djangocms_reversion2 Documentation

Release 0.2

Blueshoe

Apr 30, 2018

Contents

UNSTABLE

Django-CMS Reversion2 is a plugin for **Django CMS** which aims to provide a revision system for **Django-CMS**.

These are the core features of Django-CMS Reversion2:

- **Create PageVersion:** Revisions for page drafts in given language (only if changes were made see `dirty` flag)
- **Revert to PageVersion:** Reverting to any previous revision of page
- **Trash-Bin:** Moves deleted pages to a hidden PageRoot before really deleting it
- **Batch-mode:** Adds reversion for every page (only for superusers)
- **Page Permissions**: Integrates with djangocms' pagepermissions
- **Multi-editor**: Work on the hidden drafts of PageVersion in order to realize multi-editor workflow

To be implemented (see Issues on Github https://github.com/Blueshoe/djangocms-reversion2/issues)

- Auto-Revisions when reverting from unsaved drafts
- Integration with *divio/djangocms_moderation* once they publish a stable release
- Build a multi-editor djangocms_toolbar and disable buttons that make unwanted changes

CHAPTER 1

Contents

1.1 Implementation

We were trying to solve the page version history topic with a simple and pragmatic fix. The old branch of this repository still contains our reversion2 with the json backend, but that had some structural problems. And we had some new requirements:

1.1.1 Motivation

1. Working on multiple drafts simultaneously
2. Not loosing constraints (any model connected to a plugin was a potential insecurity with the old backend)
3. Performance (serialization and de-serialization comes with performance cost) -> we prefer database cost
4. Avoiding the registration logic of the reversion backend for every models.py

1.1.2 Idea

1. When the user creates a new version of a page the draft is copied to a hidden root node in the page tree
2. This so called `hidden_page` is linked to the `PageVersion` model that keeps a reference to the `visible draft`
3. There is always a PageVersion with the attribute `primary draft`
4. The PageVersion is language specific so a rollback doesn't affect the translations of a page
5. The PageVersions are organized in an independent MP_Tree so they are chainable. This feature allows us to implement branching in a future step

1.2 Installation

As long as this plugin is under development it shall be used as git submodule.

1.2.1 Git submodule

- `git submodule add <url> djangocms_reversion2`
- `git submodule update --init`
- `cd djangocms_reversion2/`
- `pip install -r requirements.txt`
- add the "djangocms_reversion2" to `INSTALLED_APPS`
- `python manage.py migrate`

1.2.2 Pypi package

The current package ist outdated. We will push a package once we have a stable release.

1.3 Usage

(1) You can use this plugin if you want to add a trash bin for pages where all deleted pages will be kept until final deletion.

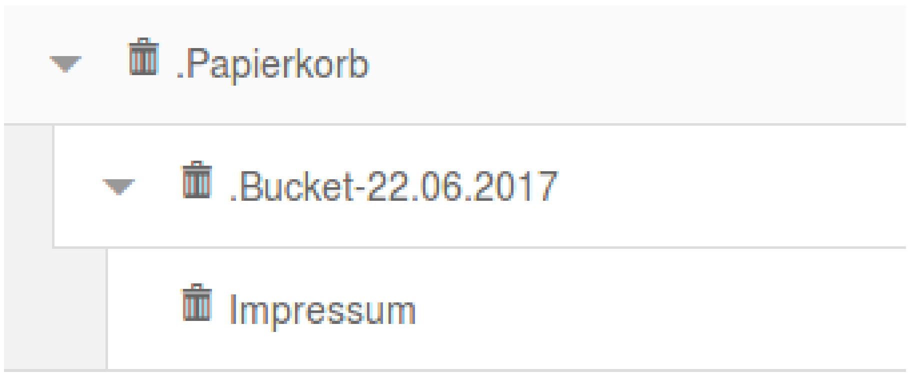
2. You can save breakpoints of pages in order to roll back to one of them. So called PageVersion.

(3) And finally you can work on different versions of a page by creating+editing page versions and in the end setting one of them to the actual HEAD of the page.

The following paragraphs describe example workflows for 1-3.

1.3.1 Page bin

The customer doesn't want pages to be deleted immediately. So we store them under a new root node. To keep the JS tree of the frontend working, the total amount of deleted pages is divided into buckets.

