localeurl Documentation

Release 2.0.2.post

Joost Cassee, Artiom Diomin and Carl Meyer

December 07, 2016

Contents

Warning: django-localeurl is currently un-maintained (so pull requests will not be reviewed or merged), and its approach has been obsoleted by the introduction of locale-aware URL patterns in Django itself. If you are nonetheless interested in taking over maintenance of django-localeurl, please file an issue to volunteer. Thank you!

The localeurl Django application allows you to specify the language of a page in the URL.

Suppose you have a Django website in multiple languages. Using localeurl, without modifying your URLconfs, you can have URLs like this: `http://www.example.com/nl/company/profile`. Any URLs without a language prefix will be redirected to add the prefix for the default language (or, optionally, the language preferred in the user's browser settings).

Some reasons for using localeurl:

- Search engines will index all languages.
- Every page should have a unique URL. If you feel that different languages means different pages, then each language should get its own unique URL.
- If you don't set the language via the URL, setting the language for the website should be done using a POST request (because it influences subsequent page views, see Django ticket #3651). You might prefer a simple link for changing the language, and localeurl allows this.

You can install localeurl with pip:

```
pip install django-localeurl
```

or install the in-development version:

```
pip install django-localeurl==dev
```

The localeurl code is licensed under the MIT License. See the `LICENSE.txt` file in the distribution.

CHAPTER 1

Documentation

1.1 Installation

This section describes how to install the localeurl application in your Django project.

1.1.1 Prerequisites

The localeurl application requires Django 1.3 or higher and Python 2.6 or 2.7.

1.1.2 Setup

Setup consists of installing the middleware and adding `'localeurl'` to the installed applications list.

1. Add `'localeurl.middleware.LocaleURLMiddleware'` to `settings.MIDDLEWARE_CLASSES`. It must come *before* `'django.middleware.common.CommonMiddleware'` or `settings.APPEND_SLASH` will not work. Make sure Django's built-in `LocaleMiddleware` is **not** in your `MIDDLEWARE_CLASSES` setting; `LocaleURLMiddleware` replaces it and the two will not work together. It must also come after `django.contrib.sessions.middleware.SessionMiddleware` if you plan using the session fallback (see `LOCALEURL_USE_SESSION` below).
2. Add `'localeurl'` to `settings.INSTALLED_APPS`. Because the application needs to replace the standard `urlresolvers.reverse` function, it is important to place it at the top of the list:

```
INSTALLED_APPS = (
    'localeurl',
    ...
)
```

3. If you want to use the view, include the localeurl URLconf module in your project:

```
urlpatterns = patterns('',
    ...
    (r'^localeurl/', include('localeurl.urls')),
    ...
)
```

4. Make sure `settings.LANGUAGE_CODE` or its root language is in `settings.LANGUAGES`. For example, if `LANGUAGE_CODE == 'en-us'` then `LANGUAGES` must contain either `'en-us'` or `'en'`. If you have not changed either option you do not have to do anything.

1.1.3 Configuration

The application can be configured by editing the project's `settings.py` file.

LOCALE_INDEPENDENT_PATHS A tuple of regular expressions matching paths that will not be redirected to add the language prefix. For example, a site with a language selection splash page would add `'^/$'` as a locale independent path match.

Example:

```
LOCALE_INDEPENDENT_PATHS = (
    r'^/$',
    r'^/games/',
    r'^/ajax/',
)
```

LOCALE_INDEPENDENT_MEDIA_URL (default: `True`) Whether paths starting with `settings.MEDIA_URL` (if it is a path, i.e. not a full URL) are considered to be locale-independent.

LOCALE_INDEPENDENT_STATIC_URL (default: `True`) Whether paths starting with `settings.STATIC_URL` (if it is a path, i.e. not a full URL) are considered to be locale-independent.

PREFIX_DEFAULT_LOCALE (default: `True`) Whether to add the prefix for the default language (`settings.LANGUAGE_CODE`). For example, if `LANGUAGE_CODE == 'en'` then the path `/about/` will be passed to the URL resolver unchanged and `/en/about/` will be redirected to `/about/`.

