Redmine - Patch #32424
CommonMark Markdown Text Formatting

2019-11-06 09:27 - Jens Krämer

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Marius BALTEANU	**% Done:**	0%
Category:	Text formatting	**Estimated time:**	0.00 hour
Target version:	5.0.0		

Description

This patch introduces a new text formatting named *CommonMark Markdown (GitHub Flavored)*. It is based on CommonMarker and HTMLPipeline. The formatter was extracted from Planio where it will soon become the default for new accounts.

We built this instead of going with the existing RedCarpet Markdown implementation for a number of reasons:

- From time to time users who are using the current Markdown formatter ask for a spec / formal list of all supported features. No such thing exists for RedCarpet. There is CommonMark but RedCarpet isn't going to support it in the short to medium term (see next point).
- The future development of RedCarpet is uncertain. Few excerpts from a GitHub issue , a year ago:

> *Commonmark won't be supported anytime soon*

and

> *A general message about the project for people skimming this thread: I'm sorry Redcarpet isn't really active anymore but my resources are pretty limited since I'm committed to other open source projects and still a student. Feel free to use any existing alternative ; this project isn't the great tool it used to be when it was maintained by Vicent.*

- With CommonMark evolving as a Markdown spec that is supported by many implementations and endorsed by organizations like Gitlab and GitHub (which both did the switch from RedCarpet to CommonMarker a while ago), it quickly becomes what users expect when they hear 'Markdown'.
- Migrating existing Textile content is a bit easier since Pandoc has a dedicated Github Flavored Markdown writer module.
- The HTML pipeline approach encourages splitting up the formatting process into it's different aspects (html generation, sanitizing, syntax highlighting etc) which allows for better testability and has potential for future re-use of components with other text formatters. Further, HTML pipeline filters work on actual DOM nodes, making tasks like adding classes to links etc much more straight forward and less prone to bugs than doing so with regular expressions.

Last but not least, this formatter solves a number of currently open issues regarding the RedCarpet based *Markdown* Formatter:

- #19880 (Incorrect syntax for links in Markdown)
- #20497 (Markdown formatting supporting HTML)
- #20841 (Bare URLs in Markdown don't have "external" class)
- #29172 (Markdown: External links broken)

The main reason why we want to introduce this as a third formatting option and not just replace the existing Markdown formatter is line endings - this formatter does not insert hard breaks (
) for simple newlines, but requires either a \ or two spaces at the end of the line to do so. Over time the new formatter could become the default and the RedCarpet based one might be renamed to *Markdown (legacy)* or similar.

English help files are included, and the patch makes sure these are delivered for any other languages as well until corresponding

localized versions are available.

Related issues:

Related to Redmine - Defect # 29172: Markdown: External links broken	**Closed**
Related to Redmine - Defect # 20841: Bare URLs in Markdown don't have "extern...	**Closed**
Related to Redmine - Feature # 20497: Markdown formatting supporting HTML	**Closed**
Related to Redmine - Defect # 19880: Incorrect syntax for links in Markdown	**Closed**
Related to Redmine - Defect # 32563: Redmine 4 crashing with SEGFAULT under s...	**Closed**
Related to Redmine - Feature # 33037: Support for Gitlab flavored Markdown	**Closed**
Related to Redmine - Feature # 32766: Remove the URI limitation from external...	**New**
Related to Redmine - Feature # 35742: Enable task list items for Common Mark ...	**New**
Related to Redmine - Feature # 35747: Allow style attribute for HTML elements...	**New**
Related to Redmine - Defect # 35752: Warning message "nokogumbo: Using Nokogi...	**Closed**
Related to Redmine - Defect # 35754: Redmine::WikiFormatting::CommonMark::For...	**Closed**
Related to Redmine - Feature # 35035: Refactor text formatting to HTML::Pipeline	**New**
Related to Redmine - Defect # 35765: Code with unsupported code language is n...	**Closed**
Related to Redmine - Patch # 35104: Code blocks - consistent rendering and re...	**Closed**
Related to Redmine - Feature # 35889: Textile and Markdown attachment renderi...	**New**
Related to Redmine - Defect # 35892: Redmine::WikiFormatting::CommonMark::For...	**New**
Duplicated by Redmine - Defect # 22323: Markdown newline rendering broken	**Closed**

Associated revisions

Revision 21156 - 2021-08-11 23:40 - Marius BALTEANU

Adds CommonMark Markdown (GitHub Flavored) as third text formatting option (#32424).

Patch by Jens Krämer.

Revision 21157 - 2021-08-11 23:45 - Marius BALTEANU

Pin commonmarker to 0.22 if Ruby version is greater than 2.5 (#32424).

Revision 21158 - 2021-08-11 23:46 - Marius BALTEANU

Pin html-pipeline to 2.13.2 (#32424).

Revision 21159 - 2021-08-11 23:47 - Marius BALTEANU

Pin sanitize to 5.2 (#32424).

Revision 21160 - 2021-08-11 23:48 - Marius BALTEANU

Mark CommonMark Markdown (GitHub Flavored) as experimental (#32424).

Revision 21161 - 2021-08-11 23:49 - Marius BALTEANU

Relax allowed protocols in links by denying specific protocols for CommonMark text formatting (#32424).

Patch by Martin Cizek.

Revision 21162 - 2021-08-11 23:50 - Marius BALTEANU

Replace deprecated Sanitize keywords (#32424).

Patch by Martin Cizek.

Revision 21163 - 2021-08-11 23:50 - Marius BALTEANU

Fixed Layout/HeredocIndentation: Use 2 spaces for indentation in a heredoc by using <<~ instead of <<- (#32424).

Revision 21164 - 2021-08-11 23:51 - Marius BALTEANU

Fixed failing test on CommonMark by striping the trailing whitespace returned by the footnote (#32424).

Revision 21165 - 2021-08-11 23:52 - Marius BALTEANU

Render markdown attachments using markdown or common_mark based on the text formatting setting (#32424).

Patch by Marius BĂLTEANU and Martin Cizek.

Revision 21166 - 2021-08-11 23:52 - Marius BALTEANU

Fixed Replace class var @@allowlist with a class instance var (#32424).

Revision 21171 - 2021-08-13 08:37 - Marius BALTEANU

Pin sanitize to 6.0 (#35752, #32424).

Patch by Go MAEDA.

Revision 21177 - 2021-08-15 07:57 - Marius BALTEANU

Update failing test after hardbreaks enabled by default (#35754, #32424).

Patch by Go MAEDA.

Revision 21178 - 2021-08-15 08:51 - Go MAEDA

Don't try to install CommonMarker 0.22 on Ruby 2.6.0-rc (#32424).

Revision 21182 - 2021-08-15 22:59 - Marius BALTEANU

Add "data-language" attribute to code block with the user-supplied language for CommonMark formater (#35104, #32424).

Patch by Martin Cizek.

History

#1 - 2019-11-06 10:02 - Marius BALTEANU

I really, really like the proposed patch, especially for the HTML Pipeline gem that can help us a lot to improve the current code.

