# Django User Sessions Documentation

***Release 1.0.0***

**Bouke Haarsma**

June 15, 2014

Contents

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_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. Run `python manage.py syncdb` (or `migrate`) and start hacking!

## 1.1 GeoIP

You need to setup GeoIP for the location detection to work. See the Django documentation on installing GeoIP.

CHAPTER 2

# Usage

## 2.1 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
```

## 2.2 All sessions

To get the list of a user's sessions:

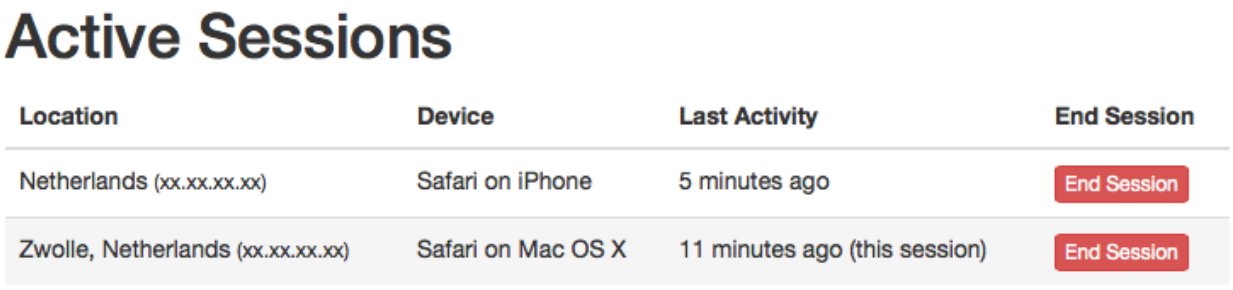
```
sessions = user.session_set.filter(expire_date__gt=now())
```

You could logout the user everywhere:

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

## 2.3 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:



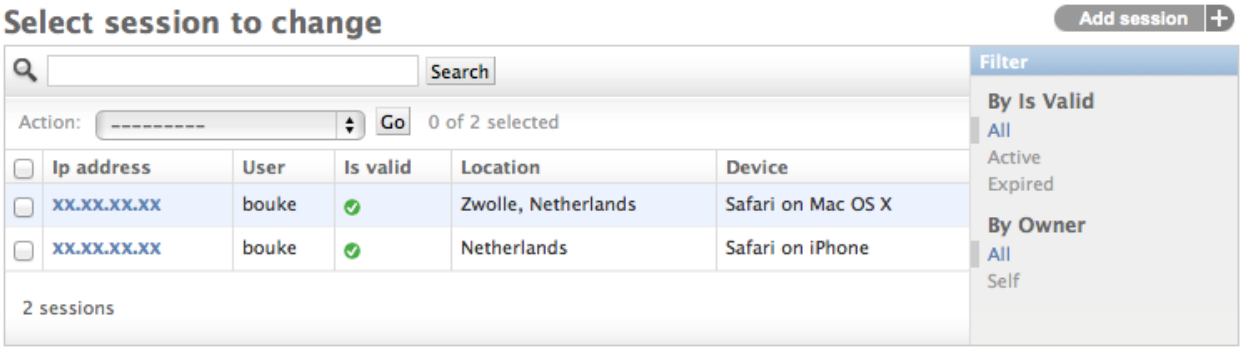
### 2.3.1 Template tags

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
```

## 2.4 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:



CHAPTER 3

# Reference

## 3.1 Middleware

**class** user_sessions.middleware.**SessionMiddleware**
Middleware that provides ip and user_agent to the session store.

## 3.2 Models

**class** user_sessions.models.**Session**(**args*, ***kwargs*)
Session objects containing user session information.

Django provides full support for anonymous sessions. The session framework lets you store and retrieve arbitrary data on a per-site-visitor basis. It stores data on the server side and abstracts the sending and receiving of cookies. Cookies contain a session ID – not the data itself.

Additionally this session object providers the following properties: `user`, `user_agent` and `ip`.

## 3.3 Session Backends

**class** user_sessions.backends.db.**SessionStore**(*user_agent*, *ip*, *session_key=None*)
Implements database session store.

## 3.4 Template Tags

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

Example output:

- Safari on iPhone
- Chrome on Windows 8.1
- Safari on OS X

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

Example output:

- Zwolle, The Netherlands
- unknown

## 3.5 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 locking out a session from a computer at the local library or a friend's place.

## 3.6 Unit tests

**class** user_sessions.utils.tests.**Client**(*enforce_csrf_checks=False*, ***defaults*)
Custom implementation of django.test.Client.

It is required to perform tests that require to login in sites using django-user-sessions since its implementation of SessionStore has to required parameters which is not in concordance with what is expected from the original Client

CHAPTER 4

# Release Notes

## 4.1 1.0.0

*No changes from 1.0.0-beta1.*

## 4.2 1.0.0-beta1

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

## 4.3 0.1.4

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

## 4.4 0.1.3

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

## 4.5 0.1.2

- Ship with default templates
- Added Dutch translation

## 4.6 0.1.1

- Added South migrations

## 4.7 0.1.0

- Initial release

CHAPTER 5

# Indices and tables

- *genindex*
- *modindex*
- *search*

## u