LOCALEURL_USE_ACCEPT_LANGUAGE (default: `False`) Whether to check the `Accept-Language` header from the browser as an intermediate fallback in case no locale is specified in the URL. (The default behavior, preserved for backwards compatibility, is to fallback directly to `settings.LANGUAGE_CODE`).

`LOCALEURL_USE_SESSION` (default: `False`)

Whether to check for a user-selected locale in the Django session as a fallback in case no locale is specified in the URL. If `True`, the `change_locale` view will save the chosen locale to the user's session under the `django_language` session key. When used and available, the session locale takes precedence over the `Accept-Language` header fallback (see `LOCALEURL_USE_ACCEPT_LANGUAGE` above).

A localized URL still takes precedence over a locale in the user's session. Following a localized link such as one generated using the `chlocale` filter won't switch the session's `django_language`; only the `change_locale` view will.

To use this feature, sessions must be in use, and `django.contrib.sessions.middleware.SessionMiddleware` must come *before* `localeurl.middleware.LocaleURLMiddleware` in `settings.MIDDLEWARE_CLASSES`.

LOCALE_REDIRECT_PERMANENT (default: `True`) Whether to use a permanent redirect (301 status code) or temporary redirect (302) when redirecting users from the no-locale version of a URL to the default locale (or the locale specified in their Accept-Language header if `LOCALEURL_USE_ACCEPT_LANGUAGE` is True). If Accept-Language is not used, these redirects will never change (as long as the default locale never changes), so 301 (the default) is a fine choice. If you use Accept-Language you may want to consider switching this to `False`, as the redirect will then be dependent on the user's Accept-Language header.

1.2 Usage

The localeurl application provides a middleware class that sets `request.LANGUAGE_CODE` based on a prefix on the URL path. It stripping off this language prefix from `request.path_info` so that the URLconf modules do not need to change. Existing applications should work transparently with localeurl if they follow the usual Django

convention of using `url` tags in templates and (and the `urlresolvers.reverse` function in Python code) to generate internal links.

Paths without locale prefix are redirected to the default locale, either from `request.LANGUAGE_CODE` (set by a previous language discovery middleware such as `django.middleware.locale.LocaleMiddleware`) or from `settings.LANGUAGE_CODE`. So a request for `/about/` would be redirected to `/fr/about/` if French is the default language. (This behavior can be changed using `settings.PREFIX_DEFAULT_LOCALE`.) Determination of the default locale can also take into account the `Accept-Language` browser header (see `settings.LOCALEURL_USE_ACCEPT_LANGUAGE`), or a previously user-selected locale (see `settings.LOCALEURL_USE_SESSION`, and the `change_locale` view below).

1.2.1 Templates

The application adds one template tag and two filters. Add the following at the top of a template to enable them:

```
{% load localeurl_tags %}
```

The `locale_url` tag

The localeurl application replaces the `urlresolvers.reverse` function to return locale-specific URLs, so existing templates should not need to be changed. To manipulate the language on rendered URLs you can use the `locale_url` tag. This tag behaves exactly like the standard `url` tag, except you specify a language.

Note: In Django 1.3 and later, using the `locale_url` tag from `localeurl_tags` will result in a deprecation warning about changed url tag syntax. To avoid this warning, `{% load locale_url from localeurl_future %}` in your template after you `{% load localeurl_tags %}`. This also requires that you adopt the new url tag syntax.

Example

You can refer to a specific URL in a specified language like this:

```
<a href="{% locale_url "de" articles.views.display id=article.id %}">Show article in German</a>
```

If you are using Django 1.3+ and you loaded `locale_url` from the `localeurl_future` library, you'd need quotes around the view name:

```
<a href="{% locale_url "de" "articles.views.display" id=article.id %}">Show article in German</a>
```

The `chlocale` filter

To add or change the locale prefix of a path use `chlocale`. It takes one argument: the new locale. If the path is locale-independent any prefix on the path will be stripped. This is also the case if `settings.PREFIX_DEFAULT_LOCALE == False` and the locale argument is the default locale.

