wlan slovenija
A plugin for `cmsplugin-markup` which adds Trac wiki engine support to Django CMS. It enables you to have content in Trac wiki syntax. It also supports Trac macros and it is also integrated with `django-filer` and `cmsplugin-blog`. You can easily (with GUI) add files and images to the content, or any other text-enabled plugin content.
You should just install it somewhere Django can find it, add `cmsplugin_markup` to `INSTALLED_APPS` and add `cmsplugin_markup_tracwiki` to `CMS_MARKUP_OPTIONS`. You should of course also have Trac installed and an otherwise working Django CMS installation. Plugin was tested with 0.12 Trac version.
CHAPTER 2

Wiki Syntax

You should check out the wiki syntax documentation for introduction to Trac wiki engine. As this integration uses this engine directly, everything supported there is also available in Django CMS. (If something is missing or not working properly, please let me know.)

There are some differences:

- not all known wiki macros are enabled by default (like Image) as they are not needed or reasonable
- wiki namespace is not available (as there is no wiki)

There are three new namespaces available:

- cms to access Django CMS pages (using optional reverse ID to identify them) or anything else in Django namespace, accessible by reverse
- filer to access django-filer files (using original filename, current name, SHA-1 hash or stored file path)
- blog to access cmsplugin-blog entries (using slug, and optionally language code)

Examples:

```
[cms:page-with-name Page with reverse ID name]
[cms:admin:index Admin main page]
[filer:original-filename.png File]
[blog:my-first-entry First blog entry]
[blog:en:my-first-entry First blog entry in English]
```

There are two macros which bridges the gap to Django template tags:

- url macro which wraps Django’s url template tag
- now macro which wraps Django’s now template tag (an example of a dynamic macro)

There is a special CMSPlugin macro which renders a Django CMS plugin which was inserted into the wiki markup. Probably you should not use it manually.

Additional macros or other features can be added on request. Or you can just clone the repository and implement them yourself and send me a pull request.
If you want to render Trac syntax in Django templates, you can use `tracwiki` Django template tag from `tracwiki` template tags library:

```
{% load tracwiki %}
{% tracwiki object.trac_content %}
```

If you want to process Trac links (Trac link syntax) in Django templates, you can use `tracwiki_link` Django template tag from `tracwiki` template tags library:

```
{% load tracwiki %}
<a href="{% tracwiki_link "filer:original-filename.png" %}">File</a>
```
Settings

There are the following Django settings.

**CMS_MARKUP_TRAC_INTERTRAC** configures Trac’s InterTrac links. For example:

```python
CMS_MARKUP_TRAC_INTERTRAC = {
    'trac': {
        'TITLE': 'The Trac Project',
        'URL': 'http://trac.edgewall.org',
    },
}
```

allows you to link to the ticket #1234 in Trac’s official installation with `[trac:ticket:1234 #1234]`.

Similar **CMS_MARKUP_TRAC_INTERWIKI** allows general InterWiki links:

```python
CMS_MARKUP_TRAC_INTERWIKI = {
    'wikipedia': {
        'TITLE': 'Wikipedia',
        'URL': 'http://en.wikipedia.org/wiki/',
    },
}
```

**CMS_MARKUP_TRAC_COMPONENTS** configures which additional Trac plugins (components) should be enabled. They should of course be in Python path. And probably defined fully, like:

```python
CMS_MARKUP_TRAC_COMPONENTS = (
    'tracdashessyntax.plugin.DashesSyntaxPlugin',
    'footnotemacro.macro.FootNoteMacro',
    'mathjax.api.MathJaxPlugin',
    'tracmath.tracmath.TracMathPlugin',
)
```

**CMS_MARKUP_TRAC_CONFIGURATION** allows defining any additional Trac configuration options, like:
CMS_MARKUP_TRAC_CONFIGURATION = {
    'tracmath': {
        'cache_dir': os.path.join(PROJECT_PATH, 'tracwiki', 'cache'),
    }
}

And CMS_MARKUP_TRAC_TEMPLATES_DIR specifies directory with Trac templates. Example:

    CMS_MARKUP_TRAC_TEMPLATES_DIR = os.path.join(PROJECT_PATH, 'tracwiki', 'templates')

CMS_MARKUP_TRAC_HEADING_OFFSET configures the heading offset when rendering wiki syntax. Useful when you are including wiki content inside some other content with existing headings. Default is 1 which means that Heading = becomes <h2>Heading</h2>. Setting it to 0 disables this feature.
For development GitHub is used, so source code and issue tracker is found there.
CHAPTER 6

Indices and tables

- genindex
- search