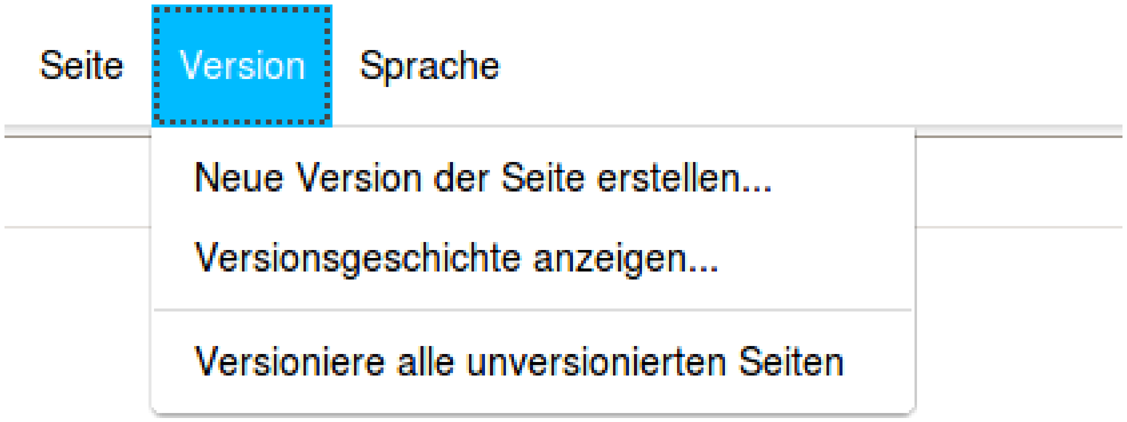


It is possible to drag pages into the bin. We disable all of the buttons so the user cannot interact with the pages. The pages in the bin are not visible because of djangoCMS's policy "unpublished pages must not have published children"

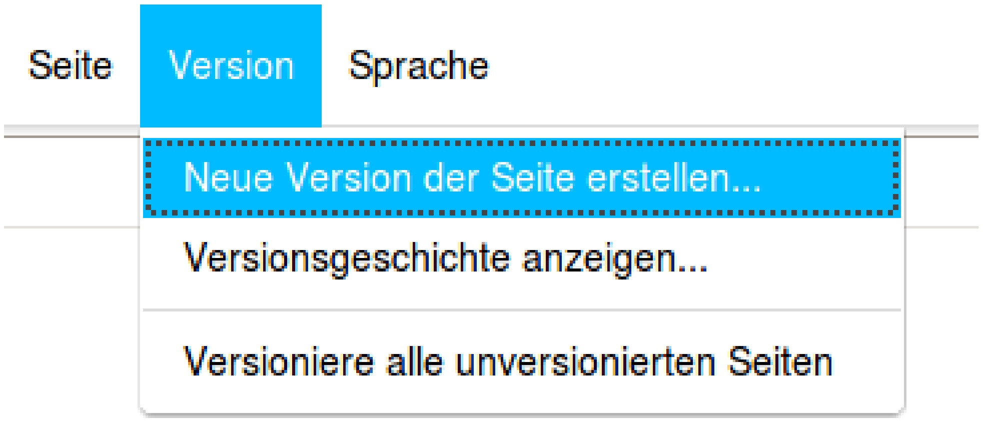
1.3.2 Page versions

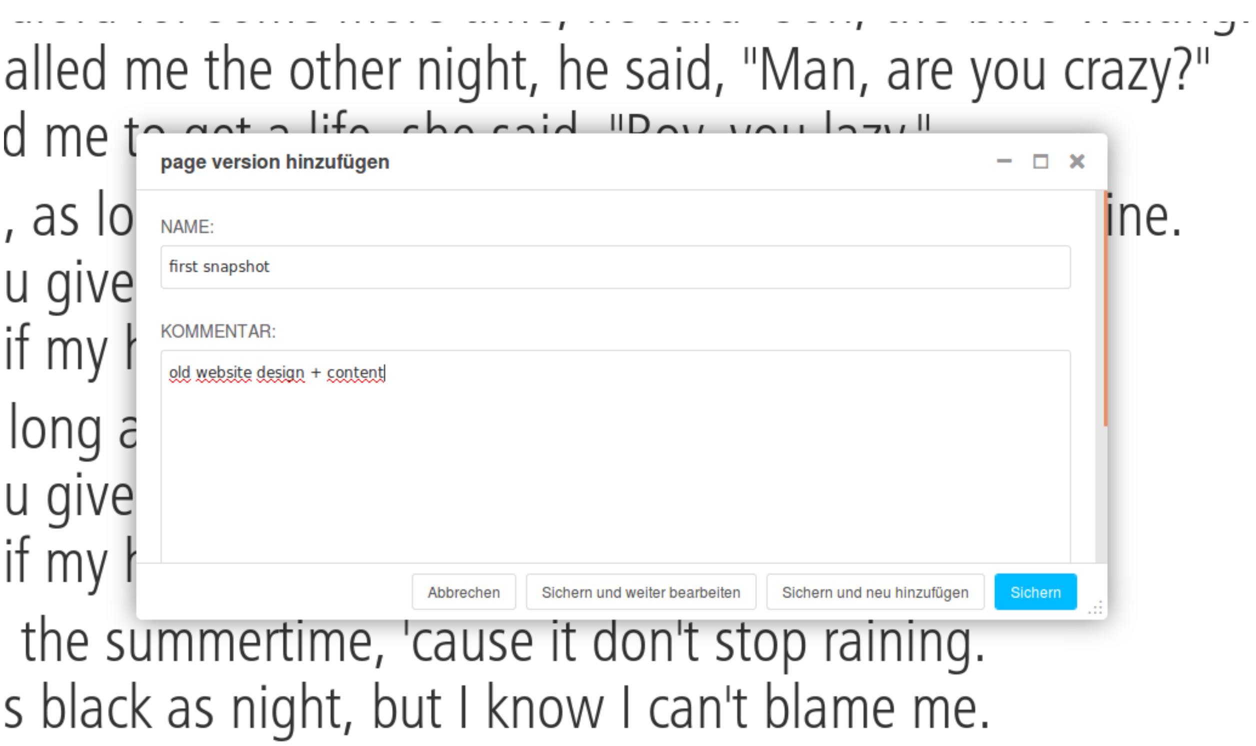
Before editing a page (adding a plugin, changing content. . .) users might want to store the current state of the page. Reversion2 adds a Toolbar menu entry called 'Version' with the following options:

