
Django User Sessions Documentation

Release 1.5.1

Bouke Haarsma

Aug 11, 2017

Contents

1	Installation	3
1.1	GeoIP	3
1.2	IP when behind a proxy	3
2	Usage	5
2.1	Current session	5
2.2	All sessions	5
2.3	Generic views	5
2.4	Admin views	6
3	Reference	7
3.1	Middleware	7
3.2	Models	7
3.3	Session Backends	7
3.4	Template Tags	7
3.5	Views	8
3.6	Unit tests	8
4	Release Notes	9
4.1	1.5.1	9
4.2	1.5.0	9
4.3	1.4.0	9
4.4	1.3.1	10
4.5	1.3.0	10
4.6	1.2.0	10
4.7	1.1.1	10
4.8	1.1.0	10
4.9	1.0.0	10
4.10	1.0.0-beta1	10
4.11	0.1.4	11
4.12	0.1.3	11
4.13	0.1.2	11
4.14	0.1.1	11
4.15	0.1.0	11
5	Indices and tables	13

Django includes excellent built-in sessions, however all the data is hidden away into base64 encoded data. This makes it very difficult to run a query on all active sessions for a particular user. *django-user-sessions* fixes this and makes session objects a first class citizen like other ORM objects.

Contents:

CHAPTER 1

Installation

1. `pip install django-user-sessions`
2. In `INSTALLED_APPS` replace `'django.contrib.sessions'` with `'user_sessions'`.
3. In `MIDDLEWARE` or `MIDDLEWARE_CLASSES` replace `'django.contrib.sessions.middleware.SessionMiddleware'` with `'user_sessions.middleware.SessionMiddleware'`.
4. Add `SESSION_ENGINE = 'user_sessions.backends.db'`.
5. Add `url(r'', include('user_sessions.urls', 'user_sessions'))`, to your `urls.py`.
6. Make sure `LOGOUT_REDIRECT_URL` is set to some page to redirect users after logging out.
7. Run `python manage.py syncdb` (or `migrate`) and browse to `/account/sessions/`.

GeoIP

You need to setup GeoIP for the location detection to work. See the Django documentation on [installing GeoIP](#). For Django versions 1.9 and newer, [GeoIP2](#) should be used instead as GeoIP was deprecated in 1.9.

IP when behind a proxy

If you're running Django behind a proxy like nginx, you will have to set the `REMOTE_ADDR` META header manually using a middleware, to stop it from always returning the ip of the proxy (e.g. 127.0.0.1 in many cases).

An example middleware to fix this issue is [django-xforwardedfor-middleware](#) which simply does this for each request:

```
request.META['REMOTE_ADDR'] = request.META['HTTP_X_FORWARDED_FOR'].split(',')[0].strip()
```

Your particular configuration may vary, *X-Forwarded-For* must be set by a proxy that you have control over, otherwise it might be spoofed by the client.

Current session

The current session is available on the request, just like the normal session middleware makes the session available:

```
def my_view(request):  
    request.session
```

All sessions

To get the list of a user's sessions:

```
sessions = user.session_set.filter(expire_date__gt=now())
```

You could logout the user everywhere:

```
user.session_set.all().delete()
```

Generic views

There are two views included with this application, *SessionListView* and *SessionDeleteView*. Using this views you have a simple, but effective, user session management that even looks great out of the box:

Active Sessions

Location	Device	Last Activity	End Session
Netherlands (xx.xx.xx.xx)	Safari on iPhone	5 minutes ago	End Session
Zwolle, Netherlands (xx.xx.xx.xx)	Safari on Mac OS X	11 minutes ago (this session)	End Session

Template tags

Two template tags are included `device()` and `location()`. These can be used for respectively humanizing the user agent string and showing an approximate location of the IP address:

```
{% load user_sessions %}
{{ session.user_agent|device }} -> Safari on OS X
{{ session.ip|location }}      -> Zwolle, The Netherlands
```

