django-wiki Documentation

Release 0.8.2

Benjamin Bach

May 07, 2022

CONTENTS

INSTALLATION

1.1 Pre-requisite: Pillow

For image processing, django-wiki uses the Pillow library (a fork of PIL). The preferred method should be to get a system-wide, pre-compiled version of Pillow, for instance by getting the binaries from your Linux distribution repos.

1.1.1 Debian/Ubuntu

You need to get development libraries which Pip needs for compiling:

```
sudo apt-get install libjpeg8 libjpeg-dev libpng12-0 libpng12-dev
```

After that, install with `sudo pip install Pillow`. You might as well install Pillow system-wide, because there are little version-specific dependencies in Django applications when it comes to Pillow, and having multiple installations of the very same package is a bad practice in this case.

1.1.2 Mac OS X 10.5+

Ethan Tira-Thompson has created ports for OS X and made them available as a .dmg installer. Download and install the universal combo package here.

Once you have the packages installed, you can proceed to the pip installation. PIL will automatically pick up these libraries and compile them for django use.

1.2 Installing

To install the latest stable release:

```
pip install wiki
```

Install the latest pre-release (alpha, beta or rc):

```
pip install --pre wiki
```

1.3 Upgrading

Always read the *Release notes* for instructions on upgrading.

1.4 Configuration

1.4.1 Configure `settings.INSTALLED_APPS`

The following applications should be listed - NB! it's important to maintain the order due to database relational constraints:

```
'django.contrib.sites.apps.SitesConfig',
'django.contrib.humanize.apps.HumanizeConfig',
'django_nyt.apps.DjangoNytConfig',
'mptt',
'sekizai',
'sorl.thumbnail',
'wiki.apps.WikiConfig',
'wiki.plugins.attachments.apps.AttachmentsConfig',
'wiki.plugins.notifications.apps.NotificationsConfig',
'wiki.plugins.images.apps.ImagesConfig',
'wiki.plugins.macros.apps.MacrosConfig',
```

1.4.2 Configure `context_processors`

django-wiki uses the *Django Templates* backend. Add `'sekizai.context_processors.sekizai'` and `'django.template.context_processors.debug'` to `context_processors` section of your template backend settings. Please refer to the Django templates docs to see the current default setting for this variable.

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'APP_DIRS': True,
        # ...
        'OPTIONS': {
            'context_processors': [
                'django.contrib.auth.context_processors.auth',
                'django.template.context_processors.debug',
                'django.template.context_processors.i18n',
                'django.template.context_processors.media',
                'django.template.context_processors.request',
                'django.template.context_processors.static',
                'django.template.context_processors.tz',
                'django.contrib.messages.context_processors.messages',
                "sekizai.context_processors.sekizai",
            ],
        },
    },
]
```

1.4.3 Database

To sync and create tables, do:

```
python manage.py migrate
```

1.4.4 Set `SITE_ID`

If you're working with fresh Django installation, you need to set the SITE_ID

```
SITE_ID = 1
```

1.4.5 User account handling

There is a limited account handling included to allow users to sign up. Its settings are shown below with their default values. To switch off account handling entirely, set `WIKI_ACCOUNT_HANDLING = False`.

```
WIKI_ACCOUNT_HANDLING = True
WIKI_ACCOUNT_SIGNUP_ALLOWED = True
```

After a user is logged in, they will be redirected to the value of `LOGIN_REDIRECT_URL`, which you can configure in your project's settings.py to point to the root article:

```
from django.urls import reverse_lazy
LOGIN_REDIRECT_URL = reverse_lazy('wiki:get', kwargs={'path': ''})
```

1.4.6 Include urlpatterns

To integrate the wiki in your existing application, you should ensure the following lines are included in your project's urls.py.

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('notifications/', include('django_nyt.urls')),
    path('', include('wiki.urls'))
]
```

The above line puts the wiki in / so it's important to put it at the end of your urlconf. You can also put it in */wiki* by putting `'^wiki/'` as the pattern.

