

---

# **django-cookie-consent Documentation**

*Release 0.1.2dev*

**Bojan Mihelac**

**Sep 26, 2017**



---

# Contents

---

|          |                           |          |
|----------|---------------------------|----------|
| <b>1</b> | <b>User Guide</b>         | <b>3</b> |
| 1.1      | Installation . . . . .    | 3        |
| 1.2      | Configuration . . . . .   | 3        |
| 1.3      | Main concepts . . . . .   | 3        |
| 1.4      | Getting started . . . . . | 4        |
| 1.5      | Example app . . . . .     | 5        |
| 1.6      | Settings . . . . .        | 5        |
| 1.7      | Contributing . . . . .    | 6        |
| 1.8      | Change Log . . . . .      | 6        |
| <b>2</b> | <b>API documentation</b>  | <b>7</b> |
| 2.1      | Models . . . . .          | 7        |
| 2.2      | Views . . . . .           | 7        |
| 2.3      | Util . . . . .            | 7        |
| 2.4      | Template tags . . . . .   | 7        |
| 2.5      | Middleware . . . . .      | 7        |



django-cookie-consent is a reusable application for managing various cookies and visitors consent for their use in Django project.

Features:

- cookies and cookie groups are stored in models for easy management through Django admin interface
- support for both opt-in and opt-out cookie consent schemes
- removing declined cookies (or non accepted when opt-in scheme is used)
- logging user actions when they accept and decline various cookies
- easy adding new cookies and seamlessly re-asking for consent for new cookies

The latest version of Django cookie consent is available at <https://github.com/bmihelac/django-cookie-consent/>



## Installation

django-cookie-consent is on the Python Package Index (PyPI), so it can be installed with standard Python tools like `pip` or `easy_install`:

```
$pip install django-cookie-consent
```

## Configuration

1. Add `cookie_consent` to your `INSTALLED_APPS`.
2. Add `django.core.context_processors.request` to `TEMPLATE_CONTEXT_PROCESSORS` if it is not already added.
3. Include `django-cookie-consent` urls in `urls.py`:

```
url(r'^cookies/', include('cookie_consent.urls'))
```

4. Run `syncdb` or `migrate` django management command.

## Main concepts

### Cookie Group

Cookie Group model represents a group of related cookies. For all but required cookie groups, user gives consent or decline their use.

### Versions

Each Cookie Group has current version that is timestamp when last cookie is added. When user accept cookie group, current version is saved in `cookie_consent` cookie.

Versions allows django-cookie-consent app to know if new cookies have been introduced since user given a consent for specific cookie use and to ask them to re-accept new cookies when needed.

### Important attributes:

**varname** Variable name that will be used for this cookie group.

**is\_required** Required cookies are not deleted and user cannot affect them. This could be `sessionid`, `csrftoken` and others. Without this cookies website will not work properly and user can't opt-out.

**is\_deletable** If cookie group is deletable, django-cookie-consent will try to delete cookies in this group when declined or through `CleanCookiesMiddleware` middleware.

### Cookie

Cookie model represent each cookie. Note that `domain` and `path` attributes are important for deleting cookies.

### Saving user selection

User selection regard cookie use are saved in a cookie with default name `cookie_consent`.

Example of `cookie_consent` value could be:

```
optional=-1|social=2013-06-04T03:17:01.421395
```

In above example user declined cookie group with `optional` varname and accepted cookie group `social` with all cookies created before stated timestamp.

### Caching

To avoid hitting database for each request, non required cookies and cookie groups are cached.

## Getting started

### Checking for cookie consent in views

```
from cookie_consent.util import get_cookie_value_from_request

def myview(request, *args, **kwargs):
    cc = get_cookie_value_from_request(request, "mycookies")
    if cc:
        # add cookie
```

Checking if specific cookie in Cookie group is accepted is possible:



```
cc = get_cookie_value_from_request(request, "mycookies", "mycookie1")
```

## Checking for cookie consent in templates

Use `cookie_group_accepted` or `cookie_group_declined` template filters.

```
{% load cookie_consent_tags %}
{% if request|cookie_group_accepted:"analytics" %}
  {# load 3rd party analytics #}
{% endif %}
```

Bot filters takes cookie group varname and optional cookie name with domain. If cookie name with domain is used, format is `VARNAME=COOKIE_NAME:DOMAIN`.

## Checking for 3rd party cookies dynamically

Using `js_type_for_cookie_consent` templatetag for script type attribute would set `x/cookie_consent` thus making browser skip executing this block of javascript code.

When consent for using specific cookies is given, code can be evaluated without reloading page.

```
{% load cookie_consent_tags %}
<script type="{% js_type_for_cookie_consent request "social" "*:google.com" %}" data-
↪varname="social">
  alert("Social cookie accepted");
</script>
```

## Example app

```
cd tests && ./manage.py runserver
```

Username and password for admin are 'admin', 'password'.

## Settings

**COOKIE\_CONSENT\_NAME** name of consent cookie that remembers user choice

Default: `cookie_consent`.

**COOKIE\_CONSENT\_MAX\_AGE** max-age of consent cookie

Default: 1 year

**COOKIE\_CONSENT\_DECLINE** decline value Default: -1

**COOKIE\_CONSENT\_ENABLED** boolean or callable that receives request and return boolean.

IE if you want to enable cookie consent for debug or staff only:

```
COOKIE_CONSENT_ENABLED = lambda r: DEBUG or (r.user.is_authenticated() and r.user.
↪is_staff)
```

Default: `True`

**COOKIE\_CONSENT\_OPT\_OUT** Boolean value represents if cookies are opt-in or opt-out opt-out cookies are set until declined opt-in cookies are set only if accepted

Default: `False`

**COOKIE\_CONSENT\_CACHE\_BACKEND** Alias for backend to use for caching.

Default: `default`

## Contributing

### Code guidelines

- As most projects, we try to follow PEP8 as closely as possible
- Most pull requests will be rejected without proper unit testing

## Change Log

### 0.1.2 (dev)

#### 0.1.1

- tweak admin
- Add `accepted_cookies` template filter
- Add `varname` property to Cookie model
- Add translation catalog

#### 0.1.0

- Initial release

**Models**

**Views**

**Util**

**Template tags**

`cookie_consent`

**Middleware**

`CleanCookiesMiddleware`