- create a new page version
- show the history (all page versions that belong to the page)
- create a page version for all pages that don't have a single page version

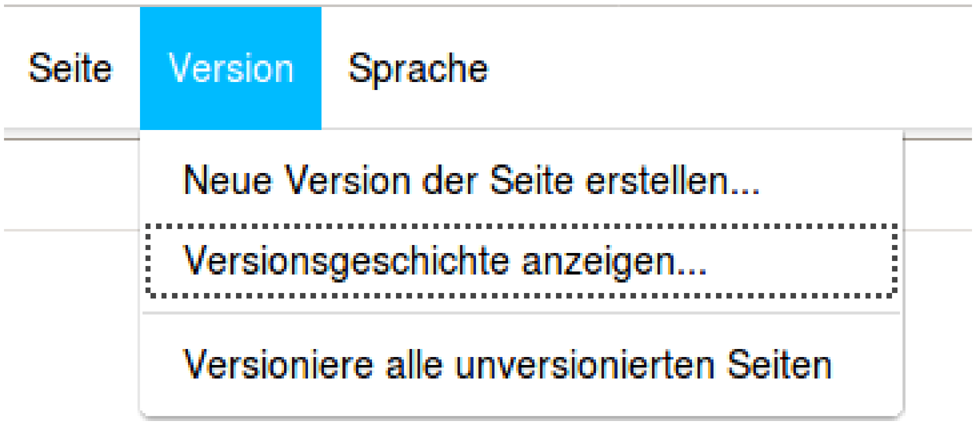

Seite
Version
Sprache
Neue Version der Seite erstellen...
Versionsgeschichte anzeigen...
Versioniere alle unversionierten Seiten

On creating a new page version the user can enter a title and description.


Seite
Version
Sprache
Neue Version der Seite erstellen...
Versionsgeschichte anzeigen...
Versioniere alle unversionierten Seiten


page version hinzufügen
NAME:
first snapshot
KOMMENTAR:
old website design + content
Abbrechen
Sichern und weiter bearbeiten
Sichern und neu hinzufügen
Sichern