#2 - 2019-11-06 11:12 - Jan from Planio www.plan.io

- Related to Defect #29172: Markdown: External links broken added

#3 - 2019-11-06 11:12 - Jan from Planio www.plan.io

- Related to Defect #20841: Bare URLs in Markdown don't have "external" class added

#4 - 2019-11-06 11:12 - Jan from Planio www.plan.io

- Related to Feature #20497: Markdown formatting supporting HTML added

#5 - 2019-11-06 11:13 - Jan from Planio www.plan.io

- Related to Defect #19880: Incorrect syntax for links in Markdown added

#6 - 2019-11-26 14:37 - Go MAEDA

- Target version set to Candidate for next major release

#7 - 2019-12-07 05:30 - Go MAEDA

- Target version changed from Candidate for next major release to 4.2.0

I agree that Redcarpet is not active anymore. The number of commits made in this year is only 3. And I think many Markdown users expect Redmine to behave as CommonMark/GFM compliant because many apps/services that support it.

Let's start discussion to deliver this in 4.2.0.

#8 - 2019-12-08 14:23 - Go MAEDA

- Related to Defect #32563: Redmine 4 crashing with SEGFAULT under stress test when Markdown is used added

#9 - 2019-12-08 23:43 - Martin Cizek

- File 0002-attachments_helper-commonmark.patch added

Love you guys! We are playing with markup format conversions and preparing bulletproof arguments to integrate commonmarker + HTML sanitization for almost half a year. :) Jens did this job perfectly and the only unused argument regarding Redcarpet was #32563. I even tried to make a bulletproof pull request to solve an unpleasant Redcarpet issue observed in our Textile->Markdown migration (since 2013) to test if the project is really dead. It is.

Just a very small improvement is attached.

I'd sign every word written by Jens and would like to share a few remarks in subsequent comments, hope they help.

Go MAEDA, would you mind adding #22323 to related issues regarding the line breaks?

#10 - 2019-12-08 23:46 - Martin Cizek

Temporary workaround before the patch is merged

At the moment, it is possible to use redmine_common_mark plugin. It also allows for configuring commonmarker, but it has no HTML sanitizing implemented. Actually I found this patch when I was about to offer a pull request with html-pipeline to the plugin's author.

We'd still appreciate merging this patch ASAP.

#11 - 2019-12-08 23:47 - Martin Cizek

commonmarker configuration

These are the options used by GitLab. The options in the patch are the same (thumbs up!).
GitHub's configs have a few differences (it does not mean that we want them):

- STRIKETHROUGH_DOUBLE_TILDE is not used in repos.
- It uses HARDBREAKS in their issues, which is inconsistent with their wiki and repository rendering.
- They have the tasklist extension enabled. I'd consider enabling it in Redmine, as tasklits is an officially documented GFM feature.

I can imagine that some Redmine users would like other options than us, as we have seen before in Redmine. And it would be quite legit e.g. for the tasklists mentioned above.

Making commonmarker configurable can mean way too many config options, which is difficult to support. A good compromise might be to make a hook for plugins, so that they can change the config.

I can create a follow-up ticket after getting some feedback.

#12 - 2019-12-08 23:48 - Martin Cizek

HTML sanitizing

HTML sanitizing is currently embedded in the commonmark formatter in the patch. That's a good start.

As #807 suggests, sanitizing should rather be a shared concept, eventually with small differences for different formats. GitLab did it this way, their base_sanitization_filter.rb contains common sanitization setup and CommonMark sanitization_filter.rb customizes the HTML::Pipeline::SanitizationFilter on top of that.

I can create a follow-up ticket after getting some feedback.

#13 - 2019-12-08 23:48 - Martin Cizek

Textile to Markdown migration

Pandoc is actually bad at this job. We tried Jens'es redmine_convert_textile_to_markdown (thanks for that!), we ran rendering comparison tests[1] covering hundreds of thousands of strings and the results were poor.

[1] Rendering comparison test = grab all rendered issues and wiki pages from the Textile Redmine instance and a converted-to-Markdown Redmine instance, normalize HTML, compare the HTMLs.

So we forked it, and similarly to Jens, we were adding more and more preprocessors to make Pandoc happy and postprocessors to render it correctly. This is the ~~latest version of the fork~~. Later, we reworked it completely to a new project, which we'll publish soon.

But if we were doing it again, we would get rid of Pandoc completely. The preprocessing is done by partial rendering using code adapted from Redmine / Redcloth3. The amount of the code is comparable to normal rendering and invoking Pandoc just makes it slow.

Pandoc is a great tool, but this use case is just too specific for a universal format converter.

So the message is just: *a good converter exists for Redmine*. :)

#14 - 2019-12-10 03:18 - Jens Krämer

Hi, thanks for the feedback!

Nice catch about the rendering of Markdown files, that indeed should be switched to commonmark as well. Also great work on the Migrator :)

General use of HTML Pipeline

It would really make sense to use HTML pipeline for Textile rendering as well, as it will open up a lot of options for refactoring and sharing code (i.e. for HTML sanitization) between the different pipelines via filters. I intentionally stopped before doing this to gauge interest and get commonmark in as quickly as possible. We can (and imho should) rework the Textile rendering at a later point, as well as do some cleanup by moving rendering related code from helpers to the pipeline. The same could be done to the Redcarpet based markdown formatter if we decide to still keep that around at this time.

Configuration

For the sake of not overcomplicating this already complex patch I went with what I think are sensible defaults. I have no idea why Github is so inconsistent with their line break rendering and would definitely stay away from doing the same.

Introducing a way for configuring the rendering pipeline through plugins makes sense, maybe as part of the refactoring when we move over all formatters to the pipeline?

Regarding the task lists - to be honest I left this out since at Planio this would collide with the checklists plugin we have as a standard feature. Also this would require additional code to handle the checking / unchecking of the checkboxes in the rendered text (everywhere markdown is rendered) again making the patch more complex. I feel like this also really only makes sense in issue descriptions and (maybe) wiki pages. So this would be a case where we need slightly different pipeline config depending on the context.

#15 - 2019-12-11 16:19 - Hans Riekehof

Oh my this patch would be so nice. I know its always hard to say but is there any roadmap when this patch is available in an official release ?

#16 - 2019-12-12 01:01 - Mischa The Evil

Hans Riekehof wrote in #note-15:

> *[...] I know its always hard to say but is there any roadmap when this patch is available in an official release ?*

Given all the parameters (size/scope/state/complexity of the patch, current Redmine release cycle, current issue scheduling) I'd say not earlier than before the end of 2020. However, this does not mean that it won't be available on the Redmine trunk earlier.

NB: Please keep this issue clean, focused, on-topic and free from superfluous comments...

#17 - 2019-12-12 11:08 - Go MAEDA

Thank you for posting the patch.

I think we should remove Redcarpet instead of adding the second Markdown formatter. Although I understand that CommonMaker is not fully compatible with Redcarpet, having two different Markdown formatter introduces some problems.

- It is confusing. Maybe some users cannot understand why Redmine has two Markdown formatter and cannot determine which they should use
- Consumes more memory

Someday after adding support for CommonMaker, we need to remove Redcarpet. In my opinion, when the time we add CommonMaker is the best

chance to remove Redcarpet. Replacing Redcarpet with CommonMaker is a simple and understandable story for everyone.

#18 - 2020-02-10 09:18 - Jens Krämer

I agree regarding complexity / confusion of users. So in order to move this forward, should I proceed to change the patch so it directly replaces Redcarpet with CommonMarker for the markdown formatting?

