wagtail-robots Documentation

Release dev

Adrian Turjak

Nov 15, 2020

Contents

1 Wagtail Robots In Action 3

2 Installation 9

3 Initialization 11

4 Rules 13

5 URLs 15

6 Caching 17

7 Sitemaps 19

8 Host directive 21

9 Development/Staging Override 23

10 Bugs and feature requests 25

This is a basic Django application for Wagtail to manage robots.txt files following the robots exclusion protocol, complementing the Django Sitemap contrib app.

This started as a fork of Django Robots but because of the differences between the Django Admin and the Wagtail Admin, and other project requirements git history has not been retained.

For installation and configuration instructions, keep reading.

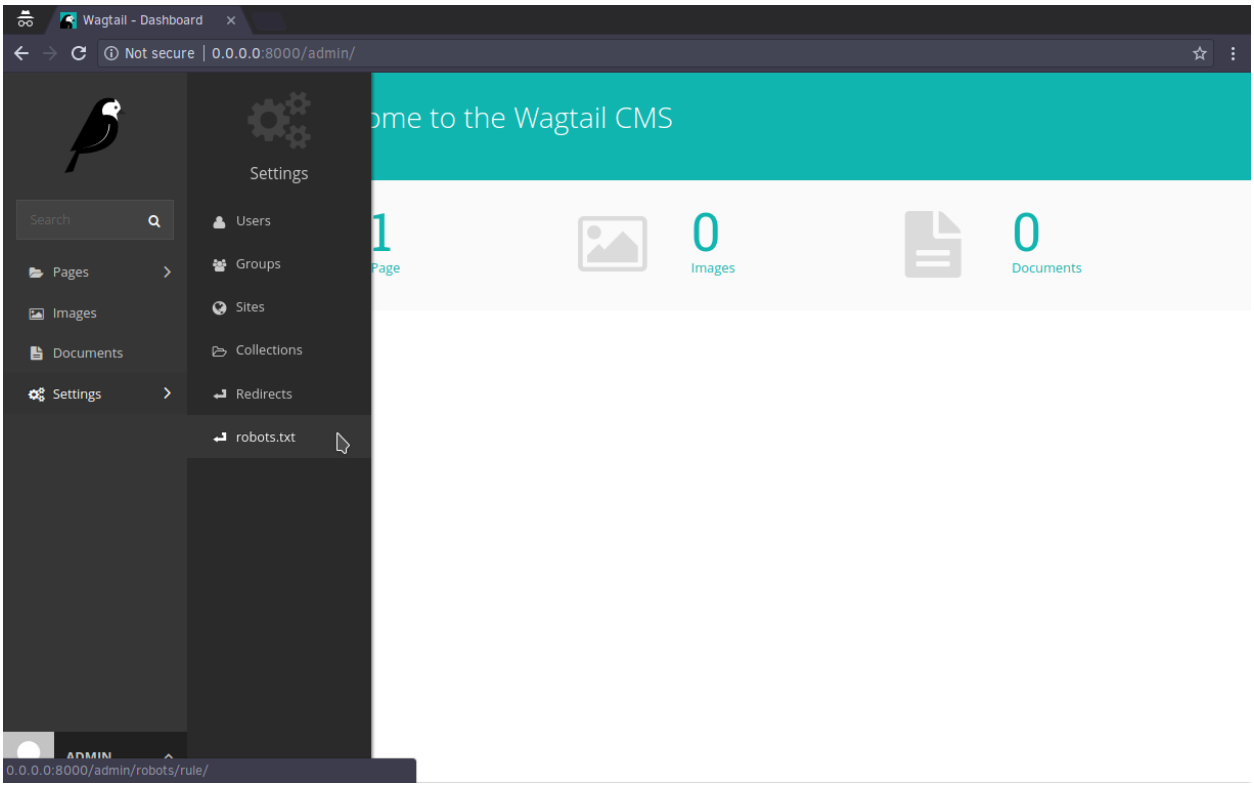
Contents:

CHAPTER 1

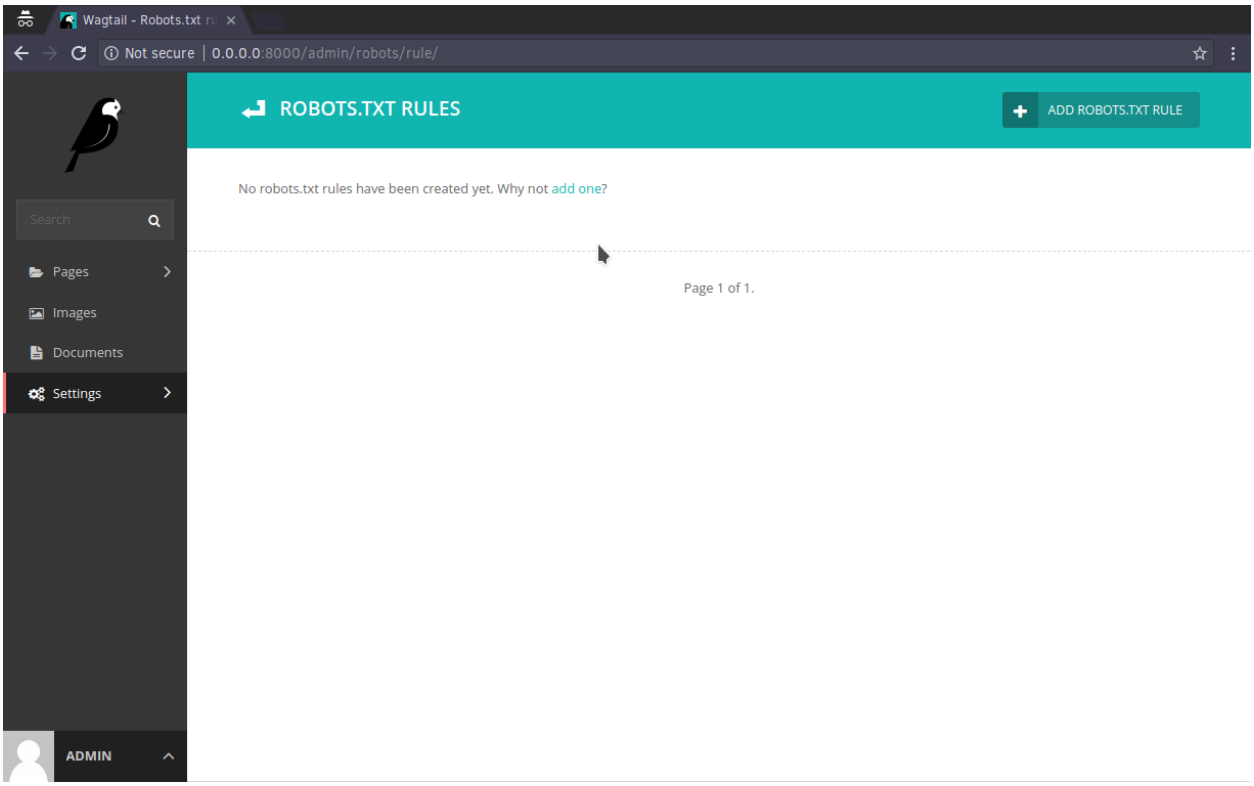
Wagtail Robots In Action

Here are some images so you can see Wagtail Robots in the Wagtail admin interface.

