

---

# **django-hitcount Documentation**

***Release 1.1.1***

**Damon Timm**

October 04, 2015



<b>1</b>	<b>Overview</b>	<b>3</b>
1.1	Requirements and Compatibility . . . . .	3
<b>2</b>	<b>Installation and Usage</b>	<b>5</b>
2.1	Settings.py . . . . .	5
2.2	Urls.py . . . . .	5
2.3	Models.py . . . . .	5
2.4	Template Magic . . . . .	6
<b>3</b>	<b>Additional Settings</b>	<b>9</b>
3.1	HITCOUNT_KEEP_HIT_ACTIVE . . . . .	9
3.2	HITCOUNT_HITS_PER_IP_LIMIT . . . . .	9
3.3	HITCOUNT_EXCLUDE_USER_GROUP . . . . .	9
3.4	HITCOUNT_KEEP_HIT_IN_DATABASE . . . . .	9
<b>4</b>	<b>Management Commands</b>	<b>11</b>
<b>5</b>	<b>Example Project</b>	<b>13</b>
<b>6</b>	<b>Contribution and Testing</b>	<b>15</b>
6.1	Testing . . . . .	15
<b>7</b>	<b>Additional Authors and Thanks</b>	<b>17</b>
<b>8</b>	<b>Changelog</b>	<b>19</b>
8.1	Version 1.1.1: . . . . .	19
8.2	Version 1.1.0: . . . . .	19
<b>9</b>	<b>Issues</b>	<b>21</b>



Django-Hitcount allows you to track the number of hits/views for a particular object.



---

# Overview

---

Django-Hitcount allows you to track the number of hits (views) for a particular object. This isn't meant to be a full-fledged tracking application or a real analytic tool; it's just a basic hit counter.

How one tracks a “hit” or “view” of a web page is not such a simple thing as it might seem. That's why folks rely on Google Analytics or similar tools. It's tough! This is a simple app with some settings and features that should suit the basic needs of smaller sites.

It comes ready to track hits (to use the out-of-the-box method, you will need jQuery – although writing your own JavaScript implementation is not hard).

## 1.1 Requirements and Compatibility

Currently supporting Django  $\geq 1.4$  and any Python version supported by your Django version.





---

## Installation and Usage

---

For a working implementation, you can view the [example project](#) on Github.

Install django-hitcount:

```
pip install django-hitcount
```

### 2.1 Settings.py

Add django-hitcount to your `INSTALLED_APPS`:

```
# settings.py
INSTALLED_APPS = (
    ...
    'hitcount'
)
```

View the [additional settings section](#) for a list of the django-hitcount settings that are available.

### 2.2 Urls.py

In your `urls.py` file add the following:

```
# urls.py
urlpatterns = [
    ...
    url(r'hitcount/', include('hitcount.urls', namespace='hitcount')),
]
```

### 2.3 Models.py

There is nothing you are required to do with your own models as that django-hitcount relies on a `GenericForeignKey` to create the relationship to your model's `HitCount`. You can add a `GenericRelation` to your model if you would like to be able to access its `HitCount` model easily:

```
from django.db import models
from django.contrib.contenttypes.fields import GenericRelation

from hitcount.models import HitCount

# here is an example model with a GenericRelation
class MyModel(models.Model):
    ...
    # Note that you need to specify the object_id_field as written below
    hit_count = GenericRelation(HitCount, object_id_field='object_pk')
```

## 2.4 Template Magic

Django-hitcount comes packaged with a [jQuery implementation](#) that works out-of-the-box to record the Hits to an object (be it a blog post, poll, etc). To use the [jQuery implementation](#) you can either include the app's script file (as the documentation below shows) or to copy-paste the script into your own jQuery code. Of course: you could also implement this without relying on jQuery.

Start by loading hitcount tags on the desired templates:

```
{% load hitcount_tags %}
```

### 2.4.1 Recording a Hit

If you want to use the [jQuery implementation](#) in your project, you can add the Javascript file to your template like so:

```
{% load staticfiles %}
<script src="//code.jquery.com/jquery-1.11.3.min.js"></script>
<script src="{% static 'hitcount/hitcount-jquery.js' %}"></script>
```

Then, on your object detail page (blog, page, poll, etc) you inject the needed javascript variables:

```
# use default insertion method for hitcount-jquery.js:
{% insert_hit_count_js_variables for object %}

# or you can use a template variable to inject as you see fit
{% get_hit_count_js_variables for object as hitcount %}
({ hitcount.ajax_url })
({ hitcount.pk })
```

### 2.4.2 Displaying Hit Information

You can retrieve the number of hits for an object many different ways:

```
# Return total hits for an object:
{% get_hit_count for [object] %}

# Get total hits for an object as a specified variable:
{% get_hit_count for [object] as [var] %}

# Get total hits for an object over a certain time period:
{% get_hit_count for [object] within ["days=1,minutes=30"] %}
```

```
# Get total hits for an object over a certain time period as a variable:
{% get_hit_count for [object] within ["days=1,minutes=30"] as [var] %}
```



---

## Additional Settings

---

There are a few additional settings you can use to customize django-hitcount and are set in your `settings.py` file.