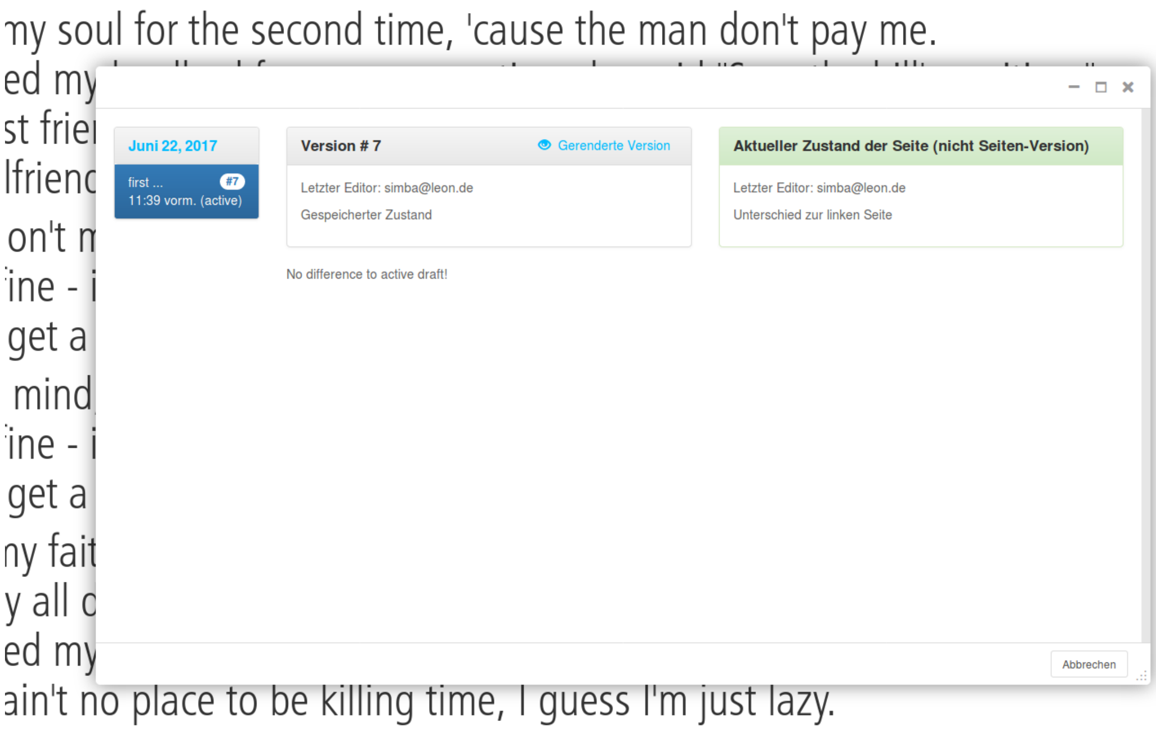
If a user wants to compare page versions, perform a rollback or edit snapshotted versions she or he has to use the 'History view'.


Seite
Version
Sprache
Neue Version der Seite erstellen...
Versionsgeschichte anzeigen...
Versioniere alle unversionierten Seiten

It is divided into three columns. The left sidebar is ordered by date and grouped by day. The collapsible with the current active version is expanded by default. The hover contains the page version description.

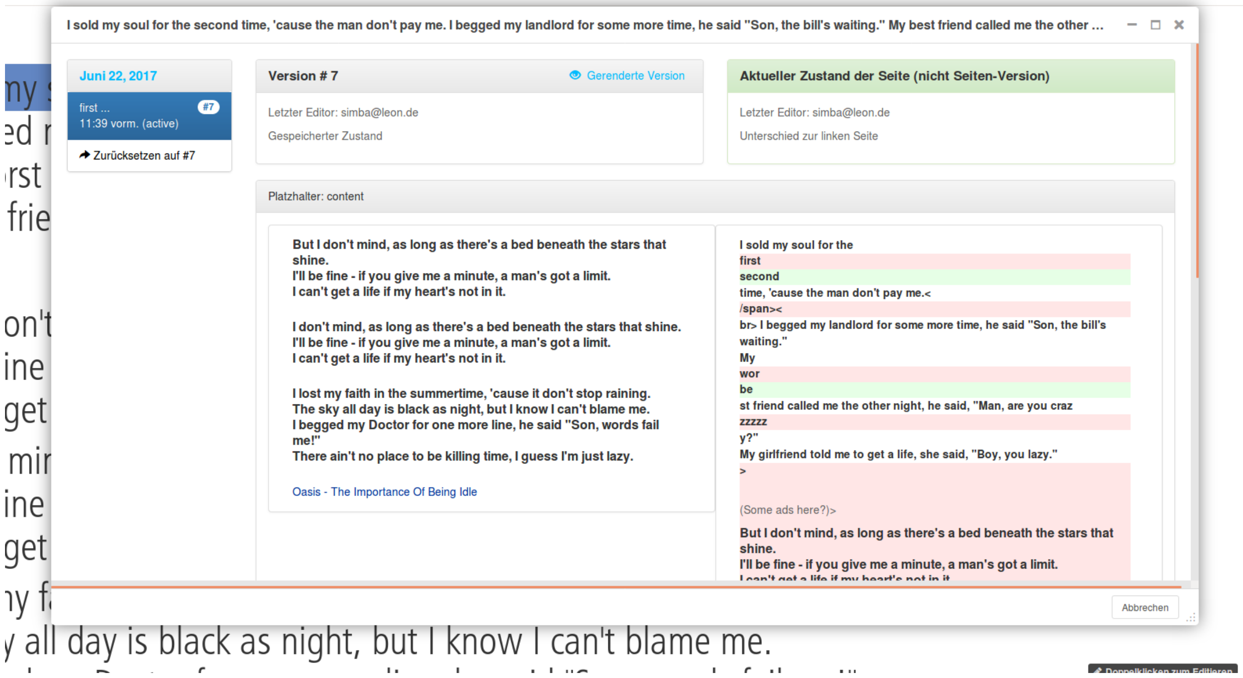
By clicking on an entry in the sidebar the panel in the middle will change.