Admin views

The user's IP address and user agent are also stored on the session. This allows to show a list of active sessions to the user in the admin:

Select session to change [Add session](#) [+](#)

Action: 0 of 2 selected

☐	Ip address	User	Is valid	Location	Device
☐	xx.xx.xx.xx	bouke	✓	Zwolle, Netherlands	Safari on Mac OS X
☐	xx.xx.xx.xx	bouke	✓	Netherlands	Safari on iPhone

2 sessions

Filter

- By Is Valid**
 - [All](#)
 - Active
 - Expired
- By Owner**
 - [All](#)
 - Self

Middleware

`class user_sessions.middleware.SessionMiddleware (get_response=None)`
Middleware that provides ip and user_agent to the session store.

Models

Session Backends

Template Tags

`user_sessions.templatetags.user_sessions.device (value)`
Transform a User Agent into human readable text.

Example output:

- Safari on iPhone
- Chrome on Windows 8.1
- Safari on OS X
- Firefox
- Linux
- None

`user_sessions.templatetags.user_sessions.location (value)`
Transform an IP address into an approximate location.

Example output:

- Zwolle, The Netherlands
- The Netherlands
- None

Views

class `user_sessions.views.SessionListView` (**kwargs)

View for listing a user's own sessions.

This view shows list of a user's currently active sessions. You can override the template by providing your own template at `user_sessions/session_list.html`.

class `user_sessions.views.SessionDeleteView` (**kwargs)

View for deleting a user's own session.

This view allows a user to delete an active session. For example log out a session from a computer at the local library or a friend's place.

Unit tests

1.5.1

- Updated documentation for GeoIP2 library
- Correctly detect macOS version on Firefox

1.5.0

- Added Django 1.11 support
- Added support for GeoIP2 library
- Added detection of Windows 10 and macOS from user-agent
- Fixed #73 – Error when deleting individual session from list view
- Fixed #74 – user agent not being shown in list view
- Resolved Django’s deprecation warnings (preliminary Django 2.0 support)
- Make templatetags return None instead of ‘unknown’, provide your own fallback value with *default_if_none*:
- Allow translation of fallback values

1.4.0

- Added Django Channels support
- Fixed #62 – Provide request.user in signals
- Ending current session will logout instead, make sure LOGOUT_REDIRECT_URL is set

1.3.1

- Added Django 1.10 support

1.3.0

- Added Django 1.9 support
- Dropped support for Django 1.7 and below

1.2.0

- New feature: delete all-but-current sessions
- Added clearsessions command

1.1.1

- Added Django 1.8 support

1.1.0

- Fixed #14 – Truncate long user_agents
- Fixed #23 – Cannot use admin view search
- Added Django 1.7 migrations

1.0.0

No changes from 1.0.0-beta1.

1.0.0-beta1

- #8 – Consistent URL patterns
- #11 – Support Django 1.6's ATOMIC_REQUESTS
- German translation added

0.1.4

- Python 3.4 support
- Django 1.7 (beta) support
- Italian translation added
- Chinese translation added
- Arabic translation updated

0.1.3

- Documentation
- Hebrew translation added
- Arabic translation added
- Fixed #3 – Reset user_id on logout
- Fixed #4 – Add explicit license text

0.1.2

- Ship with default templates
- Added Dutch translation

0.1.1

- Added South migrations

0.1.0

- Initial release

CHAPTER 5

Indices and tables

- `genindex`
- `modindex`
- `search`

u

`user_sessions.templatetags.user_sessions,`
[7](#)

D

device() (in module `user_sessions.templatetags.user_sessions`),
[7](#)

L

location() (in module `user_sessions.templatetags.user_sessions`),
[7](#)

S

SessionDeleteView (class in `user_sessions.views`), [8](#)

SessionListView (class in `user_sessions.views`), [8](#)

SessionMiddleware (class in `user_sessions.middleware`),
[7](#)

U

`user_sessions.templatetags.user_sessions` (module), [7](#)