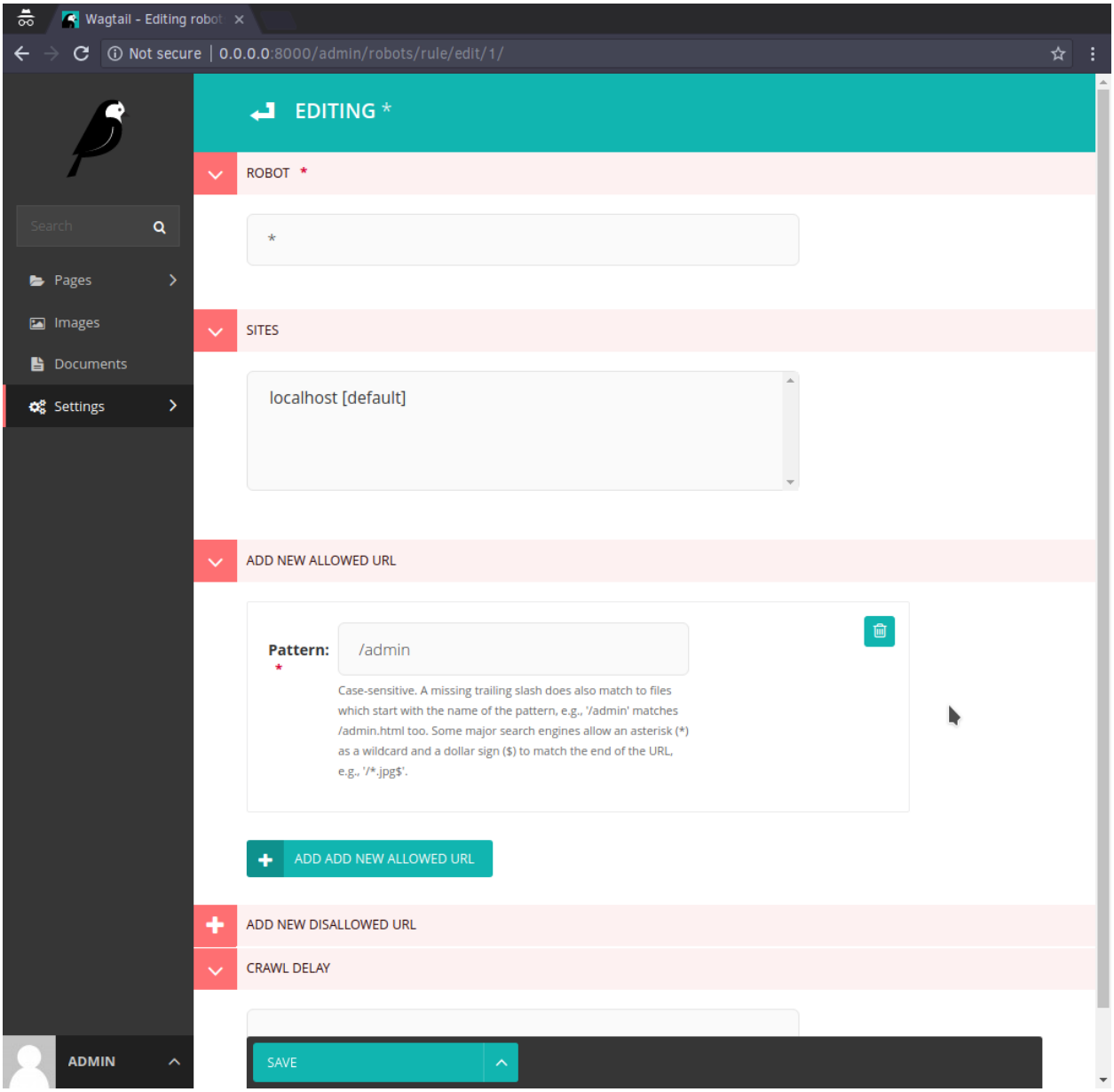
In the menu:



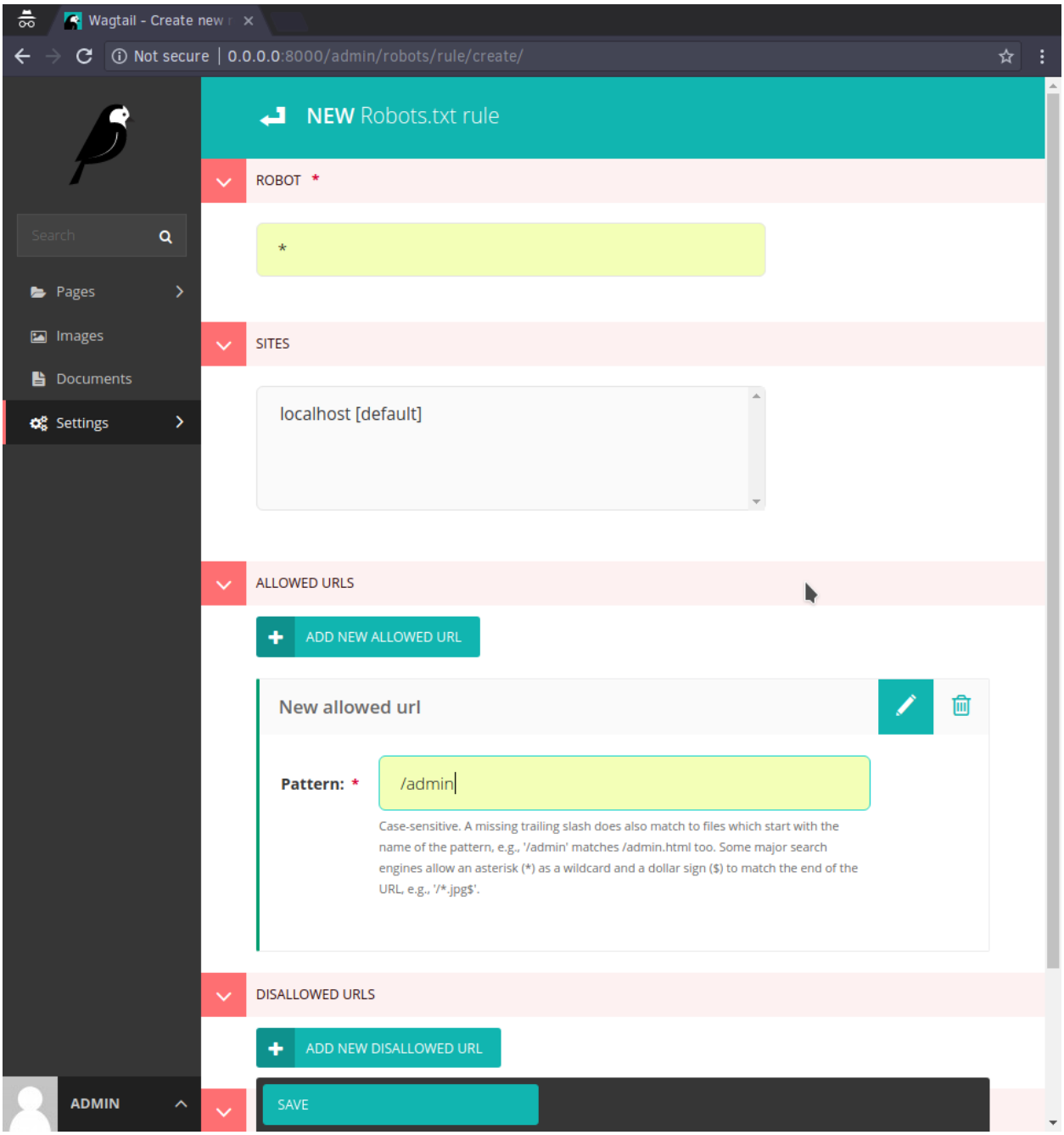
The rules index page:



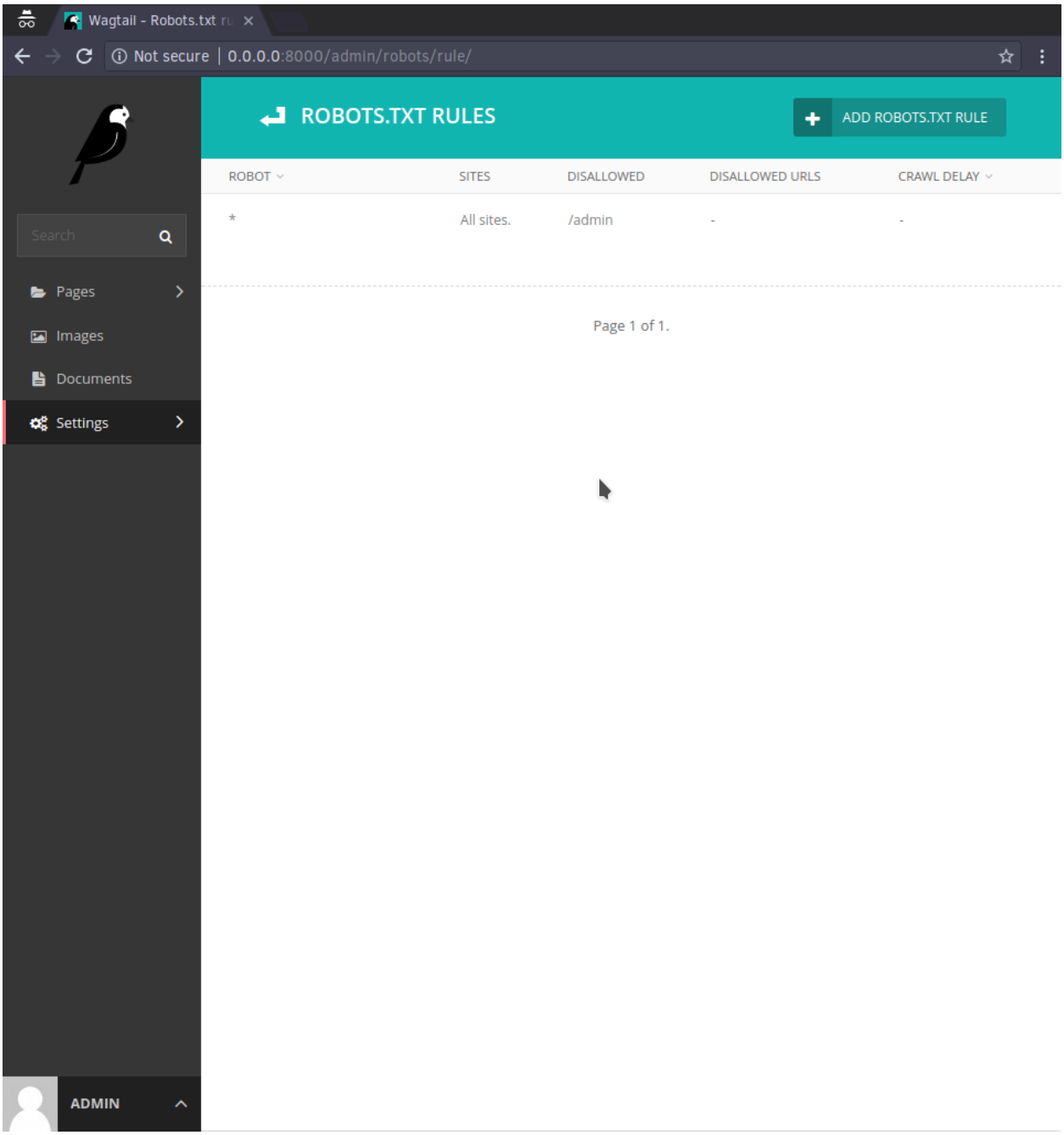
Create/edit view:



Create/edit view with the CondensedInlinePanel:



Index view with at 1 rule created:


Wagtail - Robots.txt ru
Not secure | 0.0.0.0:8000/admin/robots/rule/
ROBOTS.TXT RULES
ADD ROBOTS.TXT RULE
ROBOT
SITES
DISALLOWED
DISALLOWED URLS
CRAWL DELAY
*
All sites.
/admin
-
-
Page 1 of 1.
Search
Pages
Images
Documents
Settings
ADMIN

CHAPTER 2

Installation

Use your favorite Python installer to install it from PyPI:

```
pip install wagtail-robots
```

Or get the source from the application site at:

```
http://github.com/adrian-turjak/wagtail-robots/
```

Then follow these steps:

1. Add `'wagtail.contrib.modeladmin'` and `'robots'` to your INSTALLED_APPS setting.
2. Run the `migrate` management command

You may want to additionally setup the Wagtail sitemap generator.

And if you install or already happen to be using CondensedInlinePanel this library will automatically use it in place of InlinePanel for the Rule create and edit pages.

CHAPTER 3

Initialization

To activate robots.txt generation on your Wagtail site, add this line to your URLconf:

```
url(r'^robots\.txt', include('robots.urls')),
```

This tells Django to build a robots.txt when a robot accesses `/robots.txt`. Then, please migrate your database to create the necessary tables and create `Rule` objects in the admin interface or via the shell.

CHAPTER 4

Rules

`Rule` - defines an abstract rule which is used to respond to crawling web robots, using the robots exclusion protocol, a.k.a. robots.txt.

You can link multiple URL pattern to allows or disallows the robot identified by its user agent to access the given URLs.

The crawl delay field is supported by some search engines and defines the delay between successive crawler accesses in seconds. If the crawler rate is a problem for your server, you can set the delay up to 5 or 10 or a comfortable value for your server, but it's suggested to start with small values (0.5-1), and increase as needed to an acceptable value for your server. Larger delay values add more delay between successive crawl accesses and decrease the maximum crawl rate to your web server.

The Wagtail sites are used to enable multiple robots.txt per Wagtail instance. If no rule exists it automatically allows every web robot access to every URL except Wagtail's admin path (*/admin*).

Please have a look at the database of web robots for a full list of existing web robots user agent strings.

CHAPTER 5

URLs

`Url` - defines a case-sensitive and exact URL pattern which is used to allow or disallow the access for web robots. Case-sensitive.

A missing trailing slash does also match files which start with the name of the given pattern, e.g., `'/admin'` matches `/admin.html` too.

Some major search engines allow an asterisk (`*`) as a wildcard to match any sequence of characters and a dollar sign (`$`) to match the end of the URL, e.g., `'/*.jpg$'` can be used to match all jpeg files.

CHAPTER 6

Caching

You can optionally cache the generation of the `robots.txt`. Add or change the `ROBOTS_CACHE_TIMEOUT` setting with a value in seconds in your Django settings file:

```
ROBOTS_CACHE_TIMEOUT = 60*60*24
```

This tells Django to cache the `robots.txt` for 24 hours (86400 seconds). The default value is `None` (no caching).

If you need to, you can also specify exactly which cache to use:

```
ROBOTS_CACHE_ALIAS="robots"
```

Unless specified otherwise it will use the `default` cache.

CHAPTER 7

Sitemaps

By default a `Sitemap` statement is automatically added to the resulting robots.txt by reverse matching the URL of the installed Wagtail Sitemap app. This is especially useful if you allow every robot to access your whole site, since it then gets URLs explicitly instead of searching every link.

To change the default behaviour to omit the inclusion of a sitemap link, change the `ROBOTS_USE_SITEMAP` setting in your Django settings file to:

```
ROBOTS_USE_SITEMAP = False
```

In case you want to use specific sitemap URLs instead of the one that is automatically discovered, change the `ROBOTS_SITEMAP_URLS` setting to:

```
ROBOTS_SITEMAP_URLS = [
    'http://www.example.com/sitemap.xml',
]
```

If the sitemap is wrapped in a decorator, dotted path reverse to discover the sitemap URL does not work. To overcome this, provide a name to the sitemap instance in `urls.py`:

```
urlpatterns = [
    ...
    url(r'^sitemap.xml$', cache_page(60)(sitemap_view), {'sitemaps': [...]}, name=
↪'cached-sitemap'),
    ...
]
```

and inform django-robots about the view name by adding the following setting:

```
ROBOTS_SITEMAP_VIEW_NAME = 'cached-sitemap'
```

Use `ROBOTS_SITEMAP_VIEW_NAME` also if you use custom sitemap views.

CHAPTER 8

Host directive

By default a `Host` statement is automatically added to the resulting robots.txt to avoid mirrors and select the main website properly.

To change the default behaviour to omit the inclusion of host directive, change the `ROBOTS_USE_HOST` setting in your Django settings file to:

```
ROBOTS_USE_HOST = False
```

if you want to prefix the domain with the current request protocol (**http** or **https** as in `Host: https://www.mysite.com`) add this setting:

```
ROBOTS_USE_SCHEME_IN_HOST = True
```

CHAPTER 9

Development/Staging Override

Sometimes when you have duplicate database content in both a production and staging website, it can be useful to override any and all database entries for the this application and explicitly disallow all.

To do that add this setting:

```
ROBOTS_DISALLOW_ALL = True
```

The resulting *robots.txt* will look as follows:

```
User-agent: *
Disallow: /
```

CHAPTER 10

Bugs and feature requests

As always your mileage may vary, so please don't hesitate to send feature requests and bug reports:

https://github.com/adrian-turjak/wagtail-robots/issues