Note: If you are running `manage.py runserver`, you need to have static files and media files from `STATIC_ROOT` and `MEDIA_ROOT` served by the development server. `STATIC_ROOT` is automatically served, but you have to add `MEDIA_ROOT` manually:

```
urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Please refer to the Django docs.

TWO

RELEASE NOTES

2.1 Release plan

- **0.3.x** series suppors Django 1.11. As with the upstream Django release, 0.3 was be the last series with Python 2.7 support.

- **0.4.x** supports Django 1.11 and Django 2.1 and Python 3.4+.

- **0.5.x** Remove Django 1.11 support, adds Django 2.2 and 3.x support. Python 3.5+.

- **0.6.x** Targets Bootstrap v4, if you are interested in this work, please get in touch on Github!

- **0.7.x** Removes Django 2.1 support, adds Django 3.1, 3.2

2.2 0.8.2

Released on 2022-05-07

2.2.1 Changed

- Python-Markdown updated to 3.3 #1180 (Benjamin Balder Bach)

2.2.2 Fixed

- Support for revision history w/ IP Address from `HTTP_X_REAL_IP` #1184 (David van Rijn)

2.3 0.8.1

Released on 2022-04-25

2.3.1 Fixed

- Bootstrap pagination is updated to render as intended #1187 (Alexander Johan Arntzen)
- Toolchain updates: pre-commit, black, pytest, flake8, django-functests, bleach #1187 (Benjamin Balder Bach)

2.4 0.8

Released on 2022-02-14

2.4.1 Added

- Support for Django 4.0 and Python 3.10 #1165 (Mads Jensen)

2.4.2 Changed

- Bump dependency versions of sorl-thumbnail, django-nyt, django-mptt and django-functest #1165 (Benjamin Balder Bach)

2.4.3 Fixed

- Attachment search failing if files exceptionally missing on server #1162 (Benjamin Balder Bach)

2.5 0.7.10

Released on 2021-12-29

2.5.1 Fixed

- Missing validation for username uniqueness in in signup view #1152 (Benjamin Balder Bach)
- Uploading and unpacking .zip files as attachments was broken #1159 (kylecapricious2)

2.6 0.7.9

Released on 2021-11-15.

2.6.1 Security fixes

- XSS vulnerability: Unescaped HTML in title propagated to notification (WhiteSource Vulnerability Research Team)

2.7 0.7.8

Released on 2021-10-25.

2.7.1 Fixed

- Notification menu bug after Bootstrap upgrades #1142 (Fred Dyc)

2.8 0.7.7

Released on 2021-08-28.

2.8.1 Changed

- Bundled JQuery upgraded from 3.4.1 to 3.6.0 #1138 (Benjamin Balder Bach)

2.8.2 Fixed

- Small notification plugin registration bug introduced in 0.7.6 #1132 (SlyPerdix)

2.9 0.7.6

Released on 2021-08-01.

2.9.1 Fixed

- Notification menu bug after Bootstrap upgrades #1097 (SlyPerdix)
- Compatibility with future python-markdown, use `register` instead of `add` #1099 (Iqbal Abdullah)

2.10 0.7.5

Released on 2021-04-10.

2.10.1 Added

- Django 3.2 support #1121 (Benjamin Bach)

2.11 0.7.4

Released on 2021-04-10.

2.11.1 Fixed

- Settings page on article broken for non-superusers #1058 (Benjamin Balder Bach)
- Dependency `bleach` is bumped to >=`3.3.0`,<`3.4` after security advisory #1109 (Benjamin Balder Bach)
- Font-awesome icon on external links was not showing #1111 (Benjamin Balder Bach)
- Red links were not correctly displaying in some cases #1114 (Gereon Kaiping)

2.11.2 Translation updates

- Chinese: 100% translated, 100% reviewed
- Czech: 100% translated
- Dutch: 100% translated, 100% reviewed
- French: Some syntax fixes, remains 100% translated
- Hungarian: 100% translated
- Korean: 100% translated, 96% reviewed
- Russian: 100% translated
- Turkish: 100% translated

2.12 0.7.3

Released on 2021-01-11.

2.12.1 Fixed

- Plugin `editsection` failing with unexpected contents #1094 (Teury Diaz, Benjamin Balder Bach, OTR)

2.13 0.7.2

Released on 2021-01-08.

2.13.1 Fixed

- Use `.iter()` instead of long deprecated and removed method `.getiterator` [#1083](Teury Diaz, Benjamin Balder Bach)

2.14 0.7.1

Released on 2020-12-28.

2.14.1 Fixed

- Fixed cache key exceptions for `SafeString`` objects ``no attribute append` [#1072](Gert-Jan Braas)

2.15 0.7

Released on 2020-10-28.

2.15.1 Added

- Django 3.1 support [#1061](and [#1082](Mads Jensen, Benjamin Bach)

2.15.2 Fixed

- Do not fail prematurely during Django checks framework (rare issue) [#1059](Benjamin Bach)
- Cache keys failing in memcached if username contains space characters (rare) [#1065](Benjamin Bach)

2.15.3 Removed

- Django 2.1 support removed [#1061](Mads Jensen)

2.15.4 Translations

- Japanese (ja): 100% translated
- Brazilian Portuguese (pt_BR): 100% translated

2.16 0.6

Released on 2020-06-03.

> **Warning:** These release contains Bootstrap v4. If you have overridden django-wiki's templates but rely on the distributed Bootstrap CSS, then a lot of CSS class names have changed. Please refer to the Bootstrap Documentation.

2.16.1 Added

- Bootstrap 4 replaces Bootstrap 3: Improved default theming. #1035 (slinkymanbyday, Benjamin Bach)
- Django 3.0 support #1019 (Benjamin Bach, slinkymanbyday)
- New plugin `wiki.plugins.editsection` displays an `[edit]` link next to section headers (Frank Loemker) #652

2.16.2 Fixed

- Python 3.7 issue with notifications plugin main view `/_plugin/notifications/` #1000 (Mads Jensen)
- Broken Delete and Deleted pages #976 (Benjamin Bach)
- Can't delete article with `USE_THOUSAND_SEPARATOR = True` #1014 (tim3towers)
- Deleting images fails :url-issue:'936' (Gert-Jan Braas, Steckelfisch)

2.16.3 Changed

- Use SASS instead of LESS for Stylesheets. The compiler in `Makefile` is `lessc`. #1035 (Benjamin Bach)
- Removed `src/wiki/static/wiki/css/wiki-bootstrap.css` - Only distribute a minified CSS version. #1035 (Benjamin Bach)
- Test coverage upped from 75 to 80+% #976 (Mads Jensen, Benjamin Bach)
- PDF attachment Content-Disposition header changed to `inline` for browser previewing #1010 (nicolazilio)
- PyTest upgraded to latest 5.3
- django-mptt updated from 0.9 to 0.11.0 #1019 (Benjamin Bach, slinkymanbyday)
- sorl-thumbnail bumped to 12.6.2 #1019 (Benjamin Bach, slinkymanbyday)
- Upgrade bleach from 2.1 to 3.1 #1020 (slinkymanbyday)
- Python-Markdown 3.2 compatibility (Benjamin Bach)

2.16.4 Removed

- Python 3.4 support more or less definitively removed (no longer supported by test suite PyTest)
- Removed unmaintained plugin `wiki.plugins.haystack`

2.16.5 Translations

- Dutch translation 100% completed #1037 (Gert-Jan Braas)
- Polish 100% completed

2.17 0.5

Compatibility note, 2020-02-18: Django 3.0 support was scheduled but never completed, as dependencies lacked the support. It was completed in 0.6.

2.17.1 Changed

- Update to Markdown >= 3.1 #920 (Don Bowman, Benjamin Bach)
- Several code-cleanups, test improvements and test

2.17.2 Added

- Django 2.2

2.17.3 Removed

- Django 1.11 support

2.17.4 Translations

We need help to complete translations. It's done easily by creating a profile and joining the django-wiki project on Transifex. You are also encouraged to create new languages if you would like to translate to a language that doesn't yet exist.

2.18 0.4.5

2.18.1 Fixed

- Django admin error when uploading images: Column 'revision_number' cannot be null #950 (Benjamin Bach)

2.18.2 Translations

- Added: Romanian `ro`

2.19 0.4.4

2.19.1 Fixed

- Projects fail to load with custom `User` models without a `username` field #865 (trevorpeacock)
- Use `User.get_username()` for article cache instead of `User.__str__` #931 (Ole Anders Stokker)

2.20 0.4.3

Discarded release due to git errors (the actual fixes were not merged in).

- Automated language updates from Transifex

2.21 0.4.2

2.21.1 Fixed

- Using customized `MESSAGE_TAGS` setting caused `KEY_ERROR` #922 (Benjamin Bach)

2.22 0.4.1

2.22.1 Security

- jQuery upgrade from 1.12.4 to 3.3.1. jQuery UI also upgraded (for dynamic resizing of modals) #882 (Benjamin Bach)

2.23 0.4

2.23.1 Added

- Django 2.0 and 2.1 support #755 (Raffaele Salmaso & Mads Jensen)
- Python 3.7 support
- Added `wiki.sites.WikiSite` for easy customization #827
- Automatic link highlighting of URLs handles lots of new patterns #816 (Branko Majic)
- Red links: Internal links turn red and link to Create Page (Mathias Rav) #889

2.23.2 Translations

- Merged `pt` and `pt_PT`, then deleted `pt` and linked it to `pt_PT` #858
- Linked `zh_Hans` to `zh_CN` #711

Complete / almost complete:

- Chinese (China) `zh_CN` 100.00%
- Danish `da` 100.00%
- Dutch `nl` 94.32%
- French `fr` 97.95%
- German `de` 95.00%
- Korean (Korea) `ko_KR` 95.00%
- Polish (Poland) `pl_PL` 98.18%
- Portuguese (Brazil) `pt_BR` 95.00%
- Russian `ru` 99.55%
- Slovak `sk` 94.77%
- Spanish `es` 94.77%

Well under way, need support:

- Chinese (Taiwan) `zh_TW` 34.55%
- Czech `cs` 35.45%
- Finnish `fi` 81.14%
- Italian `it` 47.05%
- Japanese `ja` 79.77%
- Norwegian Bokmål (Norway) `nb_NO` 34.77%
- Portuguese (Portugal) `pt_PT` 79.32%
- Turkish (Turkey) `tr_TR` 30.68%

2.23.3 Changed

- Dependency for escaping HTML and safeguarding against injections `bleach` upgraded `>=2.1,<2.2` (last-partizan) #702
- Use full path names for `MARKDOWN_KWARGS['extensions']` as short names support wil be removed in `Markdown 2.7` #823
- Support for `include('wiki.urls')` for urls instantiation #827
- Use Django's 'checks.py' pattern to test configuration (Raffaele Salmaso & Mads Jensen) #830 #807
- Test coverage added: Images plugin + Account handling (Mads Jensen) #804
- Last couple of non-CBVs (Class Based Views) refactored to CBV (Raffaele Salmaso & Mads Jensen) #788 #819 #808
- Big cleanup: Deprecating lots of Python 2.7 specific code (Mads Jensen & Raffaele Salmaso) See: >30 PRs

- Search term highligting tweaked, first match is now highlighted instead of last (Mathias Rav) #901

- Markdown parsing for `[image]`, `[article_list]` and macros rewritten and improved to allow escaping (Mathias Rav) #896

2.23.4 Fixed

- Use `user.is_authenticated/is_anonymous` as a boolean #790 (Raffaele Salmaso)

- Use `simple_tag` for assignment tag #791 (Raffaele Salmaso)

- Direct invocation of `pytest` fixed (removing `runtests.py`) #781 (Branko Majic)

- Line breaks in help texts for macros #851 (Mathias Dannesbo)

- Table of contents now has a header by default, and several built-in django-wiki extensions can be configured using `WIKI_MARKDOWN_KWARGS` #881 (Mathias Rav)

- S3 Storage engine image deletion bug #907 (Andrea Maschio & Benjamin Bach)

- Back link on "permission denied" page should point to parent article on read errors #915 (Benjamin Bach & Christian Duvholt)

2.23.5 Deprecated/Removed

- Django < 1.11 support is dropped #779

- Python < 3.3 support is dropped #779 and #792

- Deprecate `wiki.urls.get_pattern` and `URL_CONFIG_CLASS` setting #799

- Removed `SEARCH_VIEW` setting, replaced by `WikiSite` override #837

2.24 0.3.1

- Fix error messages of missing migrations due to inconsistent change of `on_delete` on some model fields #776

2.25 0.3

2.25.1 Translation updates from Transifex

- Languages that need support:

  - Dutch 88%

  - Finnish 85%

  - Japanese 80%

  - Chinese (Taiwan) 36%

  - Norwegian 36%

  - Turkish 30%

  - Czech 13%

  - Italian 8%

- >90% completed: Chinese, French, German, Russian, Spanish, Danish, Korean, Polish, Portuguese (Brazilian), Slovak, Spanish

2.25.2 Added

- Search choice between either current or global tree (Christian Duvholt) #580 #731

- New bootstrapped image insert dialog (Frank Loemker) #628

- Allow the HTML tag `<hr>` (Frank Loemker) #629

- Global History overview of page revisions (Frank Loemker and Maximilien Cuony) #627

- Move article support with redirects (Frank Loemker) #640

- Django 1.11 compatibility (Luke Plant) #634

- Crop paginator window when there are >9 pages in a list (Frank Loemker) #646

- Extended syntax for attachment macro: `[attachment:id title:"text" size]` (Frank Loemker) #678

- Add Sphinx documentation for plugin settings (Frank Loemker) #681

- Show "log out" in menu when account handling is disabled (jenda1) #691

- Markdown tag with wiki paths now support fragments like `[Click Here](wiki:/path#header)` (Frank Loemker) #701

2.25.3 Changed

- Test refactor: Use django-functest and separate WebTest from Selenium (Luke Plant) #634

- Repo refactor: Moved `wiki` package to `src/` folder and test code to `tests/` #631

- Render django.contrib.messages with template tag and inclusion template: Configurable and bootstrap 3 compatible (Benjamin Bach and Frank Loemker) #654

- Don't hardcode redirect url in account update view (Benjamin Bach) #650

- Python 3.6 support added to test matrix (Benjamin Bach) #664

- Keep CSS global namespace clean, refactor CSS rule `label` -> `.wiki-label label` (Christian Duvholt) #679

- Plugins can whitelist HTML tags and attributes (jenda1) #700

- Optimizations to fundamental permission lookup managers (Christian Duvholt) #714

- Code quality upgrade, remove obsolete code, linting and tidying up (Mads Jensen) #797, #705, #707, #716, #717, #718, #719, #720, #721, #722, #724, #725, #726, #727, #728, #730, #732, #733, #735, #736, #737, #738, #741, #743, #743, #756, #757

- Added `AppConfig` class for all plugins (Raffaele Salmaso) #758

- Explicit `on_delete` for all `ForeignKey` fields (Raffaele Salmaso) #759

- Django 2.0 preparation: `atomic=False` for 3 migrations that rename tables/fields (Raffaele Salmaso) #760

- Set dependency `django-nyt<1.1` to avoid future breakage (Benjamin Bach) #761

2.25.4 Fixed

- Removed exception catch all in `URLPath.delete_subtree` which silenced errors while delete articles with descendents
- Fix article settings page in Django 1.11 (Frank Loemker) #682
- Fix upstream MPTT breaking deletion of articles from django-admin (Frank Loemker) #683
- Wrong HTML attribute 'type' on search result page (Geoff Clapp) #698
- Fix restoring of attachments and other RevisionPlugin types after deletion (Frank Loemker) #672
- Allowing <sup> because of footnotes (Frank Loemker) #750
- Hunted down unclosed HTML tags #750 (Mads Jensen) #741

2.26 0.2.5

- Set dependency `django-nyt<1.1` to avoid future breakage (Benjamin Bach) #761

2.27 0.2.4

- Hot-fix because of missing woff2 files #625

2.28 0.2.3

- Pulled Transifex translations and pushed source translations.
- Fix support for Py2 unicode in code blocks (Benjamin Bach) #607
- Support for Github style fenced codeblocks (Benjamin Bach) #618
- Cached articles showing up in wrong language (Benjamin Bach) #592
- Upgraded Bootstrap from 3.3.1 to 3.3.7 (Benjamin Bach) #620
- Upgraded bundled jQuery to 1.12.4 (Benjamin Bach) #620
- Setting `WIKI_MARKDOWN_HTML_STYLES` for allowing `style='..'` in user code (Benjamin Bach) #603
- Strip Markdown code in search result snippets (Benjamin Bach) #42

2.29 0.2.2

- Remove `wiki.decorators.json_view`, fixes server errors when resolving 404 links #604
- Replace usage of `render_to_response()` with `render()` #606
- Fix memory leak #609 and #611 (obtroston)
- Scroll bars and display area fixed for code blocks #601 and #608 (Branko Majic)
- Option `WIKI_MARKDOWN_SANITIZE_HTML` skips Bleach (warning: Don't use for untrusted code) #610 (Michal Hozza)

- Allow the HTML tag `<br>`. #613 (Frank Loemker)
- Add thumbnail size directive (example: `[image:123 size:large]`). #612 (Frank Loemker and @inflrscns)
- Fix error with absolute paths in wiki links (example: `[Sub-root](wiki:/sub-root)`) #616 (Benoit C. Sirois)
- Require `Django<1.11` #616 (Benoit C. Sirois)

2.30 0.2.1

- Lowercase slugs when creating new pages with `[[Like This]]` #595 (Eric Clack)
- Fix issues related to Bleach before Markdown processing esp. pertaining > characters. #596
- Remove `wiki.plugins.mediawikiimport` #597
- Pretty up the highligted code's line enumeration #598
- Customize codehilite in order to wrap highlighted code with scrollbars #598

2.31 0.2

- Translation updates from Transifex
    - Danish translation from 39% to 100% (Bo Holm-Rasmussen)
    - Updated languages since 0.1: Chinese, French, German, German, Russian, Spanish
- Added Django 1.10 support #563
- Security: Do not depend on markdown `safe_mode`, instead use `bleach`.
- Fix duplicate search results when logged in #582 (duvholt)
- Do not allow slugs only consisting of numbers #558
- Copy in urlify.js and fix auto-population of slug field in Django 1.9+ #554
- Fix memory leak in markdown extensions setting #564
- Updated translations - Languages > 90% completed: Chinese (China), Portuguese (Brazil), Korean (Korea), French, Slovak, Spanish, Dutch, German, Russian, Finnish.
- Taiwanese Chinese added (39% completed)
- Cleanup documentation structure #575

2.31.1 HTML contents

Bleach is now used to sanitize HTML before invoking Markdown.

HTML escaping is done before Markdown parsing happens. In future Markdown versions, HTML escaping is no longer done, and `safe_mode` is removed. We have already removed `safe_mode` from the default `WIKI_MARKDOWN_KWARGS` setting, however if you have configured this yourself, you are advised to remove `safe_mode`.

Allowed tags are from Bleach's default settings: `a`, `abbr`, `acronym`, `b`, `blockquote`, `code`, `em`, `i`, `li`, `ol`, `strong`, `ul`.

Please use new setting `WIKI_MARKDOWN_HTML_WHITELIST` and set a list of allowed tags to customize behavior.

2.31.2 Python and Django support

Support has been removed for:

- Python 2.6
- Django < 1.8
- South

2.32 0.1.2

- Remove unwanted items from default menu when `WIKI_ACCOUNT_HANDLING = False`. #545
- Fix broken soft-deletion and restoring of images, and "set revision" functionality #533
- Added responsiveness to tables by use of Bootstrap table-responsive class #552

2.33 0.1.1

- Several languages updated from Transifex
  - Slovak added **Thanks M Hozza**
  - Portuguese also added, but as copy of PT-BR (make changes as desired in Transifex)
- Brand new Account Settings page (email / password) **Thanks inflrscns**
- Testproject turned into Django 1.9 layout
- Replace context-processor dependent use of `{{ STATIC_URL }}` with `{% static %}`
- Bugfix for `pip install wiki` in an empty (no Django installed) virtualenv
- Precommit hooks added in repository
- Import statements sorted and codebase re-pep8'thed
- Log in page is now called "Log in" in `<title>` tag - **Thanks Eugene Obukhov**

2.34 0.1

> **Warning:** If you are upgrading from a previous release, please ensure that you pass through the 0.0.24 release because it contains the final migrations necessary before entering the django-wiki 0.1+ migration tree.
>
> If you are using django 1.7+ and have an old installation of django-wiki (which should be impossible since it wouldn't run) please downgrade to 1.6 as follows:
>
> ```
> $ pip install wiki\<0.1 --upgrade  # Latest 0.0.24 release
> $ pip install django\<1.7  # Downgrade django if necessary
> $ python manage.py migrate  # Run 0.0.24 migrations
> $ pip install wiki\<0.2 --upgrade  # Upgrade to latest 0.1 series
> $ python manage.py migrate --delete-ghost-migrations  # Run migrations again,
>                                                        # removing the (ghost)
>                                                        # migrations from previous
>                                                        # release
> $ # Feel free to upgrade Django again
> ```