#19 - 2020-02-10 10:53 - Marius BALTEANU

Jens Krämer wrote:

> *I agree regarding complexity / confusion of users. So in order to move this forward, should I proceed to change the patch so it directly replaces Redcarpet with CommonMarker for the markdown formatting?*

I'm my opinion, it is a good move to replace Redcarpet with CommonMarker, but I don't think that we can do it without helping users to migrate their existing content. Gitlab did this kind of migration and they had a very nice process:
https://about.gitlab.com/blog/2019/06/13/how-we-migrated-our-markdown-processing-to-commonmark/

#20 - 2020-02-10 11:43 - Jens Krämer

Well Gitlab is a hosted platform with paying customers, so they had to be extra careful. I don't know if/how that gradual release of commonmark also happened for users of their opensource version on their own servers. From reading that article it appears to me that they did not encounter many issues with existing content since they do not mention having done any automated conversions of Redcarpet data to commonmark.

It would certainly up complexity quite a bit if we were to add support for different formatters at the same time (i.e., rendering old content with RedCarpet and only new content with CommonMarker)...

Gitlab's diff tool could indeed be used as inspiration for a plugin that people could install before upgrading that could give them an idea about what percentage of their contents will render differently at all if they were to upgrade. I wouldn't build an automated converter however - it's really just few corner cases and chances are such a tool breaks more than it fixes.

#21 - 2020-02-21 14:27 - Martin Cizek

- File 0003-CommonMark-external_links_filter.rb.patch added

Attaching an incremental patch to handle invalid URIs in HTML anchors. It prevents HTTP 500 for inputs like:

    https://example.com:port/yourservice

A few remarks:

- This is similar to how GitHub behaves (it keeps href at the link).
- Such links are not marked as external - not sure if it is right or wrong.
- GitLab treats these cases also with removing the link's href, see sanitize_node_link.rb called at base_sanitization_filter.rb#L45 - I consider this a good approach especially for autolinks, which are still parsed as autolinks, but they are neither blue nor clickable in the end.

#22 - 2020-02-21 14:43 - Go MAEDA

- Related to Feature #33037: Support for Gitlab flavored Markdown added

#23 - 2020-02-21 18:06 - Martin Cizek

Marius BALTEANU wrote:

> *I'm my opinion, it is a good move to replace Redcarpet with CommonMarker, but I don't think that we can do it without helping users to migrate their existing content.*

I believe that keeping the content and just switching the renderer will be the best option for most Redcarpet users. A few reasons for not treating this as a big deal:

- Substantial part of typical rich text contents is quite messy. Users are usually not familiar with the syntax details and they often copy&paste pieces of code, error messages, etc. without enclosing them to code blocks.
- Users often use familiar (i.e. CommonMark) constructs without paying attention to how they exactly render.
- Redmine did breaking changes in text formatting even before. E.g. until 3.4.7, users can write spaces before list items in Textile. In Markdown, superscript syntax was just introduced at some moment and introduction of underline using underscores could have been a surprise too.

I've written a few more thoughts on this topic here.
Formally speaking, conversion will improve things only if the currently rendered output better reflects what was intended to be written than input interpreted as GFM. Which is not the case in my experience. And transforming badly rendered Markdown to GFM would push the rendering problems to the text source.

Disclosure: I have this opinion despite of being the author of a tool which can do the conversion job. :)

Conversion tool:
We've published the redmine_reformat plugin.
It should do a good job for Textile -> Markdown conversion - tested on 250k strings. And it can be also used on common_mark-patched Redmine right away.

The RedcarpetMD -> GFM conversion can be done as MD -> patched Redmine's textilizable() -> HTML -> external Turndown microservice -> GFM (see examples if interested).
Our external Turndown based microservice is still in development and we are hoping for getting close to passing backtranslation tests (MD -> HTML -> MD). Which is quite a complex task and even bulletproof GFM escaping took almost 1000 lines of Javascript source code.
Having said this, I can only repeat that even a RedcarpetMD -> GFM converter with 100% rendering match would be hardly beneficial to most users.

Sorry for another longer post and hope it helps in decision making. :)

#24 - 2020-02-23 16:25 - Martin Cizek

The previous post was a little biased, as we were using patched Redmine's configuration of Redcarpet. The biggest deal in the transition would be the hard_wrap option. Also _underline_ will change meaning to emphasis, and ^superscript^ will not work. I'm still convinced that any eventual conversion would do the best job if it just handles these particular issues at the text source level. We could try to make such converter if it were considered as a dealbreaker.

I'd discourage conversions based purely on pandoc - including the Gitlab's diff tool, which is even not finished according to its source code. :)

#25 - 2020-02-28 03:23 - Kevin Fischer

+1

I like this patch a lot, too, and agree that the HTML Pipeline will make existing code much nicer, modular and easier to maintain.

I would also vote for replacing the current markdown parser.

About the compatibility issues:

- **hard_wrap:** Commonmarker has a render option called HARDBREAKS which has the same effect (and this also seems to be the standard behaviour on GitHub)

- **underline** and **superscript** could just be replaces by their direct HTML equivalents (ins and sup) in the text source

#26 - 2020-03-27 10:26 - Jens Krämer

GitHub uses HARDBREAKS only for rendering markdown in issues and similar (more short form) content. Wiki pages and markdown documents in repositories are rendered without hard line breaks.

Generally not using hard breaks is more sensible if you intend to make your content render properly on a variety of screen widths since it allows paragraphs to flow freely and lines will only end at the end of the available space and not where the original author decided to put a line ending. As such, it is also more important in long form content, less so in shorter snippets of text where it won't make too much of a difference in terms of readability.

I would prefer to not just continue with forcing hard breaks onto all users of Redmine, but let users choose. It's important to not only think about existing content here, but also what would be the optimum for new installations.

We could introduce a global setting (i.e. in configuration.yml) where the person setting up a Redmine installation can select either the old behavior (hard breaks everywhere), no hard breaks, or the Github approach with hard breaks in issues and comments but nowhere else. I do not know the exact reasoning that led Github to their decision to have inconsistent hard break rendering across their site, but it might have to do with having content flowing into issues / comments from emails (which is the case in Redmine as well), and/or with avoiding surprises for users (issue creators / commenters) who are unfamiliar with the way markdown line endings work by default.

To handle the issue with incoming emails (which rarely are written with Markdown in mind and therefore often render poorly without hard line breaks turned on) at Planio, we currently enforce hard breaks in content from emails by appending two spaces to each line of text that goes from an email into an issue or comment. This would not be necessary if we had different rendering modes, i.e. hard breaks on in issues / comments, hard breaks off elsewhere. The current architecture however doesn't really allow for this, the markdown formatter lacks context information. It would be easy to define two different markdown formatting pipelines, one with hard breaks, one without, but still the text formatting view helper has to get some context so further down the chain can be decided which pipeline to use.

#27 - 2020-04-05 12:51 - Marius BALTEANU

Jens Krämer wrote:

> *Well Gitlab is a hosted platform with paying customers, so they had to be extra careful. I don't know if/how that gradual release of commonmark also happened for users of their opensource version on their own servers. From reading that article it appears to me that they did not encounter many issues with existing content since they do not mention having done any automated conversions of Redcarpet data to commonmark.*
>
> *It would certainly up complexity quite a bit if we were to add support for different formatters at the same time (i.e., rendering old content with RedCarpet and only new content with CommonMarker)...*
>
> *Gitlab's diff tool could indeed be used as inspiration for a plugin that people could install before upgrading that could give them an idea about what percentage of their contents will render differently at all if they were to upgrade. I wouldn't build an automated converter however - it's really just few corner cases and chances are such a tool breaks more than it fixes.*

Sorry for my late reply on this.