This panel shows the page version selected in the left sidebar. The right panel shows the calculated changes to the left panel.


Juni 22, 2017
first ... #7
11:39 vorm. (active)
Version # 7
Gerenderte Version
Letzter Editor: simba@leon.de
Gespeicherter Zustand
Aktueller Zustand der Seite (nicht Seiten-Version)
Letzter Editor: simba@leon.de
Unterschied zur linken Seite
Abbrechen

No difference to active draft!

If you make changes to the page you can see the diff…


Juni 22, 2017
first ... #7
11:39 vorm. (active)
Zurücksetzen auf #7
Version # 7
Gerenderte Version
Letzter Editor: simba@leon.de
Gespeicherter Zustand
Aktueller Zustand der Seite (nicht Seiten-Version)
Letzter Editor: simba@leon.de
Unterschied zur linken Seite
Platzhalter: content
Abbrechen
Doppelklicken zum Editieren

I sold my soul for the second time, 'cause the man don't pay me. I begged my landlord for some more time, he said "Son, the bill's waiting." My best friend called me the other …

But I don't mind, as long as there's a bed beneath the stars that shine.
I'll be fine - if you give me a minute, a man's got a limit.
I can't get a life if my heart's not in it.

I don't mind, as long as there's a bed beneath the stars that shine.
I'll be fine - if you give me a minute, a man's got a limit.
I can't get a life if my heart's not in it.

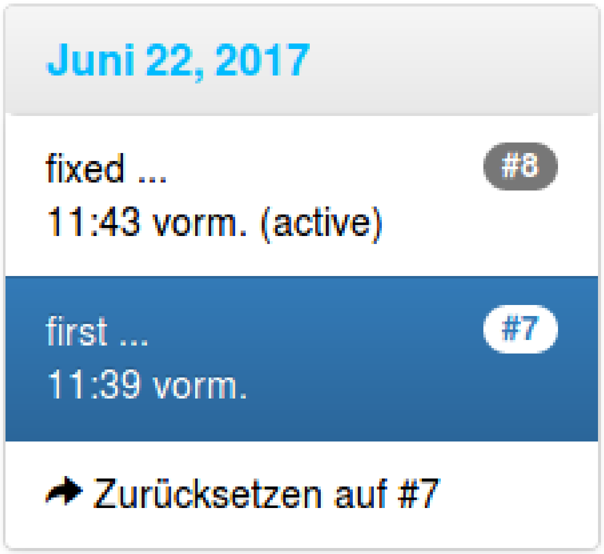
I lost my faith in the summertime, 'cause it don't stop raining.
The sky all day is black as night, but I know I can't blame me.
I begged my Doctor for one more line, he said "Son, words fail me!"
There ain't no place to be killing time, I guess I'm just lazy.

Oasis - The Importance Of Being Idle

I sold my soul for the
first
second
time, 'cause the man don't pay me.<
/span><
br> I begged my landlord for some more time, he said "Son, the bill's waiting."
My
wor
be
st friend called me the other night, he said, "Man, are you craz
zzzzz
y?"
My girlfriend told me to get a life, she said, "Boy, you lazy."
>

(Some ads here?)>
But I don't mind, as long as there's a bed beneath the stars that shine.
I'll be fine - if you give me a minute, a man's got a limit.

You can rollback in the left sidebar



1.3.3 Editing multiple version

Clicking on the ‘rendered version’ link brings you the rendered page version. You can edit this version. This enables teams to work on different versions of one page and in the end pick the better one or merge two page versions (utter has to be done manually).