Examples

To change the language of a URL to Dutch:

```
Please click <a href="{{ help_url|chlocale:"nl" }}">here</a> for Dutch help.
```

This filter can be used to allow users to go to a different language version of the same page. If you have this in your settings file:

```
_ = lambda s: s
LANGUAGES = (
    ('en', _(u'English')),
    ('nl', _(u'Nederlands')),
    ('de', _(u'Deutsch')),
    ('fr', _(u'Français')),
)
TEMPLATE_CONTEXT_PROCESSORS = (
    'django.core.context_processors.request',
    'django.core.context_processors.i18n',
    ...
)
```

... then you can add a language selection menu in templates like this:

```
{% for lang in LANGUAGES %}
    {% ifequal lang.0 LANGUAGE_CODE %}
        <li class="selected">{{ lang.1 }}</li>
    {% else %}
        <li><a href="{{ request.path|chlocale:lang.0 }}">{{ lang.1 }}</a></li>
    {% endifequal %}
{% endfor %}
```

The `rmlocale` filter

You can use the `rmlocale` filter to remove the locale prefix from a path. It takes no arguments.

Example

To remove the language prefix for a URL:

> The language-independent URL for this page is <tt>{{ request.path|rmlocale }}</tt>.

1.2.2 Views

The application supplies a view to change the locale.

The `change_locale` view

Instead of the language selection menu shown in the `chlocale` example above, you can use the `localeurl_change_locale` view to switch to a different language. It is designed to mimic the Django `set_language` redirect view.

When `settings.LOCALEURL_USE_SESSION` is set to `True` (default is `False`), It also records the user-selected locale to the current Django session. The last selected locale will then be used as the default locale when redirecting from paths missing a locale prefix.

Example

This form shows a drop-down box to change the page language:

```
{% load i18n %}

<form id="locale_switcher" method="POST" action="{% url localeurl_change_locale %}">
    <select name="locale" onchange="$('#locale_switcher').submit()">
        {% for lang in LANGUAGES %}
            <option value="{{ lang.0 }}" {% ifequal lang.0 LANGUAGE_CODE %}selected="selected"{% endi
        {% endfor %}
    </select>
    <noscript>
        <input type="submit" value="Set" />
    </noscript>
</form>
```

1.2.3 Sitemaps

Localeurl supplies a `LocaleurlSitemap` class for more convenient creation of sitemaps that include URLs in all available languages, based on this snippet.

To use, just inherit your sitemap classes from `localeurl.sitemaps.LocaleurlSitemap` instead of `django.contrib.sitemaps.Sitemap`, and instantiate one for each language in your sitemaps dictionary.

Example

The following show how might create a multilingual sitemap:

```
from localeurl.sitemaps import LocaleurlSitemap

# example Sitemap
class AdvertisementsSitemap(LocaleurlSitemap):
    def items(self):
        return Advertisement.active_objects.all()

# create each section in all languages
sitemaps = {
    'advertisements-sk': sitemaps.AdvertisementsSitemap('sk'),
    'advertisements-cs': sitemaps.AdvertisementsSitemap('cs'),
}

# add sitemap into urls
urlpatterns = patterns('',
    url(r'^sitemap.xml$', 'django.contrib.sitemaps.views.sitemap', {'sitemaps': sitemaps}),
)
```

1.3 History

1.3.1 Changelog

Tip: (unreleased)

Release 2.0.2: (2015-01-26)

- (Re-)Add import of models in middleware, to ensure monkeypatch of `reverse` happens soon enough. Fixes #5, again.
- Fix detection of locales with >2 letter codes. Fixes #38.

Release 2.0.1: (2013-09-19)

- Change canonical documentation links from http://pythonhosted.org/django-localeurl/ to http://django-localeurl.readthedocs.org

Release 2.0: (2013-09-19)

- BACKWARDS-INCOMPATIBLE: Drop support for Django 1.0 - 1.2; Python 2.4 and 2.5.
- Deprecate `localeurl_future` tag library, always use new url tag syntax.
- Try importing `patterns, url` first from `django.conf.urls`; use `django.conf.urls.defaults` only as Django 1.3 fallback. Fixes #36.
- Don't try to parse referrer if not present. Fixes #33. Thanks Simon Luijk.
- Add support for a session-stored locale fallback. Thanks Sylvain Fourmanoit for report and draft patch, and Alex Marandon for adding tests. Fixes #23.
- Make language-code matching case-insensitive, per RFC. Thanks torgeilo for the report. Fixes #32.
- Fix issue with non-ASCII characters in URLs. Thanks Chris Adams.