I thought more to an easy way for users to find the content that must be updated, not to fix it automatically. Anyway, in the meantime, I saw Martin Cizek plugin that has some features that can help users with this change. Even if we do not provide any help for users, I'm still in favour of this move because RedCarped project is almost dead and, also, I think we can improve a lot the actual code if we start use the html_pipeline gem.

Jens, what do you think if we try to plug the html_pipeline gem to wiki formatting (both Textile and CommonMark)? I will be very happy to work on moving the current regexes from ApplicationHelper#parse_redmine_links and other methods that currently do content parsing to html pipeline filters. In the end, we would have almost the same pipeline, with only one difference, Textile or CommonMark filter.

#28 - 2020-04-30 11:07 - Martin Cizek

Hey guys, although Marius already noticed something being cooked, let me announce it officialy now.

redmine_reformat now provides the MarkdownToCommonmark converter, which treats the biggest Redcarpet and GFM differences, i.e. hard line wraps, underscore _underlines_ and caret ^superscripts. It works solely by editing source markup, so it does not have the adverse effects of "full conversion" I mentioned above. On the other hand, it involves Markdown parser and recognizes Macros when locating places to edit, so it should be decently precise.

More info on MarkdownToCommonmark here. Basic usage is as easy as:
rake reformat:convert to_formatting=common_mark

Jens Krämer wrote:

> *global setting [...] select either the old behavior (hard breaks everywhere), no hard breaks, or the Github approach with hard breaks in issues and comments but nowhere else.*

+1. And actually one of the secret reasons for creating the converter was a fear that the migration issue could be "solved" just by hardcoding the HARDBREAKS setting.

As for the user settings, the plugin already supports migration to mixed format setups. It can choose different conversion chains per object type. The configs for all three mentioned options are described here.

Jens Krämer wrote:

> *I wouldn't build an automated converter however*

Sorry :)

> *it's really just few corner cases*

Actually the hard breaks can be quite a pain.

> *chances are such a tool breaks more than it fixes.*

...unless there is an approach that cannot break things by design. Bets accepted. :)

If there is anything we can do to increase your confidence, let me know.

#29 - 2020-04-30 11:08 - Martin Cizek

We have also built **Docker images** including the common_mark formatting: orchitech/redmine-gfm.

It's based on official Docker images and the patches attached to this ticket. Can be used to try it out, but I consider it production-ready. Without warranty indeed.

#30 - 2020-05-22 09:02 - Mischa The Evil

- Related to Defect #22323: Markdown newline rendering broken added

#31 - 2021-03-17 22:02 - Marius BALTEANU

- Target version changed from 4.2.0 to 5.0.0

It's a big change, we cannot do it in version:"4.2.0" which is planned to be released as soon as possible.

#32 - 2021-03-31 08:39 - Marius BALTEANU

What do you think about the following plan for this issue?

- We introduce this as third text formatting option in version:"5.0.0".
- We mark the existing Markdown as deprecated in the same version.
- We remove the existing Markdown version in a future version (5.2.0 or maybe even 6.0)?

In this way, users will have enough time to update their content and make the transition to commonmark smoother.

#33 - 2021-03-31 08:44 - Bernhard Rohloff

Marius BALTEANU wrote:

> *What do you think about the following plan for this issue?*
> *...*

I really like this approach. IMHO making the transition while getting usual stuff done in production is a good option for many users.

#34 - 2021-04-05 16:32 - Martin Cizek

- File 0004-CommonMark-formatter_test.rb.patch added

#35 - 2021-04-05 16:36 - Martin Cizek

Martin Cizek wrote:

> *We have also built **Docker images** including the common_mark formatting: orchitech/redmine-gfm.*

Announcement for those who want to check out the CommonMark formatter:
After releasing Redmine 4.2, our Docker images orchitech/redmine-gfm were automatically released for Redmine 4.2 as well.

#36 - 2021-04-05 17:20 - Martin Cizek

Marius BALTEANU wrote:

> *What do you think about the following plan for this issue?*

It totally makes sense to release the new format ASAP and have a transition period.

Are you aware of any tasks that need to be done before the patches are applied (without removing the Redcarpet format)? I'd be happy to help... By now, we have tested it quite extensively and provided the incremental patches attached to this ticket. I actually don't see any reason why it cannot be added in e.g. Redmine 4.3. Especially when our automatic CI/CD was able to release Redmine-GFM 4.2 without any human touch. :-)

However, there is another thing that might deserve to come with a major release - it's the transition to HTML::Pipeline, which might or might not be released together with Redcarpet removal. I created #35035 for it (please feel free to link it to this issue).
So my alternative proposal would be:

- We introduce this as third text formatting option in 4.3.0 (or any release that comes soon). It can be marked as experimental for the very first release if there is not enough confidence.
- We refactor the text formatting to HTML::Pipeline in 5.0.0
- We mark the existing Markdown as deprecated in the same version - thus it doesn't have to be subject of full refactoring to HTML::Pipeline.
- We remove the existing Markdown version in a future version (5.2.0 or maybe even 6.0)?

In this way, users will have even more time to update their content, they will also have extra motivation to do it (benefits described in #35035) and Redmine text formatting subsystem will be pushed in the right direction at the same time.

What do you think?

#37 - 2021-04-08 12:37 - Jens Krämer

great to see this moving forward :)

I think the addition of CommonMark as an "experimental" third formatting really soon is a good idea. Since it would be an entirely new feature without breaking changes this could be done in 4.3 or 5.0 (in case no 4.3 is planned).

Refactoring Textile to use the HTML pipeline shouldn't be something any user would notice, so we could do this anytime after the pipeline is in. I'd be happy to help with that.

The original Markdown formatter should be marked deprecated the moment we remove the "experimental" label from the CommonMark formatter (probably this would happen in the next minor release after it's introduction, depending on any issues with it that might come up). Since that will then most probably be a 5.x, we should imho wait until 6.0 before finally removing it. Users don't like changes that require some migration / effort on their part being forced onto them ever, so I think we should not rush it.

That however means we'll have the 'legacy markdown' around for quite some time and it might pay off to integrate it in the HTML pipeline as well in the same way we will do it with Textile. Otherwise we would have to carry around two versions of all the related wiki formatting helpers that we plan to move from ApplicationHelper to the pipeline until then. This would also make it trivial to later push it into a plugin for people who don't want to migrate or switch to commonmark for whatever reason at all.

#38 - 2021-04-08 23:52 - Martin Cizek

Jens Krämer wrote:

>

> *Refactoring Textile to use the HTML pipeline shouldn't be something any user would notice, so we could do this anytime after the pipeline is in. I'd be happy to help with that.*

Maybe if you don't mind reviewing/elaborating on #35035? I've created it to give some momentum to the pipeline adoption, but the description is quite abstract and maybe even naive. So your further clarification would improve credibility of the ticket. :)

Jens Krämer wrote:

> *That however means we'll have the 'legacy markdown' around for quite some time and it might pay off to integrate it in the HTML pipeline as well in the same way we will do it with Textile.*

Makes sense. Looking at #32424-32 + #32424-36 + #32424-37, *I guess we are in agreement* ... Marius?

#39 - 2021-04-15 22:17 - Martin Cizek

- File 0005-CommonMark-unify-code-blocks-35104.patch added

Adding patch to unify code block rendering across formatters, see #35104. It also adjusts language name regexp to a sequence of non-whitespace. The previous regexp was too strict even for currently supported languages - it didn't allow "C++". :)

#40 - 2021-04-16 11:49 - Heiko Robert

Thank you all for your valuable work in providing this patch!
I have issues with non http links using protocols like ssh or scp. We use lot's of them in our wiki pages. Following the commonmark spec for Autolink something like <ssh://user@test.mycompany.local> should work but actually does not.
Is this an issue of the library you embed or is there a work around to create links using other schemes than http?