1.4 Settings

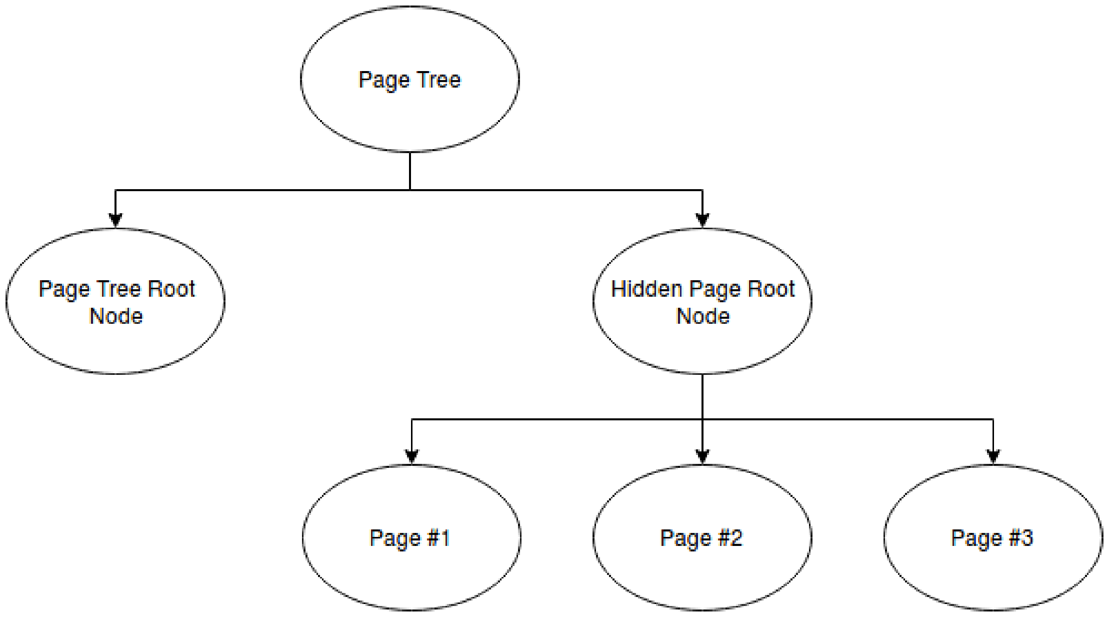
Setting	Default	Description
DJANGOCMS_REVERSION2_BIN_NAMING_PREFIX	.	the first chars of bin
DJANGOCMS_REVERSION2_BIN_NAME	Papierkorb	name of the bin
DJANGOCMS_REVERSION2_BIN_BUCKET_NAMING	Bucket-%d.%m.%Y	name of the bin
LANGUAGE_CODE		bin language

1.5 Trees

We split the page tree from the PageVersion tree because we might want to copy pages with children and separate our logic from the page logic.

1.5.1 PageTree

We add one "hidden" node to the page tree and add all page drafts flat under that node.

