you call "monkey patch" is exactly the way we should extend RedCloth. There's nothing evil in it, if you're doing it right. Ruby has open classes for a reason.

But it's evil to send inadequate data to the highlighter ;) Sending pre-processed code to RedCloth sounds troublesome to me for the same reason.

So, what do we have?

- RedCloth is using a 3 year old version of CodeRay that ** is missing support for C++, CSS, diff, Groovy, Python, SQL, and YAML ** is missing the latest (improved/fixed) PHP, Java and JavaScript scanners
- Many people would be happy to see the new languages supported.
- The latest version (0.9.1) is incompatible because of an invalid "patch": http://www.redmine.org/attachments/2803/r710-patch_r3078.patch.
- I won't add the @:escape@ option to the HTML encoder in CodeRay, as explained before.

* Solving the problems with the update might be as simple as using @unescape@ before highlighting.

Ruby has open class for good reasons, but that is not the intended use of it in my opinion. It is great to add methods to classes, adding features to some classes, but this can be achieved more elegantly with modules too. And it is also fantastic that you can monkey patch some libraries instead of waiting for some fix to happen or being forced to modify the original libraries and tell others to do the same. But I don't think this should be viewed as a normal practice.

Particularly, in this case, I don't think that monkey patches worth because there is no need for it. Monkey patches make it difficult for someone to understand what is happening when such method is called when reading the sources... And worse than that, when the patched library change its implementation, the monkey patch could no more be valid, even if the user interface is not changed.

Sending pre-processed code to RedCloth, BlueCloth or UltraViolet is expected as a normal behaviour because Textile and Markdown allow HTML tags alongside the unformatted text. So, I don't see why this practice would be evil...

I agree with you that you shouldn't add the :escape option, since it makes no sense for RedCloth...

> Solving the problems with the update might be as simple as using unescape before highlighting.

I agree, but I don't think this approach is the best approach. I mean, it wouldn't be worse than it already is, so, again, I don't see why this patch shouldn't be applied. But I still think that refactoring this part of Redmine for not using monkey patches would be much better,

avoiding extra work (escape/unescape/escape), improving code readability and, most importantly, upgrading would be a no-pain procedure.

Mmh...but we can't change RedCloth, can we? The latest version is implemented in C.

Maybe we can preprocess the wiki text, highlighting code and insert links etc. before sending it to RedCloth. But I think it's much easier to use RedCloth's hooks (I think of the methods as hooks, so it's not monkey patching :-P)

That is exactly what I proposed on a prior comment:

> My bet is that the correct approach should be:
>
> 1. scan the text to be textilizable, replacing the tags with the html result from the code highlighter library (CodeRay, UltraViolet, whatever).
> 2. pass the semi-parsed content to RedCloth, BlueCloth, RDiscount, whatever. There is no need for monkey patches in my opinion. I take this approach on my (yet in construction) own site and it is really simple.

The question here should not be how easy it is to fix now, but how easy would be to upgrade the libraries at any time...

Okay. Do we have a patch?

Tomorrow, CodeRay 0.9.2 will be released. I recommend including this version, if we solve the :escape issue.

The discussion in #2985 and at the patches at http://www.redmine.org/boards/3/topics/6890 might be relevant.

Yes, it would be great to have plugabble highlighters. But this would make it even more important to abandon the @:escape@ hack, because you would have to patch every highlighter with such an option to be compatible.

Kornelius Kalnbach wrote:

> Okay. Do we have a patch?
>
> Tomorrow, CodeRay 0.9.2 will be released. I recommend including this version, if we solve the :escape issue.

The escape issue is solved in r3582. I'll upgrade to 0.9.2 as soon as it's released.

Wow, thanks! It's out, you can get it via Rubygems or "SVN": http://svn.rubychan.de/coderay/tags/0.9.2/. I hope it goes smooth...tell me if there are any problems.

Great news! Thanks!

I'm looking forward to this being in the trunk.

Trunk was upgraded to CodeRay 0.9.2 in r3592.
Everything seems fine so far :-)

Thanks.

related_issues

relates,Closed,4253,Formatting errors in block
relates,Closed,2575,PHP Syntax Highlighting Not Working

履歴

#1 - 2022/05/10 17:25 - Admin Redmine

- カテゴリ を Text formatting_26 にセット

- 対象バージョン を 1.0.0 (RC)_14 にセット