#41 - 2021-04-16 12:31 - Heiko Robert

I found your protocol whitelist definition in lib/redmine/wiki_formatting/common_mark/sanitization_filter.rb but that seems not to work as expected. Even when trying to use ftp which is in your whitelist no link will be rendered

```
[test](ftp://test.mycompany.com)
ftp://test.mycompany.com
```

#42 - 2021-04-16 13:37 - Heiko Robert

found the issue:
in attachment:0001-CommonMark-Markdown-text-formatter.patch you used :a instead of "a" in line 522
changing to

```
whitelist[:protocols]["a"] = [
```

fix's it.

I would suggest to either add common protocols like 'scp', 'ssh', 'tel', 'call' or make the protocols configurable not to force patching if someone needs other protocols.

#43 - 2021-04-16 16:44 - Martin Cizek

Heiko Robert wrote:

> ~~*whitelist[:protocols]["a"] = [*~~

Please be warned, this would allow everything, including user-supplied javascript.
The correct fix with the allowlist approach would be:

```
whitelist[:protocols]["a"]["href"] = [
```

> *add common protocols like 'scp', 'ssh', 'tel', 'cal*

It seems that the current Textile formatter only forbids 'javascript'. And I've checked that e.g. GitLab only removes unsafe schemes too. So we probably should go that way. I'll try to prepare an incremental patch for it.

> *or make the protocols configurable not to force patching*

This would probably better fit in #35035 (if it was still relevant). To be honest, I'd happy to see at least anything merged anytime soon.

#44 - 2021-04-17 19:39 - Martin Cizek

- File 0006-CommonMark-remove-extensive-link-target-restriction.patch added

The problem was that the overly restricted href URL issue was actually copied from the old Markdown implementation (already reported in #32766). We should distinguish URI safety for automatically loaded resources (like images) and harmlessness to just render a link.

- Old Markdown uses Redmine::Helpers::URL#uri_with_safe_scheme? for both decisions.
- Textile uses Redmine::Helpers::URL#uri_with_safe_scheme? for images and a hardcoded test ref.downcase.start_with?('javascript:') for link hrefs.

That's why I've introduced Redmine::Helpers::URL#uri_with_link_safe_scheme? to determine safety of only-rendered, user-activated links. The change could have been briefer, but I've used a Sanitize transformer, as Sanitize authors suggested to us.

#45 - 2021-04-17 20:27 - Martin Cizek

- File 0007-CommonMark-update-Sanitize.patch added

Replaced the now-deprecated Sanitize keywords and updated version references. HTML::Pipeline version must be specified including its patch version because of this (2.14.0 is older than 2.13.2).

#46 - 2021-04-23 03:09 - Jens Krämer

Martin Cizek wrote:

> *The problem was that the overly restricted href URL issue was actually copied from the old Markdown implementation (already reported in #32766). We should distinguish URI safety for automatically loaded resources (like images) and harmlessness to just render a link. - Old Markdown uses Redmine::Helpers::URL#uri_with_safe_scheme? for both decisions.*
> *- Textile uses Redmine::Helpers::URL#uri_with_safe_scheme? for images and a hardcoded test ref.downcase.start_with?('javascript:') for link hrefs.*
>
> *That's why I've introduced Redmine::Helpers::URL#uri_with_link_safe_scheme? to determine safety of only-rendered, user-activated links. The change could have been briefer, but I've used a Sanitize transformer, as Sanitize authors suggested to us.*

I like that approach.

Thanks for pointing out the bug with the allow list and the correct form `whitelist[:protocols]["a"]["href"] = ...`.

#47 - 2021-04-27 22:55 - Martin Cizek

- File 0008-attachments_controller_test.rb.patch added

I believe we're 100% done now.

rake test @ orchitech/redmine-gfm (patch set):

```
Finished in 502.676237s, 10.3387 runs/s, 47.2869 assertions/s.
5197 runs, 23770 assertions, 0 failures, 0 errors, 9 skips
```

My personal advice to any concerned users would be that it is safer to migrate to common_mark than staying on the current markdown.

#48 - 2021-04-27 23:36 - Martin Cizek

Marius, I'd like to ask you to asses and merge this ASAP to master / trunk. Jens did an excellent job and I tried hard too, the result apparently meets the quality level set in Redmine and even the 4-eyes principle apparently happened.

On one hand, the current implementation is outdated, with many unresolved and/or unresolvable issues and e.g. #32563 might be even a security vulnerability. The implementation was indeed good at the time it was written, but both the software components and user expectations have evolved. It's pretty safe to assume the user expectations being set by GitHub and GitLab, but even JIRA is currently moving towards CommonMark flavour of Markdown.

On the other hand, not having the new implementation in the repo means that I can't contribute to related things (or I have to create patches against "expected future state" - which I do and it's quite frustrating). It also means that the strategic work on HTML::Pipeline is now blocked.

Any feedback would be greatly appreciated and please do make use of the contributor power. :)

#49 - 2021-04-28 20:58 - Marius BALTEANU

Martin Cizek wrote:

> *Marius, I'd like to ask you to asses and merge this ASAP to master / trunk. Jens did an excellent job and I tried hard too, the result apparently meets the quality level set in Redmine and even the 4-eyes principle apparently happened.*

> *On one hand, the current implementation is outdated, with many unresolved and/or unresolvable issues and e.g. #32563 might be even a security vulnerability. The implementation was indeed good at the time it was written, but both the software components and user expectations have evolved. It's pretty safe to assume the user expectations being set by GitHub and GitLab, but even JIRA is currently moving towards CommonMark flavour of Markdown.*
>
> *On the other hand, not having the new implementation in the repo means that I can't contribute to related things (or I have to create patches against "expected future state" - which I do and it's quite frustrating). It also means that the strategic work on HTML::Pipeline is now blocked.*
>
> *Any feedback would be greatly appreciated and please do make use of the contributor power. :)*

I agree with you, we should have this committed as soon as possible, but unfortunately, it's not only my decision, we need to have green light from Jean-Philippe about this and in general, about all major changes. I will do my best to push this forward, I think I can review the changes in May. As a side note, I don't even have rights to commit code to Redmine core :)

#50 - 2021-05-13 01:35 - Go MAEDA

- Related to Feature #32766: Remove the URI limitation from external markdown links added

#51 - 2021-07-04 12:37 - Marius BALTEANU

- File 0012-Fixes-Replace-class-var-allowlist-with-a-class-insta.patch added
- File 0011-Render-markdown-attachments-using-markdown-or-common.patch added
- File 0010-Fixes-failing-test-on-CommonMark-by-striping-the-tra.patch added
- File 0009-Fixes-Layout-HeredocIndentation-Use-2-spaces-for-ind.patch added
- File 0008-Replace-deprecated-Sanitize-keywords-32424.patch added
- File 0007-Relax-allowed-protocols-in-links-by-denying-specific.patch added
- File 0006-Fixes-HTTP-500-error-when-invalid-URIs-are-provided-.patch added
- File 0005-Mark-CommonMark-Markdown-GitHub-Flavored-as-experime.patch added
- File 0004-Pin-html-pipeline-to-2.13.2-32424.patch added
- File 0003-Pin-sanitize-to-5.2-32424.patch added
- Assignee set to Marius BALTEANU

First of all, thanks Jens and Martin for the amazing work, the patches looks good and all the notes are very useful.

Second of all, I've reviewed all the patches and I tried to keep the changes as small as possible in order to have this added to the core and work after that on top of this. I'm attaching the final result with 12 patches that I would like to commit:

1. attachment:0001-Adds-CommonMark-Markdown-GitHub-Flavored-as-third-te.patch
The initial patch posted by Jens, but with all Rubocop warnings fixed and the fix for allow list bug.

2. attachment:0002-Pin-commonmarker-to-0.22-32424.patch , attachment:0003-Pin-sanitize-to-5.2-32424.patch and attachment:0004-Pin-html-pipeline-to-2.13.2-32424.patch
Bumping up the dependencies.

3. attachment:0005-Mark-CommonMark-Markdown-GitHub-Flavored-as-experime.patch
Marks the feature as experimental in the same way as was done for Markdown.

4. attachment:0006-Fixes-HTTP-500-error-when-invalid-URIs-are-provided-.patch
The patch posted by Martin in #32424#note-21.

5. attachment:0007-Relax-allowed-protocols-in-links-by-denying-specific.patch
The patch posted by Martin in #32424#note-44 with a few changes for Rubocop.

6. attachment:0008-Replace-deprecated-Sanitize-keywords-32424.patch
The patch posted by Martin in #32424#note-45.

7. attachment:0009-Fixes-Layout-HeredocIndentation-Use-2-spaces-for-ind.patch
Another Rubocop fix to better present the fix for a failing test.

8. attachment:0010-Fixes-failing-test-on-CommonMark-by-striping-the-tra.patch
test_footnotes fails because the footnote block is returned with a trailing whitespace, I'm not sure why, but I've fixed it by using rstrip in the actual output.

```
ruby test/unit/lib/redmine/wiki_formatting/common_mark/formatter_test.rb -n test_footnotes

Run options: -n test_footnotes --seed 52357

# Running:

F

Failure:
Redmine::WikiFormatting::CommonMark::FormatterTest#test_footnotes
[test/unit/lib/redmine/wiki_formatting/common_mark/formatter_test.rb:153]:
--- expected
+++ actual
@@ -1 +1 @@
-"<p>This is some text<sup><a href=\"#fn1\" id=\"fnref1\">1</a></sup>.</p> <ol><li id=\"fn1\"><p>This is the foot note <a
href=\"#fnref1\">□ </a></p></li></ol>"
+"<p>This is some text<sup><a href=\"#fn1\" id=\"fnref1\">1</a></sup>.</p> <ol><li id=\"fn1\"><p>This is the foot note <a
href=\"#fnref1\">□ </a></p></li></ol> "

rails test test/unit/lib/redmine/wiki_formatting/common_mark/formatter_test.rb:137
```

9. attachment:0011-Render-markdown-attachments-using-markdown-or-common.patch
Slightly changed the patch posted by Martin in #32424#note-7 in order to render the markdown attachment using common_mark only when common_mark it's used, otherwise, to fall back on the old markdown.

10. attachment:0012-Fixes-Replace-class-var-allowlist-with-a-class-insta.patch
Another fix for Rubocop which I think it's safe. Martin, Jens, please let me know what do you thing.

On my CI environment, all the tests pass, except some flaky tests caused by some recent changes in the trunk.

Martin, regarding attachment:0005-CommonMark-unify-code-blocks-35104.patch, I think we can discuss it later on (#35104).
If everything is ok:
- version:"5.0.0": we deliver this as experimental
- next feature release (5.1): we remove the experimental flag, make CommonMark default formatter and mark the old markdown as deprecated
- next major release (Redmine 6.0?): we remove the deprecated markdown and we automatically migrate the formatter setting to common_mark without any content migration.

#52 - 2021-07-04 12:38 - Marius BALTEANU

- File 0002-Pin-commonmarker-to-0.22-32424.patch added
- File 0001-Adds-CommonMark-Markdown-GitHub-Flavored-as-third-te.patch added

#53 - 2021-07-07 07:32 - Jens Krämer

Awesome, thanks for your work to bring it all together :)

#54 - 2021-07-08 17:14 - Go MAEDA

It is great that a new, more expressive Markdown formatter will be available in the next version of Redmine. Thank you to everyone involved in the development of this patch.

However, there is only one concern. That is HARDBREAKS.

- As mentioned in #32424#note-26, issues created from incoming emails often have layout problems
- Some users tend to use hard breaks a lot when writing issues. It is difficult to convince them to learn the concept of paragraphs and how to break lines in CommonMark. This is because they are not writing Markdown, they are just writing text
- As far as I know, in Japan, hard breaks tend to be used a lot in issues, and if hard breaks are not available, it can cause a lot of frustration for some of them

For organizations that need hard breaks, we hope that one of the following measures will be taken:

- A setting to switch the hardbreak mode on /settings page
- Add another option "CommonMark Markdown with hard breaks" to the "Text formatting" setting

If there is no option to enable hard breaks, I am concerned that this will be a big problem in Japan.

#55 - 2021-07-17 09:01 - Marius BALTEANU

I think we should conclude over the HARDBREAKS before committing this.
To summarize:
- using HARDBREAKS or not is still CommonMark compliant, the spec covers both implementations.

> *A renderer may also provide an option to render soft line breaks as hard line breaks.*

https://spec.commonmark.org/0.17/#soft-line-breaks
- as Jens mentioned, the real advantage of not using HARDBREAKS is on long form content where the user can easily control when to break the line or not.
- as Go Maeda mentioned in his last comment, forcing the users to learn a special syntax (two spaces or \) for such a simple operation (line break) could be a real issue because Redmine, compared with Gitlab/Github, is not used only on tech side. Even in a tech company, the users who work on operational teams (ex: sales, finance, marketing) will not easily learn this special syntax when they just want to create a simple ticket. For iOS/OSx users, adding two spaces can be a real trick because of the feature that replaces the two spaces with a dot.

My proposal is the following:
- by default, to support HARDBREAKS using the regular line break.
- have an option in configuration.yml to disable this behaviour and use the special syntax to line breaks. I'm not in favour of having a mix behaviour as Github does.

If the feature of controlling when to break the line is so useful, we can do a Redmine flavour and add a special syntax that continues the line instead of breaking (something like "&" or "&&"). I'm not sure if it's the best ideea, but it's an option.

#56 - 2021-07-18 03:49 - Go MAEDA

Marius BALTEANU wrote:
>

> *My proposal is the following: - by default, to support HARDBREAKS using the regular line break.*
> *- have an option in configuration.yml to disable this behaviour and use the special syntax to line breaks. I'm not in favour of having a mix behaviour as Github does.*

I agree. Enabling HARDBREAKS by default removes the barrier to switching from RedCarpet to CommonMark.

On that basis, I have a few suggestions:
1. With HARDBREAKS enabled by default, the new CommonMark based Markdown is highly compatible with the current RedCarpet based Markdown. Maybe we can give the CommonMark base the name Markdown "Markdown" (Setting.text_formatting: markdown) and rename the current RedCarpet based Markdown to "Markdown (classic)" (Setting.text_formatting: markdown_redcarpet). This means users will start using CommonMark without being aware of it after upgrading to Redmine 5.0. Only people who experience some issue will switch the setting to "Markdown (classic)". Or we can completely remove RedCarpet in Redmine 5.0, as we switched the syntax highlighter from CodeRay to Rouge (JPL didn't want keeping two libraries. See #24681#note-23).
2. I think it will be even better if an admin can change the HARDBREAKS behavior in the /admin/settings page instead of configuration.yml. This is because Redmine admins don't always have access to configuration files on a Redmine server (e.g. companies which have a dedicated information system department, users using SaaS based Redmine)

#57 - 2021-07-27 01:16 - Marius BALTEANU

Go MAEDA wrote:

> *I agree. Enabling HARDBREAKS by default removes the barrier to switching from RedCarpet to CommonMark.*
>
> *On that basis, I have a few suggestions:*
> *1. With HARDBREAKS enabled by default, the new CommonMark based Markdown is highly compatible with the current RedCarpet based Markdown. Maybe we can give the CommonMark base the name Markdown "Markdown" (Setting.text_formatting: markdown) and rename the current RedCarpet based Markdown to "Markdown (classic)" (Setting.text_formatting: markdown_redcarpet). This means users will start using CommonMark without being aware of it after upgrading to Redmine 5.0. Only people who experience some issue will switch the setting to "Markdown (classic)". Or we can completely remove RedCarpet in Redmine 5.0, as we switched the syntax highlighter from CodeRay to Rouge (JPL didn't want keeping two libraries. See #24681#note-23).*

I don't think that it's the same case with the syntax highlighter. In this case, there are certain situation where the content must be prepared for common_mark before migration. I'm more comfortable to stick with the initial plan that will give us time to prepare the migration: introduce this an experimental feature, deprecate old markdown, migrate markdown to common_mark in a later version.

> *1. I think it will be even better if an admin can change the HARDBREAKS behavior in the /admin/settings page instead of configuration.yml. This is because Redmine admins don't always have access to configuration files on a Redmine server (e.g. companies which have a dedicated information system department, users using SaaS based Redmine)*

We can add this in admin/settings, but changing the setting will still require an application restart in order to become effective.

#58 - 2021-07-27 01:30 - Marius BALTEANU

- File demo_hardbreaks_setting.patch added

Marius BALTEANU wrote:

> *We can add this in admin/settings, but changing the setting will still require an application restart in order to become effective.*

I made a patch for testing purposes. If we choose to keep this setting in the administration page, we should add a notice to inform the administrator that a restart is required.

#59 - 2021-07-27 06:37 - Go MAEDA

Marius BALTEANU wrote:

> *I don't think that it's the same case with the syntax highlighter. In this case, there are certain situation where the content must be prepared for common_mark before migration. I'm more comfortable to stick with the initial plan that will give us time to prepare the migration: introduce this an experimental feature, deprecate old markdown, migrate markdown to common_mark in a later version.*

Your opinion makes sense.

> *1. I think it will be even better if an admin can change the HARDBREAKS behavior in the /admin/settings page instead of configuration.yml. This is because Redmine admins don't always have access to configuration files on a Redmine server (e.g. companies which have a dedicated information system department, users using SaaS based Redmine)*
>
> *We can add this in admin/settings, but changing the setting will still require an application restart in order to become effective.*

I didn't know that a restart is required. If that is the case, I think put the setting in configuration.yml as you suggested is the best way. Some Redmine admins who don't have access to the underlying OS cannot restart Redmine by themselves.

#60 - 2021-07-28 00:19 - Marius BALTEANU

Attached a WIP patch for the configuration setting and now we have both options available for testing. Go Maeda, Jens Krämer, Martin, what do you think about this?

Jens Krämer wrote:

> *To handle the issue with incoming emails (which rarely are written with Markdown in mind and therefore often render poorly without hard line breaks turned on) at Planio, we currently enforce hard breaks in content from emails by appending two spaces to each line of text that goes from an email into an issue or comment.*

Do you think it's possible to extract this change as a patch? I think we should do the same thing.

#61 - 2021-07-28 00:19 - Marius BALTEANU

- File configuration-setting-for-hardbreaks.patch added

#62 - 2021-08-10 23:01 - Marius BALTEANU

Rails 6.1 and Redmine version:"5.0.0" supports minimum Ruby 2.5, but the CommonMarker latest version (0.22.0) supports minimum Ruby 2.6.

We should drop support for Ruby 2.5 or we should stick with commonmarker version 0.21.0 if ruby is 2.5? Note that Ruby 2.5 is EOL since March this year.

#63 - 2021-08-11 17:26 - Holger Just

I would be fine with requiring Ruby 2.6 for Redmine 5.0.

Some notes here:

- Debian 11 Bullseye is going to ship with Ruby 2.7 on August 14.
- Ubuntu >= 20 ships with Ruby 2.7
- Debian 10 (Buster) and Ubuntu 18.04 (bionic) still ship Ruby 2.5. Users would have to either use a custom Ruby or stay on Redmine 4.2
- CentOS 8 / Redhat 8 can install Ruby 2.7 with dnf.
- For Windows, there are usable installers for current Rubies.

With this change, we would thus cut off Debian 10 and Ubuntu 18.04 with their respective system-default Rubies. They can install a custom Ruby with rvm / rbenv / ruby-install though.

#64 - 2021-08-11 19:16 - Marius BALTEANU

Thanks Holger for you reply, let's move the discussion about Ruby 2.5 support to #31128 (I've reopen it and I added our comments).

#65 - 2021-08-12 00:13 - Marius BALTEANU

- File 0001-Add-setting-to-controll-the-hardbreaks-behaviour-fro.patch added

To move this forward I've committed the patch series 1-12 with slight changes:

- Hardbreaks are enabled by default, but we still have to make it configurable (please view below).
- commonmarker gem version is pinned to 0.21.0 if Ruby is 2.5.

We still have the following 2 points open and they need to be addressed in the following days:

1. Make the hardbreaks configurable: I'm attaching the patch to obtain this, please take a look and tell me what you think.
2. Handle the incoming emails when hardbreaks are disabled. @Jens, @Holger, is it possible to extract the patch that you mentioned? Or I should start to work on this?

Thanks again for the hard working on this, I'm sure that will be one of the Redmine version:"5.0.0" highlights.

For other issues/improvements, feel free to open new tickets. I already open a new issue for the task list items (#35742).

#66 - 2021-08-12 00:14 - Marius BALTEANU

- Related to Feature #35742: Enable task list items for Common Mark text formatting added

#67 - 2021-08-12 09:14 - Go MAEDA

- Related to Feature #35747: Allow style attribute for HTML elements in CommonMark formatter added

#68 - 2021-08-12 23:05 - Marius BALTEANU

- Related to deleted (Defect #22323: Markdown newline rendering broken)

#69 - 2021-08-12 23:05 - Marius BALTEANU

- Duplicated by Defect #22323: Markdown newline rendering broken added

#70 - 2021-08-13 03:28 - Go MAEDA

- Related to Defect #35752: Warning message "nokogumbo: Using Nokogiri::HTML5 provided by Nokogiri" added

#71 - 2021-08-13 04:31 - Go MAEDA

- Related to Defect #35754: Redmine::WikiFormatting::CommonMark::FormatterTest#test_should_ignore_soft_breaks fails added

#72 - 2021-08-15 12:46 - Marius BALTEANU

- Related to Feature #35035: Refactor text formatting to HTML::Pipeline added

#73 - 2021-08-15 13:16 - Marius BALTEANU

- Related to Defect #35765: Code with unsupported code language is not render when CommonMark Markdown is used added

#74 - 2021-08-15 14:36 - Marius BALTEANU

- Related to Patch #35104: Code blocks - consistent rendering and retaining user-supplied language name in rendered HTML added

#75 - 2021-08-22 12:07 - Mischa The Evil

It's really nice to see the progress of this issue. :thumbsup:
May I propose the following tentative but more concrete deprecation and removal schedule:

- 5.0.0
  - Introduce the new formatter as experimental and point to users to Martin's amazing *redmine_reformat*.
- 5.1.x-5.x.x
  - Remove experimental flag and direct users to *redmine_reformat*.
- 6.0.0
  - Deprecate the old RedCarpet-based formatter and push users to *redmine_reformat*.
- 6.1.0-6.x.x
  - Finally remove the old RedCarpet-based formatter.

#76 - 2021-09-04 13:25 - Marius BALTEANU

- File 0001-Add-setting-in-admin-to-control-the-hardbreaks-behav.patch added
- Assignee changed from Marius BALTEANU to Jens Krämer

I made a new patch that allows you to control the hardbreaks behaviour from admin interface without requiring application restart.
Jens (or some else from Plan.io team), being the original author of this patch, please let me know what do you think about the two solutions that I posted for controlling the behaviour of hardbreaks:

1. #32424#note-65: which allows you to control the settings from configuration.yml.
2. the attached patch which allows you to control the setting from the administration page.

After we conclude on this, I would like to fix the incoming emails when the hardbreaks is disabled and to close this issue in the following two weeks.

#77 - 2021-09-09 15:39 - Jens Krämer

Marius BALTEANU wrote:

> *Jens (or some else from Plan.io team), being the original author of this patch, please let me know what do you think about the two solutions that I posted for controlling the behaviour of hardbreaks: 1. #32424#note-65: which allows you to control the settings from configuration.yml.*
> *2. the attached patch which allows you to control the setting from the administration page.*

I believe this setting really isn't something one would likely change more than once. Either it's decided when setting up a new instance and then kept as is, or it's set when converting the data in an existing instance to 'no hard breaks'. Therefore, I think it is perfectly fine to keep it exclusively in `configuration.yml`.

> *After we conclude on this, I would like to fix the incoming emails when the hardbreaks is disabled and to close this issue in the following two weeks.*

At Planio we use something like this (diff against master, patching cleanup_body in MailHandler):

```
diff --git a/app/models/mail_handler.rb b/app/models/mail_handler.rb
index fd2e25fb5..2a71d57cb 100644
--- a/app/models/mail_handler.rb
+++ b/app/models/mail_handler.rb
@@ -638,6 +638,9 @@ class MailHandler < ActionMailer::Base
       regex = Regexp.new("^(\\p{Space}|>)*(#{ Regexp.union(delimiters) })\\p{Space}*[\\r\\n].*", Regexp::MULTILINE)
       body = body.gsub(regex, '')
     end
+    if Setting.text_formatting == "common_mark"
+      body = Redmine::WikiFormatting::CommonMark::AppendSpacesToLines.(body)
+    end
     body.strip
   end

diff --git a/lib/redmine/wiki_formatting/common_mark/append_spaces_to_lines.rb
b/lib/redmine/wiki_formatting/common_mark/append_spaces_to_lines.rb
new file mode 100644
index 000000000..34ee1553f
--- /dev/null
+++ b/lib/redmine/wiki_formatting/common_mark/append_spaces_to_lines.rb
@@ -0,0 +1,13 @@
+module Redmine
+  module WikiFormatting
+    module CommonMark
+      class AppendSpacesToLines
+        def self.call(string)
+          string&.gsub(/(?<!  |\\)(\r?\n)/, "  \\1")
+        end
+      end
+    end
+  end
+end
```

The check for the hard breaks setting would still have to be added of course, instead of just checking for common_mark formatting. It might be worth to think about putting this into a generic postprocessing step implemented in the WikiFormatting namespace, where we then could delegate to the text format specific implementation. Something like body = Redmine::WikiFormatting.post_process_incoming_mail_body(body), maybe?

#78 - 2021-09-10 08:12 - Marius BALTEANU

- Assignee changed from Jens Krämer to Marius BALTEANU

Thanks Jens for you response, I will commit the changes in the following days.

#79 - 2021-09-18 13:04 - Mischa The Evil

- Related to Feature #35889: Textile and Markdown attachment rendering should support third-party formatters added

#80 - 2021-09-20 09:38 - Go MAEDA

- Related to Defect #35892: Redmine::WikiFormatting::CommonMark::FormatterTest#test_footnotes fails with CommonMarker 0.23.2 added

Files

0001-CommonMark-Markdown-text-formatter.patch	68.5 KB	2019-11-06	Jens Krämer
0002-attachments_helper-commonmark.patch	725 Bytes	2019-12-08	Martin Cizek
0003-CommonMark-external_links_filter.rb.patch	808 Bytes	2020-02-21	Martin Cizek
0004-CommonMark-formatter_test.rb.patch	2.09 KB	2021-04-05	Martin Cizek
0005-CommonMark-unify-code-blocks-35104.patch	5.7 KB	2021-04-15	Martin Cizek
0006-CommonMark-remove-extensive-link-target-restriction.patch	5.8 KB	2021-04-17	Martin Cizek
0007-CommonMark-update-Sanitize.patch	3.72 KB	2021-04-17	Martin Cizek
0008-attachments_controller_test.rb.patch	704 Bytes	2021-04-27	Martin Cizek
0012-Fixes-Replace-class-var-allowlist-with-a-class-insta.patch	1.03 KB	2021-07-04	Marius BALTEANU
0011-Render-markdown-attachments-using-markdown-or-common.patch	2.71 KB	2021-07-04	Marius BALTEANU
0009-Fixes-Layout-HeredocIndentation-Use-2-spaces-for-ind.patch	1.72 KB	2021-07-04	Marius BALTEANU
0010-Fixes-failing-test-on-CommonMark-by-striping-the-tra.patch	1.2 KB	2021-07-04	Marius BALTEANU
0008-Replace-deprecated-Sanitize-keywords-32424.patch	3.63 KB	2021-07-04	Marius BALTEANU
0006-Fixes-HTTP-500-error-when-invalid-URIs-are-provided-.patch	1.19 KB	2021-07-04	Marius BALTEANU
0005-Mark-CommonMark-Markdown-GitHub-Flavored-as-experime.patch	967 Bytes	2021-07-04	Marius BALTEANU
0007-Relax-allowed-protocols-in-links-by-denying-specific.patch	6.48 KB	2021-07-04	Marius BALTEANU
0004-Pin-html-pipeline-to-2.13.2-32424.patch	739 Bytes	2021-07-04	Marius BALTEANU
0003-Pin-sanitize-to-5.2-32424.patch	741 Bytes	2021-07-04	Marius BALTEANU
0002-Pin-commonmarker-to-0.22-32424.patch	733 Bytes	2021-07-04	Marius BALTEANU
0001-Adds-CommonMark-Markdown-GitHub-Flavored-as-third-te.patch	74.5 KB	2021-07-04	Marius BALTEANU
demo_hardbreaks_setting.patch	3.56 KB	2021-07-26	Marius BALTEANU
configuration-setting-for-hardbreaks.patch	4.07 KB	2021-07-27	Marius BALTEANU
0001-Add-setting-to-controll-the-hardbreaks-behaviour-fro.patch	4.57 KB	2021-08-11	Marius BALTEANU
0001-Add-setting-in-admin-to-control-the-hardbreaks-behav.patch	3.89 KB	2021-09-04	Marius BALTEANU