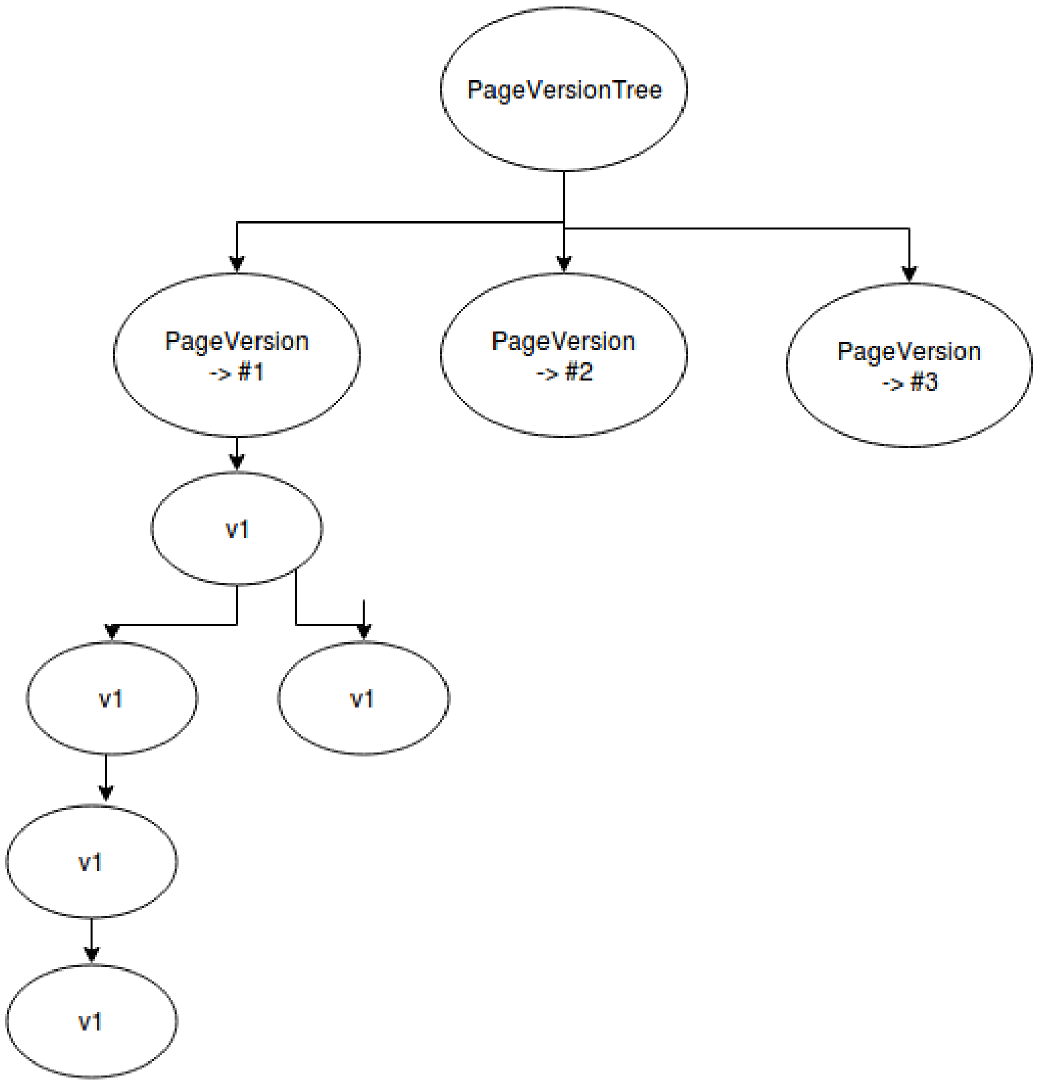


1.5.2 PageVersionTree

On the first layer there are all pages referenced. The version tree is aligned under the PageVersionNode.

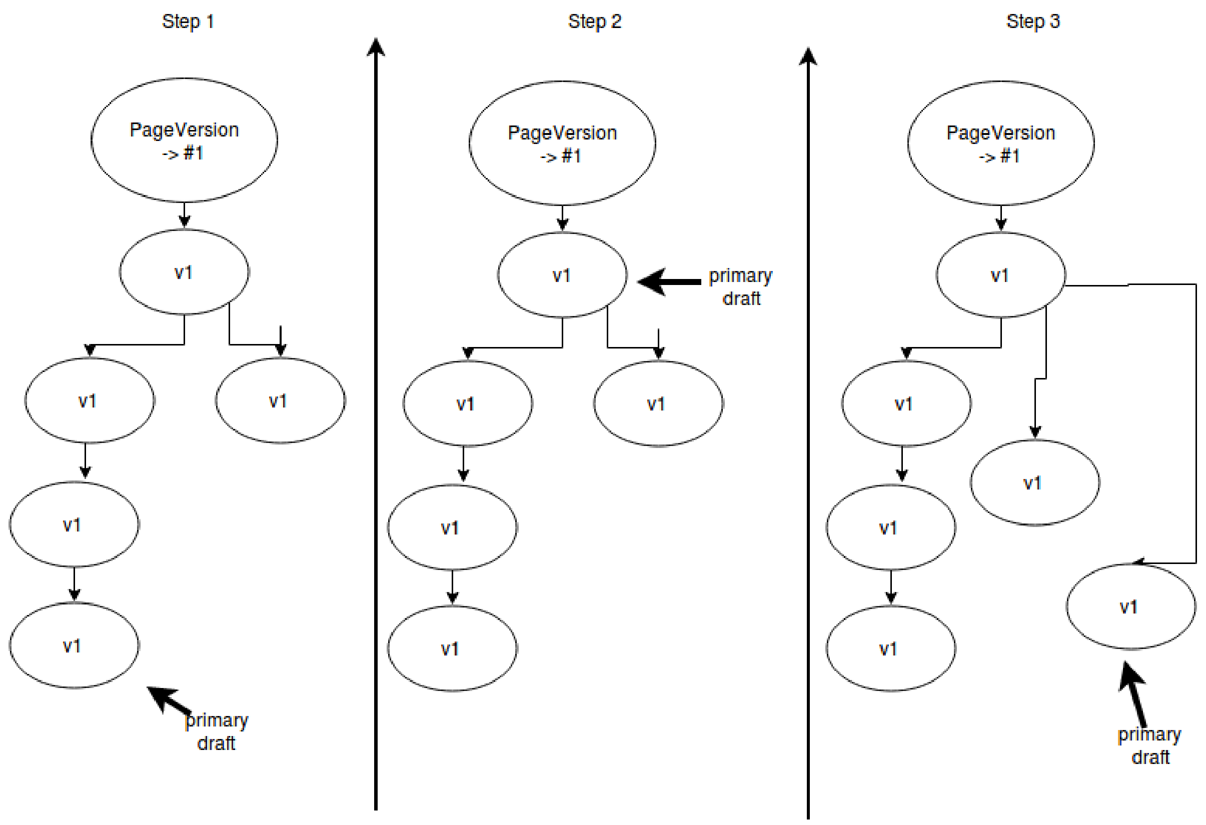
One node of the version subtree is always marked as primary. This node is synchronized with the CMS draft page.