```

```

Supported

- Python 2.7, 3.3, 3.4, 3.5 (3.2 is not supported)

- Django 1.5, 1.6, 1.7, 1.8, 1.9

- Django < 1.7 still needs South, and migration trees are kept until next major release.

2.34.1 Breaking changes

wiki.VERSION as tuple

We want to follow Django's way of enumerating versions. If you want the old string version, use `wiki.__version__`.

Plugin API

Since Django 1.8 has started making warnings about *patterns* being deprecated, we've decided to stop using them by default. Thus, as with the future Django 2.0, we will use lists of *url* objects to store the urlconf of plugins. All the bundled plugins have been updated to reflect the change.

Django-mptt

We now depend on django-mptt 0.7.2+ for Django 1.8 compatibility.

2.35 0.0.24

This release is a transitional release for anyone still using an older version of django-wiki. The code base has been heavily refactored and this is hopefully the final release.

> **Warning:** 0.0.24 is mainly a transitional release, but new features and bug fixes are included, too.

Compatibility

- Django 1.5, 1.6 (That means Django 1.7 is **not** yet fully supported)

- South 1.0+ (if you are on an older South, you **need** to upgrade)

- Python 2.6, 2.7, 3.3, 3.4

2.35.1 Upgrading

Firstly, upgrade django-wiki through familiar steps with pip