### 3.1 HITCOUNT\_KEEP\_HIT\_ACTIVE

This is the number of days, weeks, months, hours, etc (using a `timedelta` keyword argument), that an `Hit` is kept **active**. If a `Hit` is **active** a repeat viewing will not be counted. After the **active** period ends, however, a new `Hit` will be recorded. You can decide how long you want this period to last and it is probably a matter of preference.:

```
# default value
HITCOUNT_KEEP_HIT_ACTIVE = { 'days': 7 }
```

### 3.2 HITCOUNT\_HITS\_PER\_IP\_LIMIT

Limit the number of **active** `Hits` from a single IP address. 0 means that it is unlimited.:

```
# default value
HITCOUNT_HITS_PER_IP_LIMIT = 0
```

### 3.3 HITCOUNT\_EXCLUDE\_USER\_GROUP

Exclude `Hits` from all users in the specified user groups. By default, this is set to an empty list (all users counted). In the example, below, it will exclude all your ‘Editors’:

```
# example value, default is empty tuple
HITCOUNT_EXCLUDE_USER_GROUP = ( 'Editor', )
```

### 3.4 HITCOUNT\_KEEP\_HIT\_IN\_DATABASE

This setting is used with the `hitcount_cleanup` management command and specifies a `timedelta` within which to keep/save `Hits`. Any `Hit` older than the time specified will be removed for the `Hits` table.:

```
# default value
HITCOUNT_KEEP_HIT_IN_DATABASE = { 'days': 30 }
```



---

## Management Commands

---

If you would like to periodically prune your stale Hits you can do so by running the the management command `hitcount_cleanup`:

```
./manage.py hitcount_cleanup
```

The command relies on the setting `HITCOUNT_KEEP_HIT_IN_DATABASE` to determine how far back to prune. See the [additional settings section](#) for more information.





---

## Example Project

---

There is an [example project](#) that demonstrates counting hits on a blog post [using JavaScript](#) on GitHub. It's fairly easy to get this working using the Django development server. Be sure to run this inside your own `virtualenv` (but who doesn't, these days?!):

```
$ git clone git@github.com:thornomad/django-hitcount.git
$ cd django-hitcount/example_project
$ pip install -r requirements.txt # sqlite requires pytz
$ python manage.py migrate      # will load some data fixtures for you
$ python manage.py createsuperuser # for access to the admin portion
$ python manage.py runserver     # should be all set!
```

You can open your favorite browser console and see some of the Javascript `console.log` output indicating what type of `Hit` is being counted (or ignored). When you are ready to work on your own site, check out the [Installation and Usage](#) and [Additional Settings](#) sections.



---

## Contribution and Testing

---

I would love to make it better. Please fork, branch, and push. I plan to do my work in the `develop` branch before moving it to `master` for a real release. You can safely ignore the `devel` branch which is old and stale but has something in it I can't remember why I'm saving.

### 6.1 Testing

You can run the tests by installing the requirements and then executing `runtests.py`:

```
$ pip install -r tests/requirements.txt
$ ./runtests.py
```

This method using `py.test` for test discovery (so older versions of Django can find all the tests). If you would like to use Django's own test runner you can execute:

```
$ ./runtests.py --django
```



---

## Additional Authors and Thanks

---

I've had some help and I'm very grateful! You can always look at the [contributors](#) on GitHub to get a clearer picture.

This doesn't include everyone and if I missed someone let me know I will add it.

Thanks goes to:

- Basil Shubin and his work at [django-hitcount-headless](#) as well as his Russian translations
- ariddell for putting the *setup.py* package together for me



---

## Changelog

---

### 8.1 Version 1.1.1:

- fixed `session_key` returning `None` [#40](#) ( $\geq 1.8.4$ )
- removed requirement for `SESSION_SAVE_EVERY_REQUEST`
- removed `patterns` for `urls.py` ( $\geq 1.9$ )
- updated management command, using `BaseCommand` instead of `NoArgsCommand` ( $\geq 1.9$ )
- added `TEMPLATES` to `conftest.py`

### 8.2 Version 1.1.0:

- added tests (lots of them)
- added documentation
- support for Django 1.4.x - 1.8.x
- support for Python 3.x
- created an example project
- squashed bugs
- released to pip
- more, I'm sure!





---

### Issues

---

Use the GitHub [issue tracker](#) for django-hitcount to submit bugs, issues, and feature requests.