If you want to roll back to an older version in the subtree. The `primary pointer` moves. If you edit the content of that node there has to be a branch.



1.5.3 Rollback

The following diagram show how the primary draft can be rolled back and how a new change to that node is implemented.



1.6 Permissions

Djangocms_reversion2 doesn't introduce new permissions. It makes use of the page permissions of djangocms.

Activate djangocms page permissions: **You have to add CMS_PERMISSIONS = True to your settings.py!**

The natural intuition is: A person that can edit a page can also edit PageVersions and so on.

1.6.1 Integration with djangocms-moderation

This plugin replaces the "publish button" with a "moderation request button". Then a "moderation workflow" is attached to the page. Now the edit functions of the page are blocked until the workflow is either rejected or approved.

How we could use this:

1. We still hide the toolbar in the "edit page version" mode but add another button that triggers a special workflow
2. When that workflow has finished
3. **Here comes the tricky part with djangocms-moderation -> they are building a workflow for approval of page publishing**

   -> we would like to use this in future releases. This might look like this:

   - the external starts a new moderation request on the edited hidden_page

- we use @receiver(post_obj_operation) signal handler to catch the end of the workflow
- instead of clicking the publish button on the hidden_page (we make a rollback of the connected draft)

1.7 Implementation of the current permission system

We make use of the cms page permissions because they are already there and we don't get any inconsistencies with custom permissions. Every page has a `pagepermission_set`. If a new page version is created all of these permissions have to be copied to the hidden_page of the page version. But there are also permissions that can be inherited from parent.

Therefore we use `utils.inherited_permissions(page)` to obtain all relevant inherited page permissions, then we have to transform them a little bit. I am talking about the following to cases where the permission is explicitly set to only affect the children or descendants.

We change the grant_on attribute of the copied permission in the left case to the right value:

- ACCESS_DESCENDANTS: ACCESS_PAGE_AND_DESCENDANTS,
- ACCESS_CHILDREN: ACCESS_PAGE

1.7.1 Caution

As a consequence, if you make changes to the permissions of your original page they will currently not be applied to the page versions. Example: The permissions of a page are removed for a person. That person could still access the PageVersions of the past.

1.7.2 View

If a user can view a page she or he can view its page versions.

1.7.3 Creation

A user can create a page version for pages for which he has the cms.page.can_change_page. An admin can create a page if he has cms.page_global_permission.can_change_page_global_permission

cms.utils.page_permissions: user_can_change_page

1.7.4 Batch-Creation

Only superusers are allowed to use the batch creation.

1.7.5 Edit

A user can edit page versions of pages that he is allowed to edit. TODO: see copy pagepermissions on copy (see https://github.com/Blueshoe/djangocms-reversion2/issues/32)

1.7.6 Deletion

Attention: A user mustn't delete own page versions because it might harm the page version tree. TODO: So far page versions cannot be deleted at all ;)

An admin can delete page versions if he has the cms.page_global_permission.can_delete_page_global_permission

1.7.7 Rollback

A user can roll back a page to a page version if he has the permission to publish the page (cms.page.can_publish_page) An admin can rollback a page if he has any global page permission (cms.page_global_permission.can_change_page_global_permission)

1.7.8 Example: How the check works

```
# check if the current user is allowed to revert the page by checking the publish␣
↪permission
user = get_current_user()
if not user_can_publish_page(user, page_version.draft):
    messages.error(request, _('Missing permission to publish this page which is␣
↪necessary for the rollback'))
    prev = request.META.get('HTTP_REFERER')
    if prev:
        return redirect(prev)
    else:
        raise Http404
```

1.8 Tests

- Create PageVersion for Page in English
- Create PageVersion for Page in German
- Try to create create a PageVersion although page is not dirty
- Make a change
- revert page
- batch add of 2 PageVersions

1.9 Todo

1.9.1 For version 1 (stable)

[ ] Write tests [ ] Run them with Travis [ ] close popup in view-version [ ] move page out of bin [ ] wheel

1.9.2 For version 2

[ ] Create a PageVersion before user makes 'Revert to live' (issue #29) [ ] Delete PageVersions (issue #20) [ ] Auto saves (#19) [ ] Backup: Create a tagged PageVersion of all sites (#18) [ ] Attributes for PageVersion: tag, was_published… (#17)

CHAPTER 2

Disclaimer

No software is perfect, everyone's code sucks. Feel free to suggest, criticize and/or contribute.

Proper Frontend/CMS-Admin Integration - Currently, we unregister django-cms' default PageAdmin and register our own PageAdmin. The overriding of PageAdmin appears necessary as it provides the only hook into Plugins being moved.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search