```
$ pip install wiki --upgrade
```

During the upgrade, notice that django-nyt is installed. This replaces the previously bundled django_notify and you need to make a few changes in your settings and urls.

In `settings.INSTALLED_APPS`, replace *"django_notify"* with *"django_nyt"*. Then open up your project's urlconf and make sure you have something that looks like the following:

```
from wiki.urls import get_pattern as get_wiki_pattern
from django_nyt.urls import get_pattern as get_nyt_pattern
urlpatterns += patterns('',
    (r'^notifications/', get_nyt_pattern()),
    (r'', get_wiki_pattern())
)
```

Notice that we are importing *from django_nyt.urls* and no longer *django_notify* and that the function is renamed to *get_nyt_pattern*.

After making these changes, you should run migrations.

```
$ python manage.py migrate
```

Notifications fixed

In past history, django-wiki has shipped with a very weird migration. It caused for the notifications plugin's table of article subscriptions to be removed. This is fixed in the new migrations and the table should be safely restored in case it was missing.

However, you may want to bootstrap subscription notifications in case you have run into this failed migration. You can ensure that all owners and editors of articles receive notifications using the following management command:

python manage.py wiki_notifications_create_defaults

2.35.2 Troubleshooting

If you have been running from the git master branch, you may experience problems and need to re-run the migrations entirely.

```
python manage.py migrate notifications zero --delete-ghost-migrations
python manage.py migrate notifications
```

If you get *DatabaseError: no such table: notifications_articlesubscription*, you have been running django-wiki version with differently named tables. Don't worry, just fake the backwards migration:

```
python manage.py migrate notifications zero --fake
```

If you get `relation "notifications_articlesubscription" already exists` you may need to do a manual `DROP TABLE notifications_articlesubscription;` using your DB shell (after backing up this data).

After this, you can recreate your notifications with the former section's instructions.

2.36 News archive

2.36.1 April 15, 2017

0.2.3 released: Release notes

0.2.2 released: Release notes

2.36.2 February 27, 2017

0.2.1 released: Release notes

2.36.3 December 27, 2016

0.2 final released: Release notes

2.36.4 June 19, 2016

0.1.2 released: Release notes

2.36.5 May 6, 2016

0.1.1 released: Release notes

2.36.6 January 25, 2016

0.1 final released

2.36.7 December 26th, 2015

A new release 0.0.24.4 is out and has fixes for the Django `loaddata` management command such that you can create dumps and restore the dump. Notice, though, that `loaddata` only works for Django 1.7+.

Django 1.9 support is available in the current master, please help get a 0.1 released by giving feed back in the last remaining issues:

https://github.com/django-wiki/django-wiki/milestones/0.1

2.36.8 November 16th, 2015

Django 1.8 support is very ready and 0.1 is right on the doorstep now.

2.36.9 January 26th, 2015

After too long, the new release is out.

The wait was mainly due to all the confusing changes by adding support of Python 3 and readying the migrations for Django 1.7. But there's actually new features, too.

- Bootstrap 3.3.1 and Font Awesome 4 (Christian Duvholt)
- `django_nyt` instead of builtin `django_notify` (Benjamin Bach, Maximilien Cuony)
- `tox` for testing (Luke Plant)
- Appropriate use of gettext_lazy (Jaakko Luttinen)
- Fixed support of custom username fields (Jan De Bleser)
- Several fixes to the attachment plugin (Christian Duvholt)

- Errors on notifications settings tab (Benjamin Richter)
- Chinese translations (Ronald Bai)
- Finish translations (Jaakko Luttinen)
- Compatibility with custom user model in article settings (Andy Fang)
- Fixed bug when `[attachment:XX]` present multiple times on same line (Maximilien Cuony)
- Simple mediawiki import management command (Maximilien Cuony)
- Python 3 and Django 1.6 compatibility (Russell-Jones, Antonin Lenfant, Luke Plant, Lubimov Igor, Benjamin Bach)
- (and more, forgiveness asked if anyone feels left out)

PLUGINS

Add/remove the following to your `settings.INSTALLED_APPS` to enable/disable the core plugins:

- `'wiki.plugins.attachments.apps.AttachmentsConfig'`
- `'wiki.plugins.editsection.apps.EditSectionConfig'`
- `'wiki.plugins.globalhistory.apps.GlobalHistoryConfig'`
- `'wiki.plugins.help.apps.HelpConfig'`
- `'wiki.plugins.images.apps.ImagesConfig'`
- `'wiki.plugins.links.apps.LinksConfig'`
- `'wiki.plugins.macros.apps.MacrosConfig'`
- `'wiki.plugins.notifications.apps.NotificationsConfig'`

The notifications plugin is mandatory for an out-of-the-box installation. You can safely remove it from `INSTALLED_APPS` if you also override the **wiki/base.html** template.

CUSTOMIZATION

See *Settings* for the settings that can be used to configure django-wiki. Other ways to customize django-wiki for your use are listed below.

4.1 Templates

django-wiki can be customized by providing your own templates.

All templates used by django-wiki inherit from `wiki/base.html`, which in turn simply inherits from `wiki/base_site.html` (adding nothing). `wiki/base_site.html` provides a complete HTML page, but provides a number of blocks that you might want to override. The most useful are:

- `wiki_site_title`
- `wiki_header_branding`
- `wiki_header_navlinks`

These can be overridden to provide your own branding and links in the top bar of the page, as well as in browser window title. The `wiki/base_site.html` template uses Bootstrap 3, so the following example shows how to use this in practice, assuming you want a single link to your home page, and one to the wiki. Add the following as `wiki/base.html` somewhere in your `TEMPLATE_DIRS`:

```
{% extends "wiki/base_site.html" %}

{% block wiki_site_title %} - Wiki{% endblock %}

{% block wiki_header_branding %}
<a class="navbar-brand" href="/">Your brand</a>
{% endblock %}

{% block wiki_header_navlinks %}
<ul class="nav navbar-nav">
  <li class="active"><a href="{% url 'wiki:root' %}">Wiki</a></li>
</ul>
{% endblock %}
```

4.2 Site

You can override default django-wiki `wiki.sites.site` urls/views site implementation with your own: override by setting the `default_site` attribute of a custom `AppConfig` to the dotted import path of either a `WikiSite` subclass or a callable that returns a site instance.

```python
# myproject/sites.py

from wiki.sites import WikiSite

class MyWikiSite(admin.WikiSite):
    ...
```

```python
# myproject/apps.py

from wiki.apps import WikiConfig

class MyWikiConfig(WikiConfig):
    default_site = 'myproject.sites.MyWikiSite'
```

```python
# myproject/settings.py

INSTALLED_APPS = [
    ...
    'myproject.apps.MyWikiConfig',  # replaces 'wiki.apps.WikiConfig'
    ...
]
```

FIVE

SETTINGS

The following settings are available for configuration through your project. All settings are customized by prefixing `WIKI_`, so for instance `URL_CASE_SENSITIVE` should be configured as `WIKI_URL_CASE_SENSITIVE`. For plugins the prefix is `WIKI_PLUGINNAME_`, e.g. `WIKI_IMAGES` for the images plugin.

`wiki.conf.settings.`**`ACCOUNT_HANDLING = True`**

> Sign up, login and logout views should be accessible.

`wiki.conf.settings.`**`ACCOUNT_SIGNUP_ALLOWED = True`**

> Signup allowed? If it's not allowed, logged in superusers can still access the signup page to create new users.

`wiki.conf.settings.`**`ANONYMOUS = True`**

> Treat anonymous (i.e. non logged in) users as the "other" user group.

`wiki.conf.settings.`**`ANONYMOUS_CREATE = False`**

> Globally enable create access for anonymous users. Defaults to `ANONYMOUS_WRITE`.

`wiki.conf.settings.`**`ANONYMOUS_UPLOAD = False`**

> Default setting to allow anonymous users upload access. Used in plugins.attachments and plugins.images, and can be overwritten in these plugins.

`wiki.conf.settings.`**`ANONYMOUS_WRITE = True`**

> Globally enable write access for anonymous users, if true anonymous users will be treated as the others_write boolean field on models.Article.

`wiki.conf.settings.`**`CACHE_TIMEOUT = 600`**

> Seconds of timeout before renewing the article cache. Articles are automatically renewed whenever an edit occurs but article content may be generated from other objects that are changed.

`wiki.conf.settings.`**`CAN_ADMIN = None`**

> A function returning True/False if a user has permission to create new groups and users for the wiki.

`wiki.conf.settings.`**`CAN_ASSIGN = None`**

> A function returning True/False if a user has permission to assign permissions on an article. Relevance: Changing owner and group membership.

`wiki.conf.settings.`**`CAN_ASSIGN_OWNER = None`**

> A function returning True/False if the owner of an article has permission to change the group to a user's own groups. Relevance: Changing group membership.

`wiki.conf.settings.`**`CAN_CHANGE_PERMISSIONS = None`**

> A function returning True/False if a user has permission to change read/write access for groups and others.

`wiki.conf.settings.`**`CAN_DELETE = None`**

> Specifies if a user has access to soft deletion of articles.

`wiki.conf.settings.`**`CAN_MODERATE = None`**

> A function returning True/False if a user has permission to change moderate, ie. lock articles and permanently delete content.

`wiki.conf.settings.`**`CAN_READ = None`**

> A function returning True/False if a user has permission to read contents of an article and plugins. Relevance: Viewing articles and plugins.

`wiki.conf.settings.`**`CAN_WRITE = None`**

> A function returning True/False if a user has permission to change contents, i.e. add new revisions to an article. Often, plugins also use this. Relevance: Editing articles, changing revisions, editing plugins.

`wiki.conf.settings.`**`CHECK_SLUG_URL_AVAILABLE = True`**

> When True, this blocks new slugs that resolve to non-wiki views, stopping users creating articles that conflict with overlapping URLs from other apps.

`wiki.conf.settings.`**`EDITOR = 'wiki.editors.markitup.MarkItUp'`**

> The editor class to use – maybe a 3rd party or your own…? You can always extend the built-in editor and customize it!

`wiki.conf.settings.`**`GROUP_MODEL = 'auth.Group'`**

> Choose the Group model to use for permission handling. Defaults to django's auth.Group.

`wiki.conf.settings.`**`LOG_IPS_ANONYMOUS = True`**

> Do we want to log IPs of anonymous users?

`wiki.conf.settings.`**`LOG_IPS_USERS = False`**

> Do we want to log IPs of logged in users?

`wiki.conf.settings.`**`LOST_AND_FOUND_SLUG = 'lost-and-found'`**

> This slug is used in URLPath if an article has been deleted. The children of the URLPath of that article are moved to lost and found. They keep their permissions and all their content.

`wiki.conf.settings.`**`MARKDOWN_HTML_ATTRIBUTES = {'a': ['href', 'title', 'class', 'id', 'target', 'rel'], 'abbr': ['title', 'class', 'id', 'target', 'rel'], 'acronym': ['title', 'class', 'id', 'target', 'rel'], 'b': ['class', 'id', 'target', 'rel'], 'blockquote': ['class', 'id', 'target', 'rel'], 'br': ['class', 'id', 'target', 'rel'], 'code': ['class', 'id', 'target', 'rel'], 'dd': ['class', 'id', 'target', 'rel'], 'div': ['class', 'id', 'target', 'rel'], 'dl': ['class', 'id', 'target', 'rel'], 'dt': ['class', 'id', 'target', 'rel'], 'em': ['class', 'id', 'target', 'rel'], 'figcaption': ['class', 'id', 'target', 'rel'], 'figure': ['class', 'id', 'target', 'rel'], 'h1': ['class', 'id', 'target', 'rel'], 'h2': ['class', 'id', 'target', 'rel'], 'h3': ['class', 'id', 'target', 'rel'], 'h4': ['class', 'id', 'target', 'rel'], 'h5': ['class', 'id', 'target', 'rel'], 'h6': ['class', 'id', 'target', 'rel'], 'hr': ['class', 'id', 'target', 'rel'], 'i': ['class', 'id', 'target', 'rel'], 'img': ['class', 'id', 'target', 'rel', 'src', 'alt'], 'li': ['class', 'id', 'target', 'rel'], 'ol': ['class', 'id', 'target', 'rel'], 'p': ['class', 'id', 'target', 'rel'], 'pre': ['class', 'id', 'target', 'rel'], 'span': ['class', 'id', 'target', 'rel'], 'strong': ['class', 'id', 'target', 'rel'], 'sup': ['class', 'id', 'target', 'rel'], 'table': ['class', 'id', 'target', 'rel'], 'tbody': ['class', 'id', 'target', 'rel'], 'td': ['class', 'id', 'target', 'rel'], 'th': ['class', 'id', 'target', 'rel'], 'thead': ['class', 'id', 'target', 'rel'], 'tr': ['class', 'id', 'target', 'rel'], 'ul': ['class', 'id', 'target', 'rel']}`**

> Dictionary of allowed attributes in Markdown article contents.

`wiki.conf.settings.`**`MARKDOWN_HTML_STYLES = []`**

> Allowed inline styles in Markdown article contents, default is no styles (empty list).

`wiki.conf.settings.`**`MARKDOWN_HTML_WHITELIST = ['a', 'abbr', 'acronym', 'b', 'blockquote',`**
`'code', 'em', 'i', 'li', 'ol', 'strong', 'ul', 'figure', 'figcaption', 'br', 'hr', 'p',`
`'div', 'img', 'pre', 'span', 'sup', 'table', 'thead', 'tbody', 'th', 'tr', 'td', 'dl',`
`'dt', 'dd', 'h1', 'h2', 'h3', 'h4', 'h5', 'h6']`

> List of allowed tags in Markdown article contents.

`wiki.conf.settings.`**`MARKDOWN_KWARGS = {'extension_configs':`**
`{'wiki.plugins.macros.mdx.toc': {'title': 'Contents'}}, 'extensions':`
`['markdown.extensions.footnotes', 'markdown.extensions.attr_list',`
`'markdown.extensions.footnotes', 'markdown.extensions.attr_list',`
`'markdown.extensions.def_list', 'markdown.extensions.tables', 'markdown.extensions.abbr',`
`'markdown.extensions.sane_lists']}`

> Arguments for the Markdown instance, as a dictionary. The "extensions" key should be a list of extra extensions to use besides the built-in django-wiki extensions, and the "extension_configs" should be a dictionary, specifying the keyword-arguments to pass to each extension.
>
> For a list of extensions officially supported by Python-Markdown, see: https://python-markdown.github.io/extensions/
>
> To set a custom title for table of contents, specify the following in your Django project settings:

```
WIKI_MARKDOWN_KWARGS = {
    'extension_configs': {
        'wiki.plugins.macros.mdx.toc': {'title': 'Contents of this article'},
    },
}
```

> Besides the extensions enabled by the "extensions" key, the following built-in django-wiki extensions can be configured with "extension_configs": "wiki.core.markdown.mdx.codehilite", "wiki.core.markdown.mdx.previewlinks", "wiki.core.markdown.mdx.responsivetable", "wiki.plugins.macros.mdx.macro", "wiki.plugins.macros.mdx.toc", "wiki.plugins.macros.mdx.wikilinks".

`wiki.conf.settings.`**`MARKDOWN_SANITIZE_HTML = True`**

> Whether to use Bleach or not. It's not recommended to turn this off unless you know what you're doing and you don't want to use the other options.

`wiki.conf.settings.`**`MESSAGE_TAG_CSS_CLASS = {10: 'alert alert-info', 20: 'alert`**
`alert-info', 25: 'alert alert-success', 30: 'alert alert-warning', 40: 'alert`
`alert-danger'}`

> Mapping from message.level to bootstrap class names.

`wiki.conf.settings.`**`REVISIONS_MINUTES_LOOKBACK = 2`**

> Number of minutes to look back for looking up `REVISIONS_PER_MINUTES` and `REVISIONS_PER_MINUTES_ANONYMOUS`.

`wiki.conf.settings.`**`REVISIONS_PER_HOUR = 60`**

> Maximum allowed revisions per hour for any given user or IP.

`wiki.conf.settings.`**`REVISIONS_PER_HOUR_ANONYMOUS = 10`**

> Maximum allowed revisions per hour for any anonymous user and any IP.

`wiki.conf.settings.`**`REVISIONS_PER_MINUTES = 5`**

> Maximum allowed revisions per minute for any given user or IP.

`wiki.conf.settings.`**`REVISIONS_PER_MINUTES_ANONYMOUS = 2`**

Maximum allowed revisions per minute for any anonymous user and any IP.

`wiki.conf.settings.`**`SHOW_MAX_CHILDREN = 20`**

Maximum amount of children to display in a menu before showing "+more". NEVER set this to 0 as it will wrongly inform the user that there are no children and for instance that an article can be safely deleted.

`wiki.conf.settings.`**`STORAGE_BACKEND = <django.core.files.storage.DefaultStorage object>`**

Default Django storage backend to use for images, attachments etc.

`wiki.conf.settings.`**`URL_CASE_SENSITIVE = False`**

Should urls be case sensitive?

`wiki.conf.settings.`**`URL_CONFIG_CLASS = None`**

Dotted name of the class used to construct urlpatterns for the wiki. Default is wiki.urls.WikiURLPatterns. To customize urls or view handlers, you can derive from this.

`wiki.conf.settings.`**`USE_BOOTSTRAP_SELECT_WIDGET = True`**

User Bootstrap's select widget. Switch off if you're not using Bootstrap!

`wiki.conf.settings.`**`USE_SENDFILE = False`**

Use django-sendfile for sending out files? Otherwise the whole file is first read into memory and than send with a mime type based on the file.

`wiki.conf.settings.`**`WIKILINKS_TRAILING_SLASH = True`**

Weather to append a trailing slash to rendered Wikilinks. Defaults to True

5.1 Plugin attachments

`wiki.plugins.attachments.settings.`**`ANONYMOUS = False`**

Allow anonymous users upload access (not nice on an open network) `WIKI_ATTACHMENTS_ANONYMOUS` can override this, otherwise the default in `wiki.conf.settings` is used.

`wiki.plugins.attachments.settings.`**`APPEND_EXTENSION = True`**

Store files always with an appended .upload extension to be sure that something nasty does not get executed on the server. SAFETY FIRST!

`wiki.plugins.attachments.settings.`**`FILE_EXTENSIONS = ['pdf', 'doc', 'odt', 'docx', 'txt']`**

Allowed extensions for attachments, empty to disallow uploads completely. If `WIKI_ATTACHMENTS_APPEND_EXTENSION` files are saved with an appended ".upload" to the file to ensure that your web server never actually executes some script. The extensions are case insensitive. You are asked to explicitly enter all file extensions that you want to allow. For your own safety. Note: this setting is called WIKI_ATTACHMENTS_EXTENSIONS not WIKI_ATTACHMENTS_FILE_EXTENTIONS

`wiki.plugins.attachments.settings.`**`STORAGE_BACKEND =`**
`<django.core.files.storage.DefaultStorage object>`

Storage backend to use, default is to use the same as the rest of the wiki, which is set in `WIKI_STORAGE_BACKEND`, but you can override it with `WIKI_ATTACHMENTS_STORAGE_BACKEND`.

`wiki.plugins.attachments.settings.`**`UPLOAD_PATH = 'wiki/attachments/%aid/'`**

Where to store article attachments, relative to `MEDIA_ROOT`. You should NEVER enable directory indexing in `MEDIA_ROOT/UPLOAD_PATH`! Actually, you can completely disable serving it, if you want. Files are sent to the user through a Django view that reads and streams a file.

`wiki.plugins.attachments.settings.`**`UPLOAD_PATH_OBSCURIFY = True`**

> Should the upload path be obscurified? If so, a random hash will be added to the path such that someone can not guess the location of files (if you have restricted permissions and the files are still located within the web server's file system).

`wiki.plugins.attachments.settings.`**`USE_LOCAL_PATH = True`**

> Important for e.g. S3 backends: If your storage backend does not have a .path attribute for the file, but only a .url attribute, you should use False. This will reveal the direct download URL so it does not work perfectly for files you wish to be kept private.

5.2 Plugin editsection

`wiki.plugins.editsection.settings.`**`LINK_TEXT = '[edit]'`**

> Text used for the section edit links which will appear next to section headers. These links allow editing only the text of one particular section.

`wiki.plugins.editsection.settings.`**`MAX_LEVEL = 3`**

> Add "[edit]" links to all section headers till this level. By using these links editing only the text from the selected section is possible.

5.3 Plugin images

`wiki.plugins.images.settings.`**`ANONYMOUS = False`**

> Allow anonymous users upload access (not nice on an open network). `WIKI_IMAGES_ANONYMOUS` can override this, otherwise the default in `wiki.conf.settings` is used.

`wiki.plugins.images.settings.`**`IMAGE_PATH = 'wiki/images/%aid/'`**

> Location where uploaded images are stored. `%aid` is replaced by the article id.

`wiki.plugins.images.settings.`**`IMAGE_PATH_OBSCURIFY = True`**

> Should the upload path be obscurified? If so, a random hash will be added to the path such that someone can not guess the location of files (if you have restricted permissions and the files are still located within the web server's file system).

`wiki.plugins.images.settings.`**`STORAGE_BACKEND = <django.core.files.storage.DefaultStorage object>`**

> Storage backend to use, default is to use the same as the rest of the wiki, which is set in `WIKI_STORAGE_BACKEND`, but you can override it with `WIKI_IMAGES_STORAGE_BACKEND`.

`wiki.plugins.images.settings.`**`THUMBNAIL_SIZES = {'default': '250x250', 'large': '500x500', 'medium': '300x300', 'orig': None, 'small': '150x150'}`**

> Size for the image thumbnail included in the HTML text. If no specific size is given in the markdown tag the `default` size is used. If a specific size is given in the markdown tag that size is used.

5.4 Plugin links

`wiki.plugins.links.settings.`**`LOOKUP_LEVEL = 2`**

> If a relative slug is used in a wiki markdown link and no article is found with the given slug starting at the current
> articles level a link to a not yet existing article is created. Creating the article can be done by following the link.
> This link will be relative to LOOKUP_LEVEL. This should be the level that most articles are created at.

5.5 Plugin macros

`wiki.plugins.macros.settings.`**`METHODS = ('article_list', 'toc')`**

> List of markdown extensions this plugin should support. `article_list` inserts a list of articles from the current
> level. `toc` inserts a table of contents matching the headings.

DEVELOPER GUIDE

6.1 Setting up a development environment

- Fork and clone the django-wiki repo from Github, `cd` into it.

- Create and activate a virtualenv for developing django-wiki. Ensure you are using recent setuptools and pip.

- Install the requirements:

```
$ pip install --upgrade pip setuptools
$ pip install -e .[devel]
$ pip install tox
```

6.2 Running the test project

In order to quickly get setup, there is a project in the git repository.

The folder **testproject/** contains a pre-configured django project and an sqlite database. Login for django admin is `admin:admin`. This project should always be maintained, but please do not commit changes to the SQLite database as we only care about its contents in case data models are changed.

6.3 Tests

6.3.1 Running tests

To run django-wiki's tests, run `make test` or `pytest`

If you want to test for more **environments**, install "tox" (`pip install tox`) and then just run `tox` to run the test suite on multiple environments.

To run **specific tests**, see `pytest --help`.

To include Selenium tests, you need to have `Xvfb` installed (usually via system-provided package manager), chromedriver and set the environment variable INCLUDE_SELENIUM_TESTS=1. For example, run tests with (depending on whether you want to test directly or via tox):

```
INCLUDE_SELENIUM_TESTS=1 pytest
INCLUDE_SELENIUM_TESTS=1 tox
```

If you wish to also show the browser window while running the functional tests, set the environment variable `SELENIUM_SHOW_BROWSER=1` in *addition* to `INCLUDE_SELENIUM_TESTS=1`, for example:

```
INCLUDE_SELENIUM_TESTS=1 SELENIUM_SHOW_BROWSER=1 pytest
```

6.3.2 Writing tests

Tests generally fall into a few categories:

- Testing at the model level. These test cases should inherit from `tests.base.TestBase`.

- Tests for views that return HTML. We normally use [django-functest](#) for these, especially if the page involves forms and handling of POST data. Test cases should inherit from `tests.base.WebTestBase` and `tests.base.SeleniumBase` - see `tests.core.test_views.RootArticleViewTestsBase`, `RootArticleViewTestsWebTest` and `RootArticleViewTestsSelenium` for an example.

  (In the past the Django test Client was used for these, and currently there are still a lot of tests written in this style. These should be gradually phased out where possible, because the test Client does a poor job of replicating what browsers and people actually do.

- Tests for views that return JSON or other non-HTML. These test cases should inherit from `tests.base.DjangoClientTestBase`.

There are also other mixins in `tests.base` that provide commonly used fixtures for tests e.g. a root article.

> **Warning:** Views should be written so that as far as possible they work without Javascript, and can be tested using the fast WebTest method, rather than relying on the slow and fragile Selenium method. Selenium tests are not run by default.

6.4 Types of Contributions

6.4.1 Report Bugs

Report bugs at https://github.com/django-wiki/django-wiki/issues.

If you are reporting a bug, please include:

- Your operating system name and version.

- Any details about your local setup that might be helpful in troubleshooting.

- Detailed steps to reproduce the bug.

6.4.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

6.4.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

6.4.4 Write Documentation

Django-wiki could always use more documentation, whether as part of the official django-wiki docs, in docstrings, or even on the web in blog posts, articles, and such.

6.4.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/django-wiki/django-wiki/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

6.5 Get Started!

Ready to contribute? Here's how to set up *django-wiki* for local development.

1. Fork the *django-wiki* repo on GitHub.

2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/django-wiki.git
```

3. Go to your fork and install our pre-commit hooks which verify the code for errors:

```
$ pip install pre-commit
$ pre-commit install
```

4. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv django-wiki
$ cd django-wiki/
$ pip install -e '.[devel]'
```

5. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

6. As you are making changes you may want to verify that changes are passing all the relevant functional/unit tests:

```
$ pytest
```

7. If you made changes related to the style sheets (SCSS), you need to install sassc (`sudo apt install sassc`) and run this to compile css:

```
$ make assets
```

8. When you're done making changes, perform one final round of testing, and also ensure relevant tests pass with all supported Python versions with tox:

```
$ pytest
$ # Necessary to run "pip install tox" firstly
$ tox # Runs all tests that pytest would run, just with various Python/Django↵
↪combinations
```

9. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

10. Submit a pull request through the GitHub website.

6.6 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.

2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.

3. The pull request should work for Python 3.6, 3.7, 3.8 and for PyPy. Check the status messages that are automatically generated for your pull request.

6.7 Tips

To run a subset of tests:

```
$ pytest tests/core/test_basic.py # All tests from a single file.
$ pytest tests/core/test_basic.py::URLPathTests # All tests from a single class.
$ pytest tests/core/test_basic.py::URLPathTests::test_manager # Just one test.
```

6.8 Roadmap

The best way to contribute is to use our Github issue list to look at current wishes. The list is found here:

https://github.com/django-wiki/django-wiki/issues/

If you want to add a feature, consider writing a plugin. Please create an issue to discuss whether your plugin idea is a core plugin (`wiki.plugins.*`) or external plugin. If there are additions needed to the plugin API, we can discuss that as well! A discussion is always welcome in a Github issue.

Generally speaking, we need more **unit tests** to improve coverage, and new features will not be accepted without tests. To add more stuff to the project without tests wouldn't be fair to the project or your hard work. We use coverage metrics to see that each new contribution does not significantly impact test coverage.

TIPS & FAQ

7.1 FAQ

7.1.1 Q: Why can't I move articles?

A: Moving articles is not trivial. Here are a couple of reasons:

- Other articles may link to them.

- Permissions may change if you move the articles into a different hierarchy

- We keep revisions of stuff, so the action of moving an article will create a new revision.

- ...but what if the revision is reverted and we had automatically renamed stuff?

Because it isn't trivial to move articles, the work has delayed somewhat.

Resources:

- Pull Request #461

- Issue #154

7.1.2 Q: Why do I keep getting *"This slug conflicts with an existing URL."*

A: When validating a slug, django-wiki will verify through *Settings*.``WIKI_CHECK_SLUG_URL_AVAILABLE`` (default: `True`) that the URL is not already occupied.

So if you keep getting an error that the "slug" isn't available, it's probably because you left another URL pattern interfearing with django-wiki's by letting your pattern (regexp) be too open. Forgetting a closing `$` is a common mistake.

7.2 Disqus comment embed

This page describes how to embed the Disqus comment system on each wiki page.

Put the following as `wiki/base.html` somewhere in your `TEMPLATE_DIRS`:

```
{% extends "wiki/base_site.html" %}
{% load sekizai_tags %}

{% block wiki_body %}
  {{ block.super }}
```

(continues on next page)

```
{% block wiki_footer_logo %}
{% endblock wiki_footer_logo %}
{% if selected_tab == 'view' %}
  {% addtoblock "js" %}
  <script type="text/javascript">
  (function(){
    $("#wiki-footer p").eq(0).after('<div id="disqus_thread"></div>')
  })();
  var disqus_shortname = 'your_disqus_shortname';
  (function() {
    var dsq = document.createElement('script'); dsq.type = 'text/javascript'; dsq.
→async = true;
    dsq.src = '//' + disqus_shortname + '.disqus.com/embed.js';
    (document.getElementsByTagName('head')[0] || document.getElementsByTagName('body
→')[0]).appendChild(dsq);
  })();
  </script>
  {% endaddtoblock %}
{% endif %}
{% endblock wiki_body %}
```

Replace `your_disqus_sortname` with your disqus sortname.

See also in *Customization*.

7.3 Quick tips

1. **Account handling:** There are simple views that handle login, logout and signup. They are on by default. Make sure to set `settings.LOGIN_URL` to point to your login page as many wiki views may redirect to a login page.

2. **Syntax highlighting:** Python-Markdown has a pre-shipped codehilite extension which works perfectly, so add something like:

```
WIKI_MARKDOWN_KWARGS = {
    'extensions': [
        'footnotes',
        'attr_list',
        'headerid',
        'extra',
        'codehilite',
    ]
}
```

to your settings. Currently, django-wiki ships with a stylesheet that already has the syntax highlighting CSS rules built-in. Oh, and you need to ensure `pip install pygments` because Pygments is what the codehilite extension is using!

3. **Project Templates:** Create new django-wiki projects quickly and easily using django-wiki project templates https://github.com/django-wiki/django-wiki-project-template

DJANGO-WIKI

8.1 Django support

The below table explains which Django versions are supported.

Release	Django	Upgrade from
0.8.x	2.2, 3.0, 3.1, 3.2, 4.0	0.7
0.7.x	2.2, 3.0, 3.1, 3.2	0.5 or 0.6
0.6.x	2.1, 2.2, 3.0	0.5
0.5.x	2.1, 2.2	0.4
0.4.x	1.11, 2.0, 2.1	0.3
0.3.x	1.8, 1.9, 1.10, 1.11	0.2
0.2.x	1.8, 1.9, 1.10	0.1
0.1.x	1.5, 1.6, 1.7	0.0.24
0.0.24	1.4, 1.5, 1.6 1.7 (unstable)	0.0.?

For upgrade instructions, please refer to the Release Notes

8.2 Translations (Transifex)

Django-wiki has almost fully translated into 12 languages, apart from the default (English). But please help out in adding more languages! It's very easy, you don't even need to be a programmer.

Some languages. . .

- Just need a little push, as they are almost fully complete

- Got initiatied and need a new instigator to carry on the ambitions

- Do not exist yet - but you can request them and become the coordinator.

Visit the django-wiki project on Transifex

8.3 Demo

A demo running the latest `master` is available here, sign up for an account to see the notification system, or you can log in with user:`test` and password:`test`.

https://demo.django-wiki.org

8.4 Community

Please use our IRC or mailing list (google group) for getting in touch on development and support. Please do not email developers asking for personal support.

- #django-wiki on irc.freenode.net

- django-wiki@googlegroups.com

- twitter:djangowiki

8.5 *THIS IS A WORK IN PROGRE...*

Currently, the model API is subject to smaller changes, and the plugin API seems pretty stable.

In order to customize the wiki, best idea is to override templates and create your own template tags. Do not make your own hard copy of this repository in order to fiddle with internal parts of the wiki – this strategy will lead you to lose out on future updates with highly improved features and plugins. Possibly security updates as well!

The release cycle has already begun, so you can administer django-wiki through Pypi and pip.

All views are class-based, however don't take it as an encouragement to extend them, unless you are prepared to modify both templates and view classes every time there is an update.

8.6 Contributing

Please read our Developer Guide

8.7 Manifesto

Django needs a mature wiki system appealing to all kinds of needs, both big and small:

- **Be pluggable and light-weight.** Don't integrate optional features in the core.

- **Be open.** Make an extension API that allows the ecology of the wiki to grow in a structured way. Wikipedia consists of over 1100 extension projects written for MediaWiki. We should learn from this.

- **Be smart.** This is the map of tables in MediaWiki - we'll understand the choices of other wiki projects and make our own. After-all, this is a Django project.

- **Be simple.** The source code should *almost* explain itself.

- **Be structured.** Markdown is a simple syntax for readability. Features should be implemented either through easy coding patterns in the content field, but rather stored in a structured way (in the database) and managed through a friendly interface. This gives control back to the website developer, and makes knowledge more usable. Just

ask: Why has Wikipedia never changed? Answer: Because it's knowledge is stored in a complicated way, thus it becomes very static.

8.8 Docs

See the docs/ folder, or read them at:

https://django-wiki.readthedocs.io/en/latest/

If you wish to add something, please ask in the google group or raise an issue if you're in doubt about whether something might change.

8.9 Background

Django-wiki is a rewrite of django-simplewiki, a project from 2009 that aimed to be a base system for a wiki. It proposed that the user should customize the wiki by overwriting templates, but soon learned that the only customization that really took place was that people forked the entire project. We don't want that for django-wiki, we want it to be modular and extendable.

As of now, Django has existed for too long without a proper wiki application. The dream of django-wiki is to become a contestant alongside Mediawiki, so that Django developers can stick to the Django platform even when facing tough challenges such as implementing a wiki.

8.10 Q&A

- **Why is the module named just** `wiki` **?** Because when we tried `pip install wiki`, it returned "No distributions at all found for wiki", so we had to make up for that!

- **What markup language will you use?** Markdown. The markup renderer is not a pluggable part but has been internalized into core parts. Discussion should go here: https://github.com/django-wiki/django-wiki/issues/76

- **Why not use django-reversion?** It's a great project, but if the wiki has to grow ambitious, someone will have to optimize its behavior, and using a third-party application for something as crucial as the revision system is a no-go in this regard.

- **Any support for multiple wikis?** Yes, in an sense you can just imagine that you always have multiple wikis, because you always have hierarchies and full control of their permissions. See this discussion: https://github.com/django-wiki/django-wiki/issues/63

8.11 Requirements

Please refer to current release to see exact version dependencies. And make note that Pillow needs to have certain build dependencies satisfied on your host system.

- Django
- Markdown
- django-mptt
- django-sekizai
- sorl-thumbnail

- Pillow (Python Imaging Library)
- Python>=3.5

8.12 Docker tl;dr

There is a docker container available here: https://github.com/riotkit-org/docker-django-wiki

8.13 Acknowledgements

- The people at edX & MIT for finding and supporting the project both financially and with ideas.
- django-cms for venturing where no django app has gone before in terms of well-planned features and high standards. It's a very big inspiration.
- django-mptt, a wonderful utility for inexpensively using tree structures in Django with a relational database backend.
- spookylukey, jluttine, duvholt, valberg, jdcaballerov, yekibud, bridger, TomLottermann, crazyzubr, and everyone else involved!

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

W