Release 1.5: (2011-08-24)

- Added `LOCALE_REDIRECT_PERMANENT` setting; defaults to True, if set to `False` 302 redirects are used rather than 301. Thanks oyvindeh.
- `LOCALE_INDEPENDENT_PATHS` can now be plain string regular expressions, they don't have to be compiled regex objects. Fixes #9.
- Added import of `localeurl.models` to `localeurl.middleware`, to ensure that `reverse` is monkeypatched before any requests are served. Fixes #5. Thanks Antti Kaihola for the report, Andrey Shipilov and jefurii for fix confirmation.
- Added iri_to_uri encoding of non-ASCII redirect URLs. Fixes #13.
- Sorted language codes longest-first to avoid matching e.g. 'pt' before 'pt-br'. Fixes #15. Thanks Roman Barczyński for report and draft patch.
- Added `LOCALE_INDEPENDENT_STATIC_URL` setting to mirror `LOCALE_INDEPENDENT_MEDIA_URL`. This setting defaults to True, so if you want URLs under STATIC_URL to be locale-dependent, you will need to set it to False.
- Fixed middleware redirection when there is a script prefix. Fixes #10. Thanks iro for report and draft patch.
- Added `localeurl_future` template tag library to provide `locale_url` tag that wraps the new Django `url` tag to allow using the new syntax and avoid deprecation warnings under Django 1.3. Fixes #17. Thanks Fabian Büchler for the report.
- Accounted for reverse() receiving kwargs=None. Fixes #18. Thanks Alexander Clausen for report and tests, Joel Ryan for draft patch.
- Prevented change_locale view from stripping query params from `next`. Fixes #7. Thanks Sverre Johansen.

Release 1.4: (2010-03-19)

- Moved localeurl settings from localeurl/__init__.py to localeurl/settings.py.

- Added `LocaleurlSitemap` for easier creation of multilingual sitemaps.
- Added `LOCALEURL_USE_ACCEPT_LANGUAGE` setting to check HTTP Accept-Language header before resorting to `settings.LANGUAGE_CODE` when locale is not specified in URL.
- Switched to 301 permanent redirects for no-locale URL redirect.
- Moved to BitBucket for source code hosting.
- Added the `change_locale` view, contributed by Panos Laganakos.

Release 1.3: (2009-04-06)

- Changed chlocale tag to strip prefix of locale-independent paths.
- Moved the monkey-patching of urlresolvers.reverse to models.py.
- Removed `REDIRECT_LOCALE_INDEPENDENT_PATHS` settings option; this is now the default.

Release 1.2: (2009-01-19):

- Moved the documentation into the source tree. (Based on a blog post by Andi Albrecht.)
- Released version 1.2.

Release 1.1: (2008-11-20):

- Added the `PREFIX_DEFAULT_LOCALE` settings option contributed by Jonas Christian.
- Added `REDIRECT_LOCALE_INDEPENDENT_PATHS` settings option.

Release 1.0: (2008-09-10):

- Added Django 1.0 or higher as a prerequisite.
- Moved to Google Code.

1.3.2 Credits

localeurl was developed by Joost Cassee based on the work by Atli Þorbjörnsson. Contributions by Jonas Christian. Includes code from the django-localize project by Artiom Diomin. Currently maintained by Carl Meyer.

It was partly taken from and partly inspired by discussions on the django-users and django-multilingual mailinglists:

- Atli Þorbjörnsson: Locale from URL Middleware
- Panos Laganakos: creating a multilingual middleware
- Piotr Majewski: multilingual middleware NEW FEATURE!

See also this blog post on internationalisation by Yann Malet that references Atli's code.

The announcement of localeurl on these lists can be found here:

- Announcement on django-users
- Announcement on django-multilingual