django-session-security Documentation

Release 1.2.2

James Pic

Apr 20, 2017

Contents

This app provides a mechanism to logout inactive authenticated users. An inactive browser should be logged out automatically if the user left his workstation, to protect sensitive data that may be displayed in the browser. It may be useful for CRMs, intranets, and such projects.

For example, if the user leaves for a coffee break, this app can force logout after say 5 minutes of inactivity.

CHAPTER 1

Requirements

- Python 2.7
- jQuery 1.7+
- Django 1.4+
- django.contrib.staticfiles or django-staticfiles (included in Pinax) or you're on your own

CHAPTER 2

Resources

You could subscribe to the mailing list ask questions or just be informed of package updates.

- Git graciously hosted by GitHub,
- Documentation graciously hosted by RTFD,
- Package graciously hosted by PyPi,
- Mailing list graciously hosted by Google

Contents:

Quick setup

The purpose of this documentation is to get you started as fast as possible, because your time matters and you probably have other things to worry about.

Quick install

Install the package:

```
pip install django-session-security
# or the development version
pip install -e git+git://github.com/yourlabs/django-session-security.git#egg=django-
↪session-security
```

For static file service, add to `settings.INSTALLED_APPS`:

```
'session_security',
```

Add to `settings.MIDDLEWARE_CLASSES`, **after** django's AuthenticationMiddleware:

```
'session_security.middleware.SessionSecurityMiddleware',
```

Ensure settings.TEMPLATE_CONTEXT_PROCESSORS has:

```
'django.core.context_processors.request'
```

Add to urls:

```
url(r'session_security/', include('session_security.urls')),
```

At this point, we're going to assume that you have django.contrib.staticfiles working. This means that static files are automatically served with runserver, and that you have to run collectstatic when using another server (fastcgi, uwsgi, and whatnot). If you don't use django.contrib.staticfiles, then you're on your own to manage staticfiles.

Add to your base template:

```
{% include 'session_security/all.html' %}
```

Quick setup

Unless `pinax.apps.account` is found in `settings.INSTALLED_APPS`, configure these settings:

- LOGIN_URL, the absolute url to your login view
- LOGOUT_URL, the absolute url to your logout view

Full documentation

Settings

Middleware

Urls

Views

class `session_security.views.`**`PingView`**`(`***kwargs*`)`

View to update the last activity date time and get it.

Constructor. Called in the URLconf; can contain helpful extra keyword arguments, and other things.

`get``(`*request*, **args*, ***kwargs*`)`

Return the **number of seconds since last activity**. Also, **update session's last activity if** `sinceActivity` GET argument is passed and superior to 0.

- Use the `sinceActivity` and `request.session['session_security']['last_activity']` to calculate the last activity on the client (javascript), and on the server (django).
- If the client reports a later last activity, then the session's last activity variable is updated according to the client.
- Return the time since the last activity. Note that if the user generates activity in a browser tab, but not in the other, both will have the real last activity time because of this approach.

To just query the actual last activity, let `sinceActivity` inferior to 0.

Templates

session_security/dialog.html

```
{% load i18n %}

<div id="session_security_warning" class="session_security" style="display:none">
    <div class="session_security_overlay"></div>
    <div class="session_security_modal">
        <h3>{% trans 'Your session is about to expire' %}</h3>
        <p>{% trans 'Click to extend your session.' %}</p>
    </div>
</div>
```

session_security/all.html

```
{% comment %}
This demonstrates how to setup session security client side stuff on your own.
It provides sensible defaults so you could start with just::

    {% include 'session_security/all.html' %}

{% endcomment %}

{% load i18n %}
{% load url from future %}

{# If the user is not authenticated then there is no session to secure ! #}
{% if request.user.is_authenticated %}

    {# The modal dialog stylesheet, it's pretty light so it should be easy to hack #}
    <link rel="stylesheet" type="text/css" href="{{ STATIC_URL }}session_security/
↪style.css"></link>

    {# Include the template that actually contains the modal dialog #}
    {% include 'session_security/dialog.html' %}

    {# Load SessionSecurity javascript 'class', jquery should be loaded - by you - at␣
↪this point #}
    <script type="text/javascript" src="{{ STATIC_URL }}session_security/script.js"></
↪script>

    {% csrf_token %}

    {# Bootstrap a SessionSecurity instance as the sessionSecurity global variable #}
    <script type="text/javascript">
    var sessionSecurity = new SessionSecurity();

    {# Merge our settings to the sessionSecurity instance, again you can override any␣
↪method/attribute #}
    sessionSecurity = $.extend({
        LOGIN_URL: '{{ request.session.session_security.LOGIN_URL }}',
        LOGOUT_URL: '{{ request.session.session_security.LOGOUT_URL }}',
        EXPIRE_AFTER: {{ request.session.session_security.EXPIRE_AFTER }},
        WARN_AFTER: {{ request.session.session_security.WARN_AFTER }},
        pingUrl: '{% url 'session_security_ping' %}',
        token: $('input[name=csrfmiddlewaretoken]').val(),
    }, sessionSecurity);

    {# Initialize timeouts and events, don't wait for document.ready to reduce clock␣
↪skews #}
    sessionSecurity.initialize();
    </script>
{% endif %}
```

Static files

session_security/script.js

Read the script documentation

session_security/style.css

```
/* credit: http://www.csslab.cl/2008/01/30/ventana-modal-solo-con-css/ */
.session_security_overlay {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: #000;
    z-index:1001;
    opacity:.75;
    -moz-opacity: 0.75;
    filter: alpha(opacity=75);
}

.session_security_modal {
    position: absolute;
    top: 25%;
    left: 25%;
    width: 50%;
    padding: 16px;
    background: #fff;
    color: #333;
    z-index:1002;
    overflow: auto;
    text